UNIVERSITÀ
DI TRENTO
Department of
Information Engineering and Computer Science

Doctoral Programme in
Information Engineering and Computer Science

LEARNING WITHOUT LABELS
REDUCING SUPERVISION IN TRAINING, INFERENCE, AND EVALUATION OF DEEP NEURAL NETWORKS

Alessandro Conti

Advisor
Prof. Elisa Ricci
Università di Trento

Co-Advisor
Prof. Paolo Rota
Università di Trento

April 2025

Abstract

This thesis investigates how the reliance on supervision can be reduced across the entire deep learning pipeline. In the training phase, we explore unsupervised fine-tuning, focusing on Source-Free Unsupervised Domain Adaptation scenarios in visual tasks such as Facial Expression Recognition and video-based Action Recognition, primarily leveraging self-supervision and self-training. At inference, we address the challenge of removing fixed output vocabularies from Vision Language Models by formalizing the tasks of Vocabulary-free Image Classification and Vocabulary-free Semantic Segmentation and by introducing a family of efficient methods that adapt CLIP to the tasks. We also evaluate Large Multimodal Models under a similar constrained scenario, analyzing their predictions, categorizing their mistakes, and proposing tailored solutions to optimize their performance. Finally, we investigate unsupervised evaluation by proposing a framework that uses a Large Language Model and modular tools to automatically generate, execute, and interpret evaluation experiments for Large Multimodal Models without ground-truth labels. By reducing the need for human supervision at every stage of the deep learning pipeline, this thesis contributes toward a more flexible and efficient paradigm for developing and deploying deep neural networks in real-world, data-scarce, and open-ended settings.

Keywords

Fine-tuning with domain shift, Inference without labels, Automatic benchmarking

Contents

1 Introduction **1**
- 1.1 Context 1
- 1.2 Unsupervised training 2
- 1.3 Unsupervised inference 3
- 1.4 Unsupervised evaluation 4
- 1.5 Contributions 5
- 1.6 Publications 7

2 Unsupervised training **9**
- 2.1 Fine-tuning with domain shift in Facial Expression Recognition 9
  - 2.1.1 The problem 9
  - 2.1.2 The proposed approach 11
  - 2.1.3 Experimental results 15
  - 2.1.4 Related work 20
- 2.2 Fine-tuning with domain shift in Action Recognition 22
  - 2.2.1 The problem 22
  - 2.2.2 The proposed approach 25
  - 2.2.3 Experimental results 30
  - 2.2.4 Related work 37

3 Unsupervised inference **39**
- 3.1 Classification with Vision Language Models 39
  - 3.1.1 The problem 39
  - 3.1.2 The proposed setting 41
  - 3.1.3 The proposed approach 44
  - 3.1.4 Experimental results 48
  - 3.1.5 Related work 61
- 3.2 Semantic segmentation with Vision Language Models 63
  - 3.2.1 The problem 63
  - 3.2.2 The proposed setting 65
  - 3.2.3 The proposed approach 66
  - 3.2.4 Experimental results 70
  - 3.2.5 Related work 75
- 3.3 Classification with Large Multimodal Models 80

3.3.1 The problem . . . 80
3.3.2 The proposed setting . . . 82
3.3.3 Experimental results . . . 91
3.3.4 Related work . . . 104

4 Unsupervised evaluation 107
4.1 Benchmarking of Large Multimodal Models . . . 107
4.1.1 The problem . . . 107
4.1.2 The proposed approach . . . 109
4.1.3 Experimental results . . . 116
4.1.4 Related work . . . 120

5 Conclusion 123
5.1 Future directions . . . 124

Bibliography 127

A Extended results of classification with Large Multimodal Models 157

B Extended results of benchmarking of Large Multimodal Models 167

C Detailed prompts of benchmarking of Large Multimodal Models 179

List of Tables

2.1 Results in four domain adaptation settings. *Italic* indicates best among all, and **bold** best among SFUDA ones. Green indicate ours. Note that in "AFE → FER2013", CluP achieves the best result among all methods. 16

2.2 Class-wise accuracy for RAF-DB → FER2013 in SFUDA setting. Our method is indicated in green. 16

2.3 Top-1 accuracy with different pre-trained target backbones. **Bold** indicates best, green indicates ours. 19

2.4 Top-1 accuracy with varying backbone and score filters. **Bold** indicates bets, green indicates ours. 19

2.5 Results on Daily-DA. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50). 32

2.6 Results on UCF-HMDB$_{full}$. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50). 33

2.7 Results on Sports-DA. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50). 34

2.8 Ablation on the relative improvement of the different modules of our proposed DALL-V framework *w.r.t.* CLIP (ViT-B/32). **Bold** indicates best. The + operator indicates ensembling. Results are aggregated by source dataset. 34

3.1 Cluster Accuracy on the ten datasets. Green is our method, gray shows the upper bound. 50

3.2 Semantic Similarity (x100) on the ten datasets. Values multiplied by x100 for readability. Green highlights our method and gray the upper bound. 51

3.3 Semantic IoU on the ten datasets. Green is our method, gray shows the upper bound. 51

3.4 Ablation studies. Metrics are averaged across the ten datasets. Green highlights our setting. **Bold** represents best, underline indicates second best. 54
3.5 Ablation on the candidate filtering operations. Metrics are averaged across the ten datasets. 55
3.6 Ablation on the number retrieved captions. Green represents our selected configuration. Alongside the metrics, we show the number of unique words extracted from captions (*i.e.*, "candidates") and the number of words selected by CaSED for classification (*i.e.*, "selected"). Results are averaged across the ten datasets. 55
3.7 Ablation on CaSED backbone. Results are collected on ImageNet, using CC12M as the text database. Sizes are taken from [103]. 56
3.8 Cluster Accuracy on the ten datasets. Green is our method, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models. 57
3.9 Semantic IoU on the ten datasets. Green is our method, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models. 58
3.10 Semantic similarity on the ten datasets. Green is ours, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models. 59
3.11 Ablation on the databases on ImageNet. Green is our database, Blue shows the top-5 databases of PMD, which we used to create Ours. We also evaluate the semantic similarity of the closest caption to each image in the dataset (*i.e.*, C.C. S-Sim.). **Bold** represents best, while underline second best. YFCC100M* indicates we use the subset of the dataset used in PMD. 60
3.12 Ablation on prompt ensembling. Results are averaged on the ten classification datasets. Green is our configuration. 60
3.13 Computational cost of different methods. Green is our method, gray shows the upper bound. Inference time is reported on batches of size 64, as the average over multiple runs. 61
3.14 Semantic segmentation on PascalVOC-20. Green are our methods. . . 72
3.15 Segmentation on PASCAL Context-59. Green highlights our methods. 73
3.16 Semantic segmentation on ADE20K-150. Green are our methods. . . . 74
3.17 OW results averaged on the grouped datasets. TI stands for text inclusion, LI for Llama inclusion, SS for semantic similarity, and CS for concept similarity. Higher is better, **bold** indicates best. 85
3.18 Summary details of the datasets used in our analyses. 86
3.19 Summary details of the models used in our analyses. 88
3.20 Text inclusion on the ten datasets. Higher is better, **bold** indicates best. 89
3.21 Llama inclusion on the ten datasets. Higher is better, **bold** indicates best. Note that the scores for CLIP closed-world equals the textual inclusion scores. 89

3.22 Semantic similarity on ten datasets. Higher is better, **bold** indicates best. . . . 90
3.23 Concept similarity on ten datasets. Higher is better, **bold** indicates best. 90
3.24 Agreement of LMMs correct predictions across datasets. Low indicates that less than 30% of the models predicted a sample correctly, while high indicates that more than 70% did. . . . 95
3.25 Elo ratings on the ten datasets. Higher scores indicate comparatively better responses from the models. . . . 98

4.1 Ranking of data type group recognition performance from the best (top) to worst (bottom). APEx achieves the same ranking of [246] both when testing each data type independently and aggregating metrics by their group (Avg. types), and when directly querying for the group understanding (Groups). . . . 118
4.2 Summary of the average accuracy achieved by BLIP-2, IDEFICS, and LLaVA across the nine recognition tasks when adding different data type transformations. Accuracy is averaged across the set of experiments designed by APEx. . . . 120

A.1 Gains on the types of model prediction when instructing the models to be more generic/specific, and when using dataset-specific prompts techniques on six datasets, *i.e.*, DTD, FGVCAircraft, Flowers102, Food101, OxfordPets, StanfordCars. . . . 158
A.2 Relative performance variation with the generic/specific prompts on six datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity. . . . 159
A.3 Relative performance variation with dataset-specific prompts. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity. 160
A.4 Gains on the types of model prediction when instructing the models to reason with chain-of-thought, and when using reasoning models on five datasets, *i.e.*, Caltech101, DTD, Flowers102, OxfordPets, UCF101. . . 161
A.5 Relative performance variation with chain-of-thought prompts on five datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity. . . . 162
A.6 Gains on the types of model prediction when instructing the models with multi-label prompts on five datasets, *i.e.*, Caltech101, DTD, Flowers102, OxfordPets, UCF101. . . . 163
A.7 Relative performance variation with multi-label prompts on five datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity. . . . 165
A.8 OW results of reasoning models on ten datasets. Higher is better, **bold** indicates best. Note that the Llama inclusion for CLIP closed-world equals the textual inclusion scores. . . . 166

B.1 We report the complete outputs for the data types experiments associated with APEx when queries about data types are provided as input. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification. 173
B.2 We report the complete outputs for the data types groups experiments associated with APEx when considering queries about data type groups. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification. 174
B.3 We report the complete outputs for the data class experiments associated with APEx when queries about data classes are provided as input. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification. 177

List of Figures

2.1 Comparison between previous works (the left part) and our CluP on cross-domain FER (the right part). Differently from past works, we aim to learn a target model $f^{\mathtt{T}}(\cdot)$ with only source model $f^{\mathtt{S}}(\cdot)$ and unlabeled target data $\{X^{\mathtt{T}}\}$ without the source data $\{X^{\mathtt{S}},\ Y^{\mathtt{S}}\}$, a very likely scenario due to privacy concerns. Our solution, CluP, is the first method on source-free domain adaptation for FER, exploiting self-supervised learning (SSL) to warm up the target feature extractor $g^{\mathtt{T}}(\cdot)$ and a novel cluster-level pseudo-labeling technique. 10

2.2 Our proposed CluP comprises three stages of training, where the first stage produces trustworthy cluster-level pseudo-labels using the source model, the second stage warms up the target model in a self-supervised fashion, and finally the third stage performs the target model training with the refined pseudo-labels. 12

2.3 UMAP visualizations of the features spaces in the RAF-DB → FER2013 setting. 18

2.4 Top-1 accuracy of CluP on FER2013, when adapting from AFE and RAF-DB, with varying thresholds on the confidence (the dashed line) and our cluster purity score (the solid line). 20

2.5 Performance over time on the Daily-DA benchmark. CLIP [204] achieves better results than Unsupervised Domain Adaptation (UDA), Source-Free UDA and video-based Source-Free UDA methods. Our method, DALL-V, is built on top of CLIP and outperforms all prior approaches. 23

2.6 (a) In standard SFUDA setups, a model trained on the source domain is adapted to the unlabeled target dataset using self-supervised signals, such as temporal consistency across frames [268, 270]. (b) VLM approaches (*e.g.*, CLIP [204]) perform zero-shot classification by computing the cosine similarity between visual features and language embeddings. (c) Our proposed SFUDA method fuses predictions from CLIP and the source-trained model, introducing only a minimal number of learnable parameters via adapters. 24

2.7 Overview of the DALL-V pipeline. $\overline{bar}$ denotes the student components, and is used to distinguish them from the teachers. 27

2.8 UMAPs of the feature space on ***UCF-HMDB***$_{full}$ for CLIP (RN50), the source model, and the final DALL-V model. DALL-V efficiently clusters the action classes in the feature space by combining knowledge from the other two models. 36
2.9 Analysis on the number of templates used for the text input *w.r.t.* the accuracy score, on ***Daily-DA*** and ***UCF-HMDB***$_{full}$. 37

3.1 Vision-Language Model (VLM)-based classification (a) assumes a predefined set of target categories, *i.e.*, the vocabulary, while our novel task (b) lifts this assumption by directly operating on the unconstrained language-induced semantic space, without a known vocabulary. $f^v_{\texttt{VLM}}$ and $f^t_{\texttt{VLM}}$ denote the pre-trained vision and text models of a VLM. 40
3.2 Results of our preliminary study, showing the top-1 accuracy when matching semantic descriptions to ground-truth class names in ten different datasets. We compare BLIP-2 (VQA) and BLIP-2 (Captioning) with Closest Caption and Captions Centroid , *i.e.*the average representation of the retrieved captions. We additionally highlight the Upper bound for zero-shot CLIP. Representing the large semantic space as VLDs and retrieving captions from it produces semantically more similar outputs to ground-truth labels *w.r.t.* querying outputs from VQA-enabled VLMs, while requiring 10 times fewer parameters compared to the latter. . . . 43
3.3 Overview of CaSED. Given an input image, CaSED retrieves the most relevant captions from an external database filtering them to extract candidate categories. We classify image -to- text and text -to- text , using the retrieved captions centroid as the textual counterpart of the input image. 46
3.4 Qualitative results on Caltech-101. The first three samples represent success cases, the last two shows failure cases. 53
3.5 Qualitative results on Food101. The first three samples represent success cases, the last two shows failure cases. 53
3.6 Qualitative results on SUN397. The first three samples represent success cases, the last two shows failure cases. 53
3.7 We extend CaSED to Semantic Segmentation following three strategies: (a) a class-agnostic segmenter (SAM) segments all objects, then CaSED labels each mask independently; (b) CaSED provides candidate categories for the image that are fed as input to an open-vocabulary segmentation model (SAN); (c) DenseCaSED, where we directly accumulate visual features from multi-scale patches, and perform CaSED locally. . 68
3.8 Qualitative results of Vocabulary-free Semantic Segmentation methods on ADE20K-150. For visualisation purposes, we visualise only the top-6 most prominent regions. 76
3.9 Qualitative results of Vocabulary-free Semantic Segmentation methods on PASCAL Context-59. For visualisation purposes, we visualise only the top-6 most prominent regions. 77

3.10 Qualitative results of Vocabulary-free Semantic Segmentation methods on PascalVOC-20. For visualisation purposes, we visualise only the top-6 most prominent regions. 78

3.11 We extensively test 13 Large Multimodal Models (LMMs) for Open-World (OW) classification on 10 datasets using four evaluation metrics. We show that LMMs outperform contrastive-based approaches in OW (CaSED [48], and CLIP [205] with image-to-text retrieval) but still lag behind closed-world models with fixed categories (CLIP [205], dashed line). 81

3.12 Per-image examples of model predictions. **Bold** indicates the ground truth class names. For visualization purposes, we show only the predictions with the highest/lowest concept similarity. Blue and red indicate positive and negative Llama inclusion values. 92

3.13 Per-class examples of model predictions. **Bold** indicates the ground truth class names. On the x-axis we report the average LI, and on the y-axis the average CS. For visualization purposes, we show the most frequent concepts predicted for each quadrant. 93

3.14 Types of model predictions per dataset groups. We indicate with blue correct and specific and correct but generic predictions, and with red wrong but specific and wrong and generic mistakes. 94

3.15 Percentage of correct and specific predictions shared between models. Higher values indicate higher agreement. 96

3.16 Average gains per prediction type when asking models to be more generic/specific (a, b), or via dataset-specific prompts (c). Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes. 96

3.17 Types of model predictions when using the generic and specific prompts and the dataset-specific prompts. Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes. 100

3.18 Average gains per prediction type when including chain-of-thought reasoning (a, b, c), or when with reasoning models (d). Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes. 101

3.19 Percentage of model predictions considered wrong in the single-label setting but correct in multi-label. 102

3.20 Types of model predictions when using multi-label prompts. Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes. 103

4.1 Our automatic benchmarking tool has four components: an orchestrator for reasoning, an engine for function execution, a benchmark generator containing image selection and manipulation tools, and a library of LMMs. Given a user query, the orchestrator instantiates a report containing the query and the LMMs to be tested. Then orchestrator receives the report and specifies a first experiment to be executed. The relative benchmark is generated and executed by the engine, with the results collected, discussed by the orchestrator and added to the report. The orchestrator repeats the experimentation loop until it is deemed sufficient to answer the query. In that case, the orchestrator summarizes the report, returning its findings. 110
4.2 Examples of intermediate reports and final output generated with APEx in the case of two different queries. APEx progressively expands the reports as new experiments are conducted and new results are collected. 114
4.3 Summary of the normalized results (vertical axis) achieved by the models across the 27 data type identification tasks, in comparison to those obtained in [246]. The values equal the min-max normalized performance across the set of experiments designed by APEx. 117
4.4 Data classes recognition accuracy averaged over experiments of APEx for BLIP-2, IDEFICS, and LLaVA on the nine recognition tasks. . . 119

List of Abbreviations

ANN Artificial Neural Network. 1, 2

FER Facial Expression Recognition. 3, 5, 9–13, 15, 17, 18, 20, 21, 123

LLM Large Language Model. 3, 6, 7, 80, 84, 97, 107–109, 111, 113, 121, 124, 125, 167

LMM Large Multimodal Model. xi, 3–6, 39, 80–84, 86, 87, 91, 92, 94, 95, 97, 99, 101, 102, 104, 105, 107–109, 111, 112, 115–125, 167

OW Open-World. xi, 80–84, 87, 91, 95, 99, 101, 104, 105, 123, 125

SFUDA Source-Free Unsupervised Domain Adaptation. 2, 5, 9–12, 17, 20, 21, 23–27, 30, 31, 35, 38, 123

SSL Self-Supervised Learning. 2, 11, 17

UDA Unsupervised Domain Adaptation. 2, 5, 9–11, 17, 20–24, 31, 37, 38

VIC Vocabulary-free Image Classification. 4, 6, 39–42, 47, 48, 50, 56, 62–66, 71, 75, 79, 123, 124

VLD Vision Language Databases. x, 40–44, 63, 66, 70

VLM Vision Language Model. 3–6, 9, 22–25, 27, 35, 38–49, 54, 56, 62–70, 75, 87, 104, 115, 116, 121–123

VSS Vocabulary-free Semantic Segmentation. 6, 64–66, 69, 70, 73, 75, 123, 124

Chapter 1

Introduction

1.1 Context

Artificial Neural Networks (ANNs) are a cornerstone of modern machine learning, offering a biologically inspired approach that emulates the structure and function of the human brain. Inspired by the billions of interconnected neurons in the mammalian brain, ANNs aim to replicate the brain's remarkable ability to learn complex tasks such as, *e.g.*, visual recognition [205, 117, 290], speech understanding [9, 207], and natural language processing [190, 244, 209] based on diverse and complex input data. Reflecting this versatility, ANNs have achieved unprecedented success across various fields. In computer vision, they have powered breakthroughs in image classification [206, 229, 290], object detection [263], and scene understanding [117, 3, 140, 161]. In natural language processing, ANNs enable systems to translate languages [250, 10], answer questions [190], and generate coherent text [244, 2].

Two foundational machine learning paradigms lie at the core of how to train neural networks, *i.e.*, supervised and unsupervised learning. Supervised learning is characterized by using labeled data to train models that map inputs to known outputs. The model is presented with input-output pairs and learns to predict the output for new, unseen inputs. Conversely, unsupervised learning operates on unlabeled data, aiming to uncover hidden patterns or structures without predefined targets. Algorithms in this category identify clusters [65, 47, 7], or reduce data dimensionality [1, 179], relying solely on the input features. Recently, this paradigm increasingly plays a fundamental role in pre-training ANNs [206, 290], *i.e.*, establishing useful internal representations before fine-tuning with supervision, and shows potential in reducing or even eliminating the need for annotations during inference or evaluation.

This thesis investigates how such reductions in supervision can be extended and systematized across multiple stages of the machine learning pipeline. Specifically, it explores the role of unsupervised methods in three key areas: fine-tuning models without labeled data, enabling inference without predefined output categories, and developing mechanisms for models to evaluate the performance of other models. By examining these fronts, this work aims to advance our understanding of how ANNs can operate with reduced external guidance.

1.2 Unsupervised training

An increasingly prevalent strategy in modern machine learning is the separation of the training process into two distinct phases: pre-training and fine-tuning. Pre-training involves training a model on a large, generic dataset to acquire broad, transferable representations of the input space. This phase enables the model to learn general features that are not specific to any single task or domain. Fine-tuning, on the other hand, adapts the pre-trained model to a specific target task or dataset by continuing the training process on more focused data, typically with smaller scale and task-specific annotations.

To reduce reliance on labeled data, Self-Supervised Learning (SSL) and weakly supervised learning have become essential in pre-training pipelines [24, 25, 206, 106]. SSL learns from the structure of unlabeled data through tasks like masked prediction or contrastive learning. Weak supervision, in contrast, uses coarse or noisy labels, *e.g.*, image-level tags or prompt-based guidance. These methods enable scalable, annotation-efficient training that later supports effective fine-tuning. However, even with strong pre-training, the transition to task-specific adaptation often remains heavily dependent on labeled data.

Unsupervised fine-tuning encompasses a broad family of techniques aimed at adapting a pre-trained model to a new task or domain without relying on labeled examples. A prominent subset of this space is Unsupervised Domain Adaptation (UDA) [83, 271, 167, 266], where labeled data is available in a source domain but only unlabeled data exists in the target domain. While UDA has traditionally been studied independently of large-scale pre-training, it can also be viewed as a form of unsupervised fine-tuning, *i.e.*, not only with the absence of supervision but also with a distributional shift in instance representations. Other related settings include Source-Free Unsupervised Domain Adaptation (SFUDA) [153, 128, 145, 293], where only the pre-trained model and unlabeled target data are available. These scenarios provide a rich framework for exploring the boundaries and potential of unsupervised fine-tuning.

In this thesis, a dedicated part of the exploration is focused on unsupervised fine-tuning, specifically in the context of Facial Expression Recognition (FER) and video-based Action Recognition. These serve as two concrete case studies to examine how models can be adapted to new tasks without requiring labeled data. While not directly covered in this manuscript, the author has also contributed to two additional works on unsupervised training: one on unsupervised pre-training for multimodal emotion recognition [75], and another on domain adaptation at test-time for temporal action localization [157]. Together, these efforts reinforce the broader relevance and applicability of unsupervised training techniques across vision-centric tasks.

1.3 Unsupervised inference

Inference in machine learning refers to the process by which a trained model makes predictions on new, unseen data. In many real-world applications, this typically involves choosing a label or output from a pre-defined set of categories, *i.e.*, the vocabulary. Traditionally, models trained for classification rely on a fixed classifier head that maps features to this vocabulary, which must be defined at training time. In supervised settings, inference thus assumes that all possible target classes are known and encoded in the model parameters.

This paradigm has shifted with the advent of Vision Language Models (VLMs) [206, 142, 106, 290], which decouple the image encoder from the classifier. In these models, classification is performed by computing similarities between an image embedding and text embeddings representing class descriptions, enabling dynamic, zero-shot classification. This embedding-based approach allows the model to classify inputs using arbitrary, inference-specified categories and forms the foundation for modern unsupervised inference.

With unsupervised inference, we refer to the settings where the aim is to remove the dependency on pre-defined categories. Instead of requiring that a model select from a fixed set of labels, the goal becomes allowing the model to generate or retrieve plausible answers from an open-ended semantic space. This shift enables more flexible and adaptive behaviors, particularly when dealing with tasks in open-world scenarios where the complete label space is unknown or intractable. A further generalization of unsupervised inference emerges with the rise of Large Multimodal Models (LMMs) [136, 39, 253], which integrate the text generation capabilities of Large Language Models (LLMs) [244, 42] with visual encoders. These models enable inference by answering natural language prompts such as "What type of object is in the image?" or "Describe the scene." In this setting, classification becomes a generative process, with models responding in free-form language rather than selecting from a predefined list.

In this thesis, we lay the foundation for the current landscape of unsupervised inference by first analyzing VLMs in open-world settings. We begin by formalizing the task of Vocabulary-free Image Classification (VIC) and introducing metrics to quantify model performance when classifying images without relying on a pre-defined set of categories. We then extend this notion to semantic segmentation, investigating how the task of assigning pixel-level labels can also be re-framed under the same unsupervised inference paradigm. Last, we shift our focus to LMMs, which bring new capabilities by combining language understanding with visual reasoning. In this context, we evaluate how these models perform in open-world classification tasks by prompting them with natural language queries.

1.4 Unsupervised evaluation

Historically, evaluation in machine learning has relied on fixed benchmarks with well-defined splits for training and testing [60, 70, 188, 195]. In the case of early computer vision models–especially before the rise of foundation models–performance was typically assessed on "in-domain" data, *i.e.*, models were trained and tested on closely related distributions, using fixed, manually annotated labels.

With the advent of large-scale pre-trained models such as CLIP [206], evaluation began to shift. Foundation models are trained on diverse, noisy, and often loosely annotated datasets at a massive scale [218, 229]. As a result, traditional in-domain evaluations became insufficient to capture their capabilities. Instead, a new paradigm emerged in which models are evaluated in zero-shot or few-shot settings across a broad range of downstream tasks [70, 45, 188, 92, 233, 258].

This broadening of evaluation objectives reached a new level with the rise of LMMs [3, 58, 136, 12]. These models are no longer assessed purely on classification performance but are evaluated for their high-level cognitive and reasoning abilities [281, 138]. To accommodate this shift, a range of skill-based benchmarks has emerged. Rather than focusing on single tasks, these benchmarks are designed to measure capabilities, *e.g.*, visual perception [77], commonsense reasoning [239, 288, 214], and more.

As evaluation frameworks become increasingly broad and complex, a natural question arises: can we remove the need for ground-truth annotations entirely? In other words, can a model autonomously assess its own or another model's capabilities? Removing labels from the evaluation pipeline introduces a fundamental shift: the metric of performance is no longer tied to a pre-defined target but instead emerges through interaction, reasoning, or iterative experimentation.

This thesis explores this frontier by introducing APEx, Automatic Programming of Experiments, a framework for fully automatic benchmarking of LMMs. Starting from a research question expressed in natural language, APEx uses a language model and a set of modular tools to design and execute relevant experiments. As results accumulate, the system updates a scientific report and decides whether further testing is needed. Once complete, it presents the findings in natural language. APEx redefines evaluation as a dynamic, self-directed process that can operate without ground-truth labels, enabling scalable and adaptive model assessment.

1.5 Contributions

In this thesis, we present our contributions to the field of machine learning, specifically for unsupervised learning, inference, and evaluation. Our contributions are highlighted as follows:

1. We present the first Source-Free Unsupervised Domain Adaptation method for Facial Expression Recognition, which tackles the problem of adapting a model pre-trained on source data to a set of unlabeled samples in the presence of domain shift. Our method exploits self-supervised pre-training to learn visual representations on the target dataset, and then injects task-specific knowledge with a novel and robust cluster-level pseudo-labeling strategy that accounts for in-cluster statistics. Our approach is validated on four common adaptation scenarios, even achieving comparable results with Unsupervised Domain Adaptation methods, *i.e.*, with direct access to the source data.

2. We investigate solving the problem of video-based Source-Free Unsupervised Domain Adaptation by exploiting "web-supervision" from Vision Language Models, driven by the rationale that they contain a rich world prior surprisingly robust to domain-shift. We showcase the effectiveness of integrating a Vision Language Model for Source-Free Unsupervised Domain Adaptation by devising an intuitive and parameter-efficient method that distills the world prior and complementary source model information into a student network tailored for the target. Despite its simplicity, our approach achieves significant improvement over state-of-the-art video-based Source-Free Unsupervised Domain Adaptation methods.

3. We formalize the Vocabulary-free Image Classification task, aiming at removing the need for a pre-defined set of categories, *a.k.a.* the vocabulary, when using Vision Language Models in "generalized" zero-shot settings, *i.e.*, directly assigning to an input image a class that resides in an unconstrained language-induced semantic space. We demonstrate that representing this semantic space through an external vision-language database is the most effective way to obtain semantically relevant content for classifying the image, and we propose Category Search from External Databases (CaSED), a method that exploits a pre-trained Vision Language Model and an external database to address the task in a training-free manner.

4. From the previously proposed formulation of Vocabulary-free Image Classification, we extend its definition to the task of Vocabulary-free Semantic Segmentation, to remove the need for a pre-defined vocabulary also when assigning labels at the pixel level. We identify three approaches to directly extend CaSED for semantic segmentation by (i) employing a segmentation model and classifying parts, (ii) using an open-vocabulary semantic segmentation architecture with an image-specific vocabulary generated with CaSED, and (iii) applying CaSED at different scales and aggregating the predictions to generate a segmentation mask with labels with a model never trained for the task of semantic segmentation.

5. We thoroughly evaluate classification with Large Multimodal Models in a completely open-world setting, *i.e.*, by classifying images directly using natural language (*i.e.*, "What is the main object in the image?"). We first formalize the take and introduce an evaluation protocol, defining various metrics to assess the alignment between predicted and ground truth classes. Then, we evaluate 13 models across 10 benchmarks, encompassing prototypical, non-prototypical, fine-grained, and very fine-grained classes, demonstrating the challenges Large Multimodal Models face in this task. With further analyses based on the proposed metrics, we reveal the types of errors these models make, highlighting challenges related to granularity and fine-grained capabilities. Finally, we show how tailored prompting and reasoning can alleviate these issues.

6. We present APEx, Automatic Programming of Experiments, the first framework for automatic benchmarking of Large Multimodal Models (LMMs). Given a research question expressed in natural language, APEx leverages a Large Language Model (LLM) and a library of pre-specified tools to generate a set of experiments

for the model at hand, and progressively compile a scientific report. The report drives the testing procedure: based on the current status of the investigation, APEx chooses which experiments to perform and whether the results are sufficient to draw conclusions. Finally, the Large Language Model (LLM) refines the report, presenting the results to the user in a natural language. The framework is flexible and extensible as new tools become available. Empirically, we manage to reproduce the findings of existing studies while allowing for arbitrary analyses and hypothesis testing.

1.6 Publications

This thesis is accompanied by a collection of publications that highlight the research contributions made in the field of machine learning, specifically for unsupervised learning, inference, and evaluation. The following is a list of publications in chronological order:

1. **Alessandro Conti**, Paolo Rota, Yiming Wang, and Elisa Ricci. "Cluster-level pseudo-labelling for source-free cross-domain facial expression recognition." British Machine Vision Conference (2022). [52]

2. Giacomo Zara*, **Alessandro Conti***, Subhankar Roy, Stéphane Lathuilière, Paolo Rota, and Elisa Ricci. "The unreasonable effectiveness of large language-vision models for source-free video domain adaptation." International Conference on Computer Vision (2023). [285]

3. **Alessandro Conti**, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. "Vocabulary-free image classification." Advances in Neural Information Processing Systems (2023) [48]

4. **Alessandro Conti**, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. "Vocabulary-free Image Classification and Semantic Segmentation." Under submission at Transaction of Pattern Analyses and Machine Intelligence (2024) [49]

5. **Alessandro Conti**, Enrico Fini, Paolo Rota, Yiming Wang, Massimiliano Mancini, and Elisa Ricci. "Automatic benchmarking of large multimodal models via iterative experiment programming." International Conference on Image Analysis and Processing (2025) [50]

6. **Alessandro Conti**, Massimiliano Mancini, Enrico Fini, Yiming Wang, Paolo Rota, and Elisa Ricci. "On Large Multimodal Models as Open-World Image Classifiers." International Conference of Computer Vision (2025) [51]

While not directly discussed in this thesis, the reader may also find pertinent the following works the author contributed to, especially concerning unsupervised training:

1. Riccardo Franceschini, Enrico Fini, Cigdem Beyan, **Alessandro Conti**, Federica Arrigoni, and Elisa Ricci. "Multimodal emotion recognition with modality-pairwise unsupervised contrastive loss." International Conference on Pattern Recognition (2022). [75]
2. Benedetta Liberatori, **Alessandro Conti**, Paolo Rota, Yiming Wang, and Elisa Ricci. "Test-time zero-shot temporal action localization." Computer Vision and Pattern Recognition (2024). [157]

Chapter 2

Unsupervised training

While pre-training strategies have increasingly embraced unsupervised and weakly supervised approaches, fine-tuning remains largely dependent on labeled data. This chapter mainly focuses on how models can be adapted to downstream tasks without supervision in the fine-tuning stage. In particular, it explores Unsupervised Domain Adaptation (UDA) and its more challenging variant, Source-Free Unsupervised Domain Adaptation (SFUDA), where access to labeled source data is no longer assumed. The chapter presents two contributions within this context. The first introduces a novel SFUDA method for Facial Expression Recognition (FER), combining self-supervised representation learning with a robust pseudo-labeling strategy that operates at the cluster level. The second addresses video-based SFUDA by condensing the prior knowledge embedded in a Vision Language Model (VLM) and in a source model into a compact student network, enabling effective adaptation with minimal supervision.

2.1 Fine-tuning with domain shift in Facial Expression Recognition

2.1.1 The problem

Facial Expression Recognition (FER) [224, 215, 251, 90] refers to the task of automatically inferring the emotional state of a person from a facial image, which supports multiple application fields, such as assistive robotics and security monitoring. However, each individual shows their emotional state differently according to their personal traits or complicated cultural/ethical factors [23]. Such heterogeneity in the data space remains one of the main challenges for a generalizable model for FER. In the last twenty

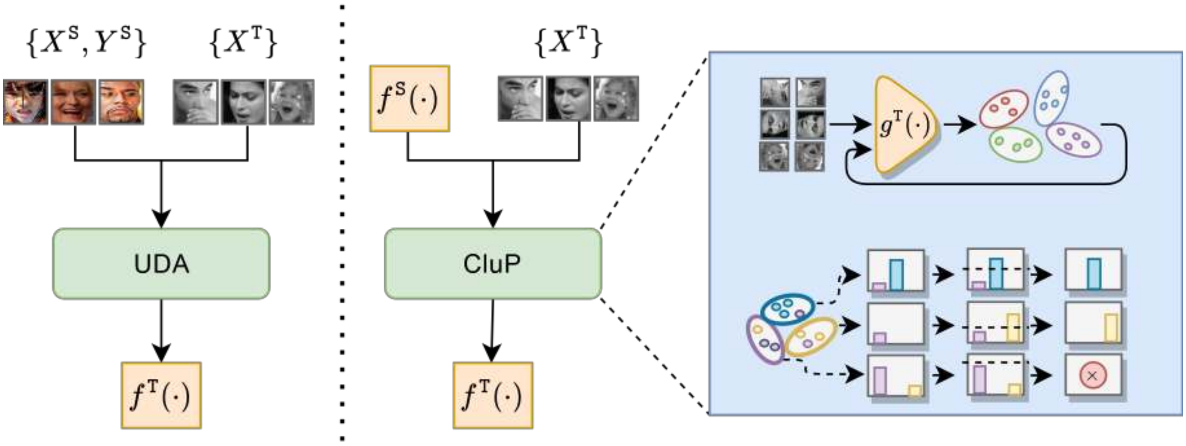


Figure 2.1: Comparison between previous works (the left part) and our CluP on cross-domain FER (the right part). Differently from past works, we aim to learn a target model $f^{\mathrm{T}}(\cdot)$ with only source model $f^{\mathrm{S}}(\cdot)$ and unlabeled target data $\{X^{\mathrm{T}}\}$ without the source data $\{X^{\mathrm{S}}, Y^{\mathrm{S}}\}$, a very likely scenario due to privacy concerns. Our solution, CluP, is the first method on source-free domain adaptation for FER, exploiting self-supervised learning (SSL) to warm up the target feature extractor $g^{\mathrm{T}}(\cdot)$ and a novel cluster-level pseudo-labeling technique.

years, the efforts to improve such technologies have been mostly split between collecting larger and more diverse datasets [284, 183] and advancing learning algorithms for improving generalization capability in the wild [295, 90]. Many recent techniques for FER exploit the attention mechanism [181, 256, 6, 196, 69], while some other works learn uncertainty via feature mixup [295], or improve feature representations by replacing the pooling layers to reduce padding erosion [224].

Recent works often frame the problem from an Unsupervised Domain Adaptation (UDA) perspective where labels of the target samples are not available [105, 147, 148]. For example, in [149], Li *et al.* introduce a novel loss function to preserve feature locality despite the domain shift. Such loss also organizes facial expressions according to their intensity in the embedding space. A more recent method [31] exploits facial landmarks and holistic features to adapt to the target domain with adversarial learning applied to graphs.

While all these methods improve the adaptability of FER models across data distributions, the source data is required during adaptation. However, when dealing with facial images, the source data might not be available due to the increasingly stringent regulations concerning privacy. Therefore, we are motivated to address the more challenging problem of Source-Free Unsupervised Domain Adaptation (SFUDA) for FER, given only the availability of the source pre-trained model (see fig. 2.1). To the best of our knowledge, we are the first to propose a domain adaptation solution for FER that works without the source facial data, embracing a privacy-preserving learning paradigm as the source data can remain private.

Our proposed method, CluP (**Clu**ster-level **P**seudo-labeling), makes use of Self-Supervised Learning (SSL) on the target data and proposes a novel cluster-level pseudo-labeling technique. Pseudo-labeling for UDA often extends the source model to the target domain using the source confidence to select the best target training inputs [294, 170]. However, the computation of confidence requires supervised training, which is only possible in UDA with access to the source data. In the case of SFUDA, as the domain gap increases, one can expect a degrading representation capability of the source model on the target domain. Recent advances in SSL show that a good data representation can be learned without annotated labels [25, 32, 91]. In this chapter, we propose to exploit SSL techniques for a good starting feature representation for the target model and further propose to improve the reliability of pseudo-labels with our newly introduced *cluster purity*, *i.e.*, the local statistics of target samples that are clustered within the feature space expressed by the source model. We validate CluP on a set of cross-domain FER benchmarks and prove its advantageous performance in terms of classification accuracy.

We summarize our contributions as follows:

- We present CluP, the first method addressing SFUDA for FER, exploiting SSL to foundation our target model.
- CluP introduces a novel cluster-level pseudo-labeling scheme to improve the pseudo-labels' reliability based on in-cluster attributes that deviate from traditional methods using pseudo-labeling with confidence thresholds.
- We demonstrate that CluP surpasses competing methods for SFUDA and is comparable with UDA techniques on several FER adaptation benchmarks.

2.1.2 The proposed approach

The traditional closed-set UDA problem setting allows access to the annotated source dataset $\mathcal{D}^{\text{S}} = \{\mathbf{x}_i^{\text{S}}, y_i^{\text{S}}\}_{i=1}^{M^{\text{S}}}$, and a target dataset $\mathcal{D}^{\text{T}} = \{\mathbf{x}_i^{\text{T}}\}_{i=1}^{M^{\text{T}}}$ without annotations, where the target domain shares the same label space with the source, *i.e.*$\mathcal{Y}^{S} = \mathcal{Y}^{\text{T}} = \{1,\ ...,\ N\}$.

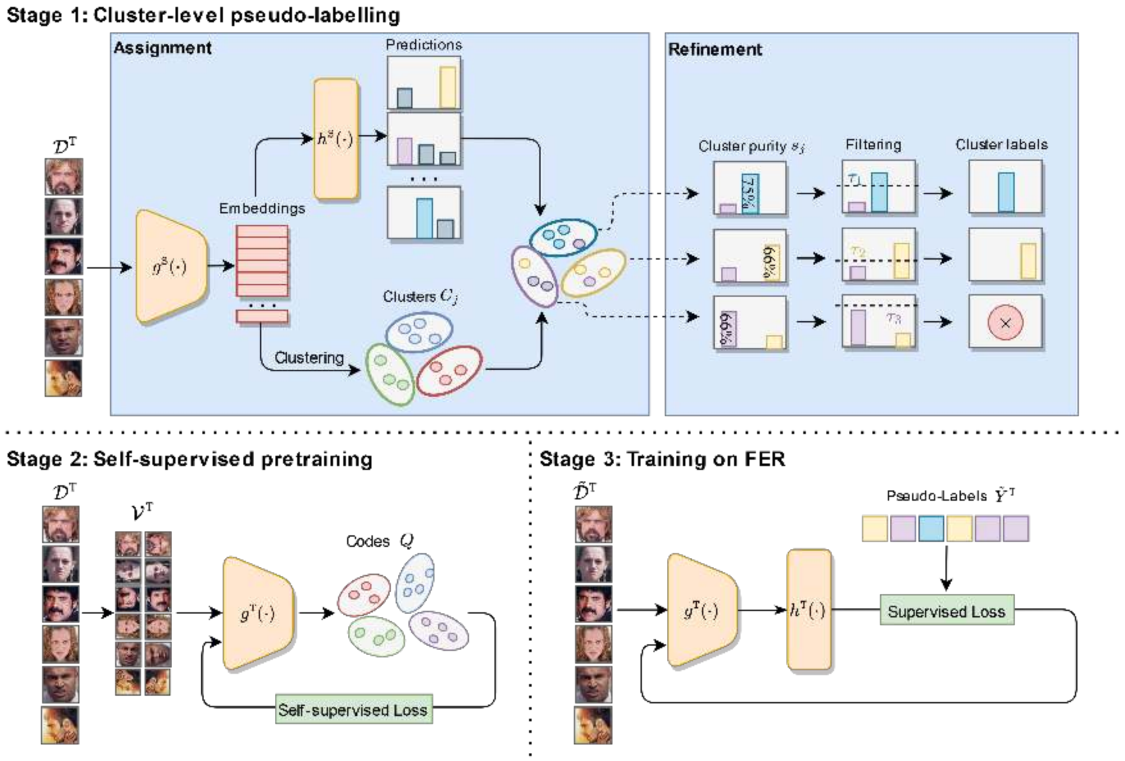


Figure 2.2: Our proposed CluP comprises three stages of training, where the first stage produces trustworthy cluster-level pseudo-labels using the source model, the second stage warms up the target model in a self-supervised fashion, and finally the third stage performs the target model training with the refined pseudo-labels.

Differently, the SFUDA protocol does not allow the access to the source dataset $\mathcal{D}^{\mathrm{S}}$, but only to a trained source model $f^{\mathrm{S}}(\cdot) : \mathcal{X}^{\mathrm{S}} \to \mathbb{R}^N$, which consists of a feature extractor $g^{\mathrm{S}}(\cdot) : \mathcal{X}^{\mathrm{S}} \to \mathbb{R}^Z$ and a classifier $h^{\mathrm{S}}(\cdot) : \mathbb{R}^Z \to \mathbb{R}^N$, where Z is the feature dimension.

Our proposed method CluP tackles the problem of SFUDA for FER. As illustrated in fig. 2.2, CluP follows a three-stage training strategy where the first two can run in parallel: the first stage produces more trustworthy cluster-level pseudo-labels $\{\tilde{y}_i^{\mathrm{T}}\}_{i=1}^{\tilde{M}^{\mathrm{T}}}$ for a subset of $\tilde{M}^{\mathrm{T}}$ target samples $\tilde{\mathcal{D}}^{\mathrm{T}} = \{\mathbf{x}_i^{\mathrm{T}}\}_{i=1}^{\tilde{M}^{\mathrm{T}}}$ by exploiting the available $f^{\mathrm{S}}(\cdot)$ and our proposed cluster purity for pseudo-label refinement (described in section 2.1.2), while in the second stage, a target feature extractor $g^{\mathrm{T}}(\cdot)$ is learned in a self-supervised fashion (described in section 2.1.2). During the third stage, $g^{\mathrm{T}}(\cdot)$ is extended with a classifier $h^{\mathrm{T}}(\cdot)$ and the whole network is trained with the subset of target samples $\tilde{\mathcal{D}}^{\mathrm{T}}$ accompanied by their refined pseudo-labels (described in section 2.1.2).

Cluster-level Pseudo-labeling

Pseudo-labels produced by source models and filtered by confidence are often unreliable, particularly when the domain gap between source and target is large. CluP exploits a clustering technique to group samples with similar characteristics (*i.e.*, *assignment*) and then uses a purity metric based on the source classifier to select the most reliable clusters (*i.e.*, *refinement*).

Cluster pseudo-label assignment. First, the target features are extracted $\{\mathbf{z}_i^{\mathtt{T}}\}_{i=1}^{M^{\mathtt{T}}} \in \mathbb{R}^Z$ using the source feature extractor $g^{\mathtt{S}}(\cdot)$. Second, we cluster the features using K-means clustering, resulting in a set of clusters $\{C_j\}_{j=1}^{K}$ Since FER often deals with highly unbalanced datasets, we perform over-clustering and consider $K \gg N$, to increase the chances that even minor classes can be expressed with some clusters. Leveraging the pseudo-labels predicted by the source model $\tilde{y}_i^{\mathtt{T}} = h^{\mathtt{S}}(\mathbf{z}_i^{\mathtt{T}})$ we assign to each cluster C_j a pseudo-label $\tilde{y}_j^{\mathtt{T}}$ that represents the majority-voted pseudo-label within each cluster.

Cluster pseudo-label refinement. As each cluster C_j should contain similar elements in the learned feature space, we might expect their pseudo-labels to expose a one-class distribution. Unfortunately, this is often not the case. However, a subset of clusters detaining a certain pseudo-label agreement can be defined using what we named as *cluster purity.*

Let us consider $m_j^{\mathtt{T}}$ as the cardinality of the j-th cluster, where $M^{\mathtt{T}} = \sum_{j=1}^{K} m_j^{\mathtt{T}}$. We define the *cluster purity* score s_j for each cluster C_j as the percentage of pseudo-labels $\{\tilde{y}_i^{\mathtt{T}}\}_{i=1}^{m_j^T}$ that agree with their cluster-level label $\tilde{y}_j^{\mathtt{T}}$:

$$s_j = \frac{\sum_{i=1}^{m_j^T} \mathbb{1}\left\{\mathbf{x}_i^{\mathtt{T}} \in C_j : \tilde{y}_i^{\mathtt{T}} = \tilde{y}_j^{\mathtt{T}}\right\}}{m_j^T}. \tag{2.1}$$

Given s_j per cluster, we can further refine the target dataset by keeping clusters with a *cluster purity* score higher than a threshold τ, *i.e.*, the more reliable clusters, for training the target model. Considering that each class might exhibit a different distribution, we design the *cluster purity* threshold τ to vary according to its category. Specifically, for the set of clusters that correspond to the same pseudo-label category $\{C_j\}_{\tilde{y}_j^{\mathtt{T}}=n}$ where $n \in \mathcal{Y}$, we select the Q-th percentile of their purity scores to serve as the threshold τ_n. Q is empirically set, and related experimental details are reported in section 2.1.3.

After the cluster refinement, only clusters whose *cluster purity* score is higher than the threshold corresponding to its category remain in the reduced target dataset $\tilde{\mathcal{D}}^{\mathtt{T}}$ and will be used for training the final target model $f^{\mathtt{T}}(\cdot)$.

Self-supervised pretraining

The pretraining of the target model is a delicate and important phase where the choice of the training method for warming up the backbone leads to relevant fluctuations in performance (see section 2.1.3). In the specific, we noticed that a pretraining relying on self-supervision largely outperforms a model initialized with source weights. For this reason, inspired by SwAV [25], CluP exploits self-supervision on the target dataset to learn an initial feature extractor $g^{\mathtt{T}}(\cdot)$.

CluP performs clustering of the sample data while enforcing the consistency between cluster assignments produced for different augmentations of the same sample. First, target features $\{g^{\mathtt{T}}(\mathbf{x}_i^{\mathtt{T}})\}_{i=1}^{M^{\mathtt{T}}}$ are grouped according to a similarity metric to retrieve N^P learnable prototypes $P = \{p_i\}_{i=1}^{N^P}$ and a set of codes $\{q_i^{\mathtt{T}}\}_{i=1}^{M^{\mathtt{T}}}$ where each sample is assigned to. Then, codes $\{q_i^{\mathtt{T}}\}_{i=1}^{M^{\mathtt{T}}}$ are used as targets to learn the optimal mapping to $\{g^{\mathtt{T}}(\mathbf{x}_i^{\mathtt{T}})\}_{i=1}^{M^{\mathtt{T}}}$ by minimizing:

$$\mathcal{L}_c(\mathbf{x}_i, \mathbf{q}_i) = -\sum_{n=1}^{N^P} \mathbf{q}_i^{(n)} \log(\mathbf{p}_i^{(n)}) \tag{2.2}$$

where $\mathbf{q}$ is the one-hot vector of q and $\mathbf{p}$ is the softmax of the dot product of $g^{\mathtt{T}}(\mathbf{x}_i^{\mathtt{T}})$ and the cluster prototypes P.

By treating each sample as a class ($i.e. M^{\mathtt{T}} = N$), contrastive learning aims to learn a feature extractor $g^{\mathtt{T}}(\cdot)$ invariant to data augmentations. For each target image $\mathbf{x}_i^{\mathtt{T}}$, we generate an arbitrary number N^J of “views” by means of augmentation, $i.e. \{\mathbf{v}_{ij}^{\mathtt{T}} = t_j(\mathbf{x}_i^{\mathtt{T}})\}_{i=1}^{M^{\mathtt{T}}}{}_{j=1}^{N^J}$ with $t_j(\cdot) \sim \mathcal{T}$. Features extracted from views $\{g^{\mathtt{T}}(\mathbf{v}_{ij}^{\mathtt{T}})\}_{i=1}^{M^{\mathtt{T}}}{}_{j=1}^{N^J}$ instead of from inputs $\{g^{\mathtt{T}}(\mathbf{x}_i^{\mathtt{T}})\}_{i=1}^{M^{\mathtt{T}}}$ are then clustered. The feature extractor $g^{\mathtt{T}}(\cdot)$ aims to optimize for a “swapped” assignment problem between pairs of views $(j, k) \in \{1, ..., N^J\}$ of the same input $i \in \{1, ..., M^{\mathtt{T}}\}$:

$$\mathcal{L}_{swapped}((\mathbf{v}_{ij}, \mathbf{q}_{ij}), (\mathbf{v}_{ik}, \mathbf{q}_{ik})) = \mathcal{L}_c(\mathbf{v}_{ij}, \mathbf{q}_{ik}) + \mathcal{L}_c(\mathbf{v}_{ik}, \mathbf{q}_{ij}) \tag{2.3}$$

We minimize $\mathcal{L}_{swapped}$ for all the pairs generated from $\mathcal{D}^{\mathtt{T}}$ to get our pretrained $g^{\mathtt{T}}(\cdot)$.

Finally, the whole model is trained by alternating between clustering features and minimizing eq. (2.3). To work online, clustering is reformulated as an optimal transport problem (as in [8]) and is applied only to the features in a batch.

Training on FER

Finally, the target model (*i.e.* $f^{\mathrm{T}}(\cdot)$) is obtained training the pseudo-labeled subset $\tilde{\mathcal{D}}^{\mathrm{T}}$ (as detailed in section 2.1.2) by using the self-supervised feature extractor $g^{\mathrm{T}}(\cdot)$ (as detailed in section 2.1.2) and a new classifier $h^{\mathrm{T}}(\cdot)$. The model is trained with supervised cross-entropy loss between $\{\tilde{\mathbf{y}}_i^{\mathrm{T}}\}_{i=1}^{\tilde{M}^{\mathrm{T}}}$ and the prediction $\{f^{\mathrm{T}}(\mathbf{x}_i^{\mathrm{T}}\}_{i=1}^{\tilde{M}^{\mathrm{T}}}$ as in eq. (2.4):

$$\mathcal{L}_{CE}^{T} = -\frac{1}{\tilde{M}^{\mathrm{T}}} \sum_{i=1}^{\tilde{M}^{\mathrm{T}}} \tilde{\mathbf{y}_i^{\mathrm{T}}} \, \log f^{\mathrm{T}}\left(\mathbf{x}_i^{\mathrm{T}}\right) \tag{2.4}$$

where $f(\cdot)$ already includes a softmax function for normalizing the network logits into a probability distribution.

2.1.3 Experimental results

We compare our method against the state-of-the-art for cross-domain FER with a set of benchmark datasets. We introduce our experimental setup and then present the main comparison, followed by an extensive ablation study to justify our design choices.
Datasets. Following [31], we use AFE [31] and RAF-DB [149] as our source datasets, and ExpW [297] and FER2013 [85] as the target datasets.

- **AFE** [31] contains 54,901 images of thousands of Asian individuals, collected from Asian movies. This dataset addresses cross-culture domain adaptation, as the other datasets in our experiment involve mainly European and American people.
- **RAF-DB** [149] contains 29,672 facial images from the web of thousands of individuals. We use RAF-DB as one of our source domains as it works as a counterpart of AFE.
- **ExpW** [297] contains 91,793 faces downloaded from Google Images, representing a large-scale in-the-wild scenario with diverse ethnic groups and facial poses.
- **FER2013** [85] is a large-scale dataset collected with the Google Images Search API, containing 35,887 gray images of low resolution. We use FER2013 as one of our target domains to show cross-color domain adaptation.

Performance metric. To evaluate the performance of our method, we consider traditional top-1 classification accuracy. In addition, we also provide class-wise accuracy in our ablations to demonstrate how our method performs on different classes.

	Method	AFE → ExpW	AFE → FER	RAF → ExpW	RAF → FER
UDA	ICID [105]	54.85	46.44	68.52	53.00
	DFA [309]	62.53	36.88	47.42	47.88
	LPL [149]	54.51	49.82	68.35	53.61
	DETN [147]	58.41	45.39	43.92	42.01
	FTDNN [287]	55.29	48.58	68.08	53.28
	ECAN [148]	62.52	46.15	48.73	50.76
	CADA [169]	58.50	48.61	63.74	54.71
	SAFN [265]	55.17	50.07	68.32	53.31
	SWD [133]	56.56	51.84	65.85	53.70
	AGRA [31]	*65.03*	51.95	*69.70*	*54.94*
SFUDA	SHOT-IM [153]	53.52	49.51	53.13	49.44
	SHOT [153]	54.12	49.44	53.51	49.36
	CluP (DeepClusterV2)	62.56	50.47	65.43	**53.83**
	CluP (SwAV)	**65.00**	***52.51***	**66.60**	53.71

Table 2.1: Results in four domain adaptation settings. *Italic* indicates best among all, and **bold** best among SFUDA ones. Green indicate ours. Note that in "AFE → FER2013", CluP achieves the best result among all methods.

Method	Surprise	Fear	Disgust	Happiness	Sadness	Anger	Neutral
SHOT-IM [153]	28.29	**45.05**	9.86	75.97	56.12	40.66	71.96
SHOT [153]	28.18	43.24	10.25	75.59	53.53	40.18	74.37
CluP (DeepClusterV2)	29.44	**45.05**	2.83	**83.15**	**77.70**	34.72	**76.65**
CluP (SwAV)	**37.89**	30.63	**13.57**	80.72	50.85	**44.51**	74.49

Table 2.2: Class-wise accuracy for RAF-DB → FER2013 in SFUDA setting. Our method is indicated in green.

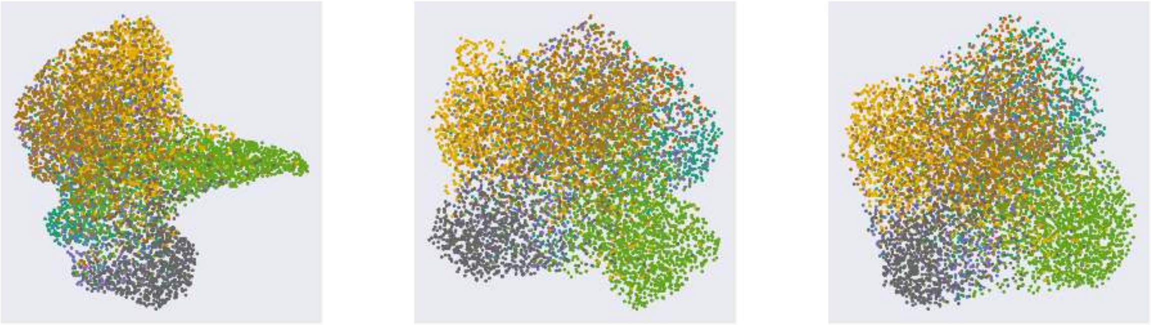
Implementation details. We implement our method using PyTorch and PyTorch Lightning, and run all the experiments on NVIDIA A100 GPUs. We pretrain ResNet18 for FER as our source model, while we perform the SSL on the target domain using the `solo-learn` library [57] for 1000 epochs with SGD and a cosine annealing scheduling policy. When performing cluster-level pseudo-labeling, we use a large K for K-means to address imbalanced datasets. We consider $K = 1000$ for AFE → ExpW and RAF-DB → FER2013, and $K = 250$ for the others. We set the Q-th percentile per class to threshold the cluster purity, where Q is usually set to large values, depending on the adaptation setup. In detail, we use $Q = 0.9$ for AFE → ExpW and AFE → FER2013, while $Q = 0.7$ and $Q = 0.8$ for RAF-DB → ExpW and RAF-DB → FER2013. Our final target model is trained for 50 epochs using SGD, using a cosine annealing scheduling policy.

Comparisons

To the best of our knowledge, CluP is the first method to tackle SFUDA for FER, so we mainly compare state-of-the-art methods in the less restrictive UDA setting. To extend the comparison, we also report the results of a couple of general-purpose methods for SFUDA, which we re-purposed for FER. CluP can be applied seamlessly to an arbitrary SSL method, to this end we report two versions where we apply different self-supervised pretraining on the target domain using SwAV [25] and DeepClusterV2 [24, 25].

Table 2.1 shows the classification accuracy of competing methods under different domain adaptation settings. Compared among SFUDA methods, our method with SwAV as self-supervised pretraining always performs better than SHOT by over ten points in most benchmarks. The same advantage holds when we adapt from the two source domains to FER2013, with a total improvement of +4%. More interestingly, CluP demonstrates comparable adaptation performance even when compared with UDA methods, which have access to the source data. In particular, when we adapt from AFE to FER2013, our CluP scores the best performance among all methods.

For an in-depth investigation of how SFUDA methods perform on FER, we present the class-wise accuracy when adapting from RAF-DB to FER2013 in Table 2.2. Noticeably, CluP consistently adapts better among all classes compared to SHOT and SHOT-IM, where for some classes, *e.g.*, Surprise, Happiness, and Sadness, the improvement is greater than +5%. We also notice that for minor classes under the SFUDA setting, *e.g.*, Disgust, the classification accuracy is much lower compared to other major classes mostly due to the limited samples for expressing the class in the target domain under a large domain gap.



(a) SHOT (b) DeepClusterV2 (Ours) (c) SwAV (Ours)

Figure 2.3: UMAP visualizations of the features spaces in the RAF-DB → FER2013 setting.

Qualitative Result. In fig. 2.3, we present the UMAP visualization of our methods with the target model pre-trained with two self-supervised methods, *i.e.*, DeepClusterV2 and SwAV, in the RAF-DB → FER2013 setting and compare them with SHOT. SHOT shows a more peculiar shape compared to our models, as it fine-tunes the source model to the new domain, thus having a tighter relationship with the source data. The inherited space indirectly constrains the target model from imitating the source domain when molding the target domain. Therefore, starting from the source model can hinder the adaptability to the new domain in situations of extreme domain shift. On the contrary, our method's UMAPs seems to fit better to the target domain. While similar, these two embedding spaces present subtle differences. Visually, the DeepClusterV2 space manages to separate emotions better.

Ablation studies

We present a thorough analysis of the main design choices of CluP. We first investigate different pre-trained networks to validate the effectiveness of the self-supervised pre-training of the target model. We then compare our novel cluster purity score against the traditional confidence score to justify its advantages in providing more reliable pseudo-labels. Finally, we examine different combinations of our proposed elements and show how they impact the final adaptation performance on FER.

Does the self-supervised pretrained backbone work better? To understand how each pre-trained backbone model serves as the target model, we experiment with four options including (i) a model pre-trained on ImageNet, (ii) the source model, *i.e.*,

Dataset	ImageNet	Source	DeepClusterV2	SwAV
ExpW	54.45	66.80	60.54	**69.13**
FER2013	36.38	56.99	58.10	**60.90**

Table 2.3: Top-1 accuracy with different pre-trained target backbones. **Bold** indicates best, green indicates ours.

Backbone	Score	AFE → ExpW	AFE → FER	RAF → ExpW	RAF → FER
Source	Conf.	56.43	48.36	59.79	50.47
Source	Purity	56.54	47.34	61.18	**54.29**
SwAV	Conf.	62.88	51.27	63.22	50.68
SwAV	Purity	**65.00**	**52.51**	**66.60**	53.71

Table 2.4: Top-1 accuracy with varying backbone and score filters. **Bold** indicates bets, green indicates ours.

“Source”, and (iii) the DeepClusterV2 and (iv) the SwAV self-supervised models. For all models, we train a linear classifier on top of their frozen feature extractor with ground-truth target labels. As shown in table 2.3, the self-supervised pre-training on the target domain using SwAV scores the *best* classification accuracy on the two target datasets. DeepClusterV2 demonstrates less consistent improvements over the source model on the two target domains, with +1.1% improvement on the FER2013 dataset, but −6.3% on ExpW. This might be due to the superiority of SwAV in learning discriminative feature representations over DeepClusterV2.

Does cluster purity perform better than confidence? We compare our novel cluster purity score to the traditional confidence to prove its capability of providing more reliable pseudo-labels. We also show the impacts of different thresholds on Q ranging from 0.5 and 0.9 on the adaptation performance. Figure 2.4 shows the top-1 accuracy of CluP on FER2013, when adapting from AFE (the green plots) and RAF-DB (the orange plots), with varying thresholds on the confidence (the dashed line) and our cluster purity score (the solid line). We can observe a general increasing tendency of the accuracy as the threshold value increases, as more reliable pseudo-labels are selected due to a stricter criterion. Our cluster purity consistently outperforms confidence at all threshold values. Specifically, when adapting from RAF-DB to FER2013, cluster purity outperforms confidence at the threshold of 80% by more than +3%.

How do all the components interact with one another? We show how different pre-trained target models and pseudo-label criteria incrementally impact the perfor-

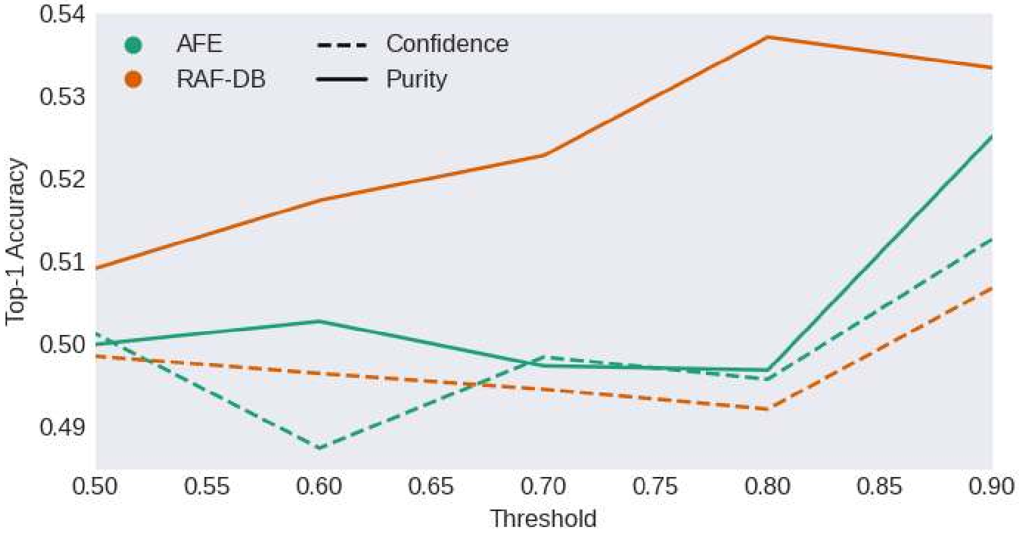


Figure 2.4: Top-1 accuracy of CluP on FER2013, when adapting from AFE and RAF-DB, with varying thresholds on the confidence (the dashed line) and our cluster purity score (the solid line).

mance of our proposed method. We consider two backbones, Source and SwAV, and two pseudo-label criteria, confidence and cluster purity. We present the classification accuracy under different adaptation setups in table 2.4. Regarding the target model, self-supervised pre-training, *i.e.*, SwAV, outperforms the source model for most of the adaptation setups, regardless of the usage of either confidence or cluster purity score for pseudo-label refinement. When our proposed cluster purity is applied, we observe a consistent improvement of about +3% over all the adaptation setups with a cluster-based self-supervised feature extractor. On the other hand, when tested on the source model its advantages are not stable.

2.1.4 Related work

In the following, we present works on UDA methods for FER and some general-purpose SFUDA solutions.

Unsupervised Domain Adaptation for FER. As a consequence of the domain bias, quite prominent among FER datasets, some works focus on domain adaptation [105, 309, 149, 147, 287, 148, 31]. In [105], Ji *et al.* apply late fusion on the outputs of two channels that learn intra-category and inter-category similarities of facial expressions. The authors of [149] introduce a locality-preserving loss that draws samples of the same class closer. They also notice that neighboring samples in the embedding space present similar emotional intensities. DETN [147] applies two variations of the Maximum Mean Discrepancy to quantify the divergence between the domains and to re-weight the class-wise source distribution to match the target. The authors extend the work in [148], where they consider the differences in the conditional distributions. Differently from the above works, AGRA [31] focuses on the well-established approach of adversarial domain adaptation, leveraging facial landmarks alongside facial images. They introduce two specialized graph neural networks while jointly considering the domain feature distributions, the local features (*i.e.*, landmarks), and the holistic features, achieving the best results on many benchmarks.

Compared to previous works, we consider a stricter setting where the source data is unavailable. Due to privacy issues, we argue that methods for understanding human behavior do not always have access to the source data. For this reason, we introduce a novel method for FER that adapts to a target domain in a source-free manner.

Source-Free Unsupervised Domain Adaptation. As a natural evolution of UDA, SFUDA has gained increasingly more interest from the community [153, 95, 128, 145, 293, 154]. The setting represents a more complex but realistic scenario of UDA, where source data is unavailable. Some works resort to entropy-minimization losses to adapt to the target domain without labels. For example, SHOT [153] employs an entropy loss alongside a classification loss on pseudo-labeled samples to adapt the network to the target domain. The work has been extended in [155] introducing an auxiliary head that solves relative rotation. Differently from the above, the authors of [95] frame the problem from an image translation perspective and translate the target images to the source style using only the source model. In [272], they perform self-training with a loss function that considers the intrinsic structure of the target domain via nearest neighbors. In the proposed work, we do not impose any constraint on the loss function, our refinement step works on source clusters, and we propose a novel score function to select the best samples to train on the target domain. Other works address open-set or universal domain adaptation [126, 125] without access to the source data.

Unlike previous works, our model does not rely solely on the source model but is constructed based on independent self-supervised training on the target data. Moreover, we refine pseudo-labels by reducing unreliable samples using a novel decision metric at the cluster level based on cluster purity.

2.2 Fine-tuning with domain shift in Action Recognition

2.2.1 The problem

Tasks involving video analysis, such as action recognition, have been extensively studied within the field of computer vision, due to their broad range of applications that include video surveillance and social robotics [310, 120, 193]. Over the past decade, significant advancements have been achieved through the introduction of specialized deep learning architectures, including 3D CNNs [73, 72] and Video Transformers [220], which leverage large-scale annotated datasets for training. Nevertheless, collecting a sufficient amount of labeled video data for training in real-world conditions remains a costly and labor-intensive task. To reduce the need for extensive annotation, video-based Unsupervised Domain Adaptation (UDA) techniques [29, 192, 56] have been proposed. These methods share the underlying principle of transferring knowledge from a labeled source domain to an unlabeled target domain. Despite employing diverse adaptation strategies, all these approaches have demonstrated substantial gains in the robustness of video models, even in the absence of labeled data from the target domain.

In recent years, the field of computer vision has also seen the rise of a new class of highly capable deep architectures trained on vast internet-scale image-text datasets [219]. These models, often referred to as *foundation models*[211, 229, 204, 107] or Vision Language Models (VLMs), have delivered remarkable results and have become central to contemporary vision research. Notably, models such as CLIP [204] and FLAVA [229] have demonstrated that rich image representations can be acquired through supervision based on natural language descriptions. These pre-trained VLMs are now publicly accessible and can be seamlessly incorporated into recognition pipelines. More recently, researchers have begun to explore their potential in the video domain, showing promising results particularly in the context of supervised action recognition [252, 261].

Despite the considerable progress made in video-based UDA, the integration of VLMs into the context of action recognition has remained largely unexplored. We contend that VLMs possess substantial potential to enhance video-based methods, which remains mostly untapped. Our study begins by showing that existing state-of-the-art video-based UDA approaches can be significantly outperformed by publicly available off-the-shelf VLMs, even without any explicit adaptation (see fig. 2.5). This finding underscores the need to rethink traditional paradigms in video-based UDA. The performance gains brought by VLMs are substantial enough to prompt a critical question: *Is*

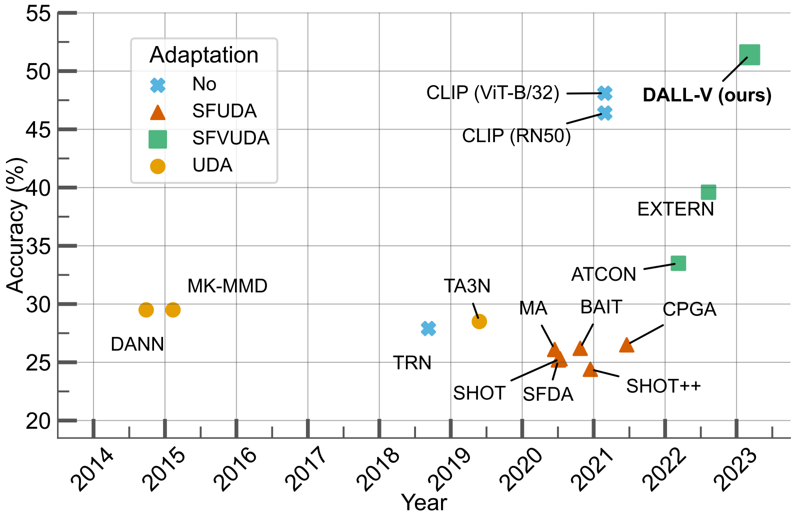
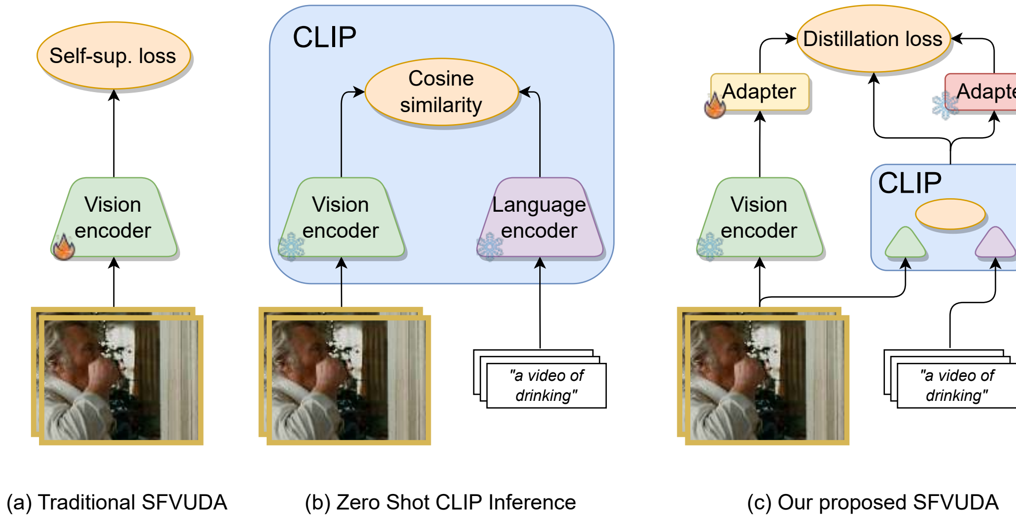


Figure 2.5: Performance over time on the Daily-DA benchmark. CLIP [204] achieves better results than Unsupervised Domain Adaptation (UDA), Source-Free UDA and video-based Source-Free UDA methods. Our method, DALL-V, is built on top of CLIP and outperforms all prior approaches.

explicit alignment, i.e., the conventional strategy in video-based UDA, still the optimal path forward? Instead, we propose a shift in focus to a new research question: *How can we effectively harness the prior knowledge embedded in VLMs to adapt models to target domains in video-based UDA?*

In this chapter, we introduce the first method in the video-based UDA literature that directly tackles this research question. Specifically, we focus on the challenging setting of video-based Source-Free Unsupervised Domain Adaptation (SFUDA) [268, 270], which involves adapting an action recognition model trained on a labeled source domain to an unlabeled target domain, without having access to the original source data. Our main motivation for leveraging VLMs in this context lies in their strong generalization ability, which we hypothesize can alleviate the effects of domain shift between the source and target data distributions. This becomes especially crucial in SFUDA, where the adaptation relies solely on a pre-trained source model, and the absence of source data can severely impact performance, particularly when the domain gap is substantial. Conversely, the broad visual knowledge acquired by VLMs during pretraining on diverse datasets allows them to mitigate this negative impact. Furthermore, VLMs-derived visual representations can serve as a powerful alternative in scenarios where additional modalities (*e.g.*, optical flow), which are often used to bridge the domain gap in video-based UDA frameworks [186, 213], are not available.



(a) Traditional SFVUDA (b) Zero Shot CLIP Inference (c) Our proposed SFVUDA

Figure 2.6: (a) In standard SFUDA setups, a model trained on the source domain is adapted to the unlabeled target dataset using self-supervised signals, such as temporal consistency across frames [268, 270]. (b) VLM approaches (*e.g.*, CLIP [204]) perform zero-shot classification by computing the cosine similarity between visual features and language embeddings. (c) Our proposed SFUDA method fuses predictions from CLIP and the source-trained model, introducing only a minimal number of learnable parameters via adapters.

Our proposed approach, termed **D**omain **A**daptation with **L**arge **L**anguage-**V**ision models (DALL-V), offers a simple yet effective strategy to integrate the visual knowledge from a pre-trained VLMs with the information encoded in the source model and the target data (see fig. 2.6c). DALL-V operates through a two-stage pipeline: (i) initially, pseudo-labels are generated using the CLIP model and utilized for adaptation on the target data, and (ii) an ensemble of CLIP, the source model, and the target model is used to distill their collective knowledge into a student network. This method introduces only a small number of trainable parameters atop CLIP, facilitated by domain-specific adapters. Despite its simplicity, DALL-V achieves a substantial performance gain, surpassing existing SFUDA and even video-based UDA methods by a large margin (+11.8% compared to the strongest baseline).

To summarize, our contributions are:

- We demonstrate, for the first time, that straightforward zero-shot approaches based on VLMs significantly outperform state-of-the-art SFUDA methods.

- Building on this insight, we introduce DALL-V, a simple SFUDA technique that effectively integrates knowledge from VLMs, a pre-trained source model, and unlabeled videos from the target domain.

- We conduct extensive experiments across 20 domain adaptation scenarios, validating the effectiveness of DALL-V in SFUDA settings.

2.2.2 The proposed approach

In this section, we aim to design a video-based Source-Free Unsupervised Domain Adaptation (SFUDA) technique capable of adapting a model trained on a source domain to a different, unlabeled target domain.

Unlike traditional SFUDA strategies that rely on optimizing various self-supervised objectives on the target data [268], we pursue a different, complementary direction. Our core idea is to leverage a Vision Language Model (VLM), such as CLIP [204], pre-trained on large-scale image-text pairs, as a tool to bridge the domain gap.

As a preliminary investigation, we evaluate CLIP in a zero-shot setting (see fig. 2.6b) by directly performing inference on the target dataset. Specifically, we assess its performance on the Daily-DA [268] benchmark using a pre-trained CLIP model, without any fine-tuning on target frames or videos (see section 2.2.2 for zero-shot inference details). We compare these results with those obtained by recent state-of-the-art SFUDA methods across four target domains, each containing eight semantic categories. As illustrated in fig. 2.5, we discover that CLIP (ViT-B/32) surpasses the best-performing SFUDA method, EXTERN [270], by a notable margin of +8.5%, despite not being tailored to the video domain. This compelling result motivates us to further explore the untapped potential of VLMs for SFUDA, offering a novel and orthogonal perspective to the existing literature.

Pursuing this path has several key implications:

- Given that VLMs are publicly accessible, they facilitate more accessible and privacy-conscious adaptation to target domains, even in the absence of source data.
- It eliminates the need for carefully balancing multiple training objectives, *i.e.*, a common requirement in SFUDA, *e.g.*, ATCoN [268] requires the joint optimization of up to six loss functions.
- It avoids reliance on ad hoc architectures or specialized training schemes, simplifying the design of the adaptation framework.

Motivated by the adaptability and expressive power of VLMs, we introduce DALL-V for SFUDA, a method that capitalizes on the *world prior* captured by VLMs and integrates it with information from the source-trained model. Before detailing our proposed approach, we provide a formal definition of the SFUDA task and introduce essential preliminaries.

Problem definition and notations

In SFUDA, we are provided with an unlabeled target dataset $\mathcal{D}^{\mathtt{T}} = \{\mathbf{X}_i^{\mathtt{T}}\}_{i=1}^{m}$, consisting of m video sequences, along with access to a source model $F^{\mathtt{S}}(\cdot)$ that was trained on a labeled source dataset $\mathcal{D}^{\mathtt{S}} = \{\mathbf{X}_i^{\mathtt{S}}, y_i^{\mathtt{S}}\}_{i=1}^{n}$—which is, however, unavailable during the adaptation phase.

We assume that the target dataset $\mathcal{D}^{\mathtt{T}}$ contains action videos whose label space matches that of the source, *i.e.*, $\mathcal{Y}^{\mathtt{S}} = \mathcal{Y}^{\mathtt{T}} = 1, \ldots, C$, where C is the total number of semantic action classes. Additionally, we assume a distributional mismatch between source and target domains, often referred to as domain shift [268].

The objective of SFUDA is to utilize both the target dataset and the source-trained model to learn a predictive function—typically instantiated via a neural network—that can accurately classify samples from the target domain, all while having no access to the original source data.

Preliminaries

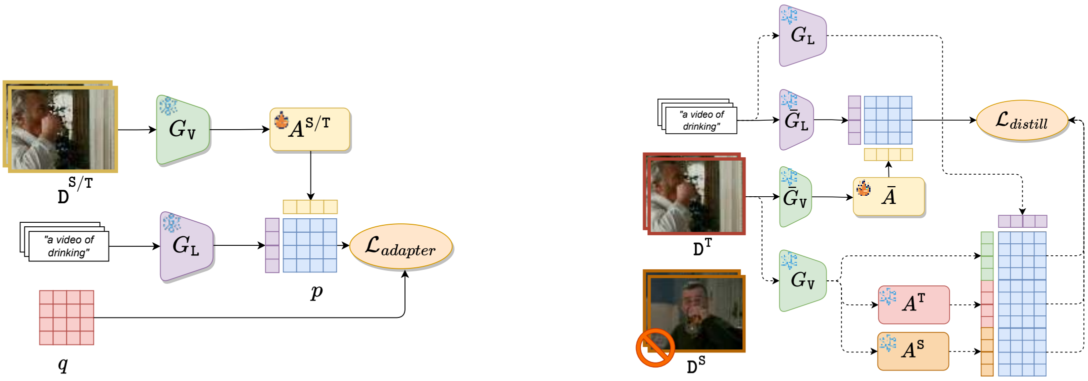
CLIP [204], short for Contrastive Language-Image Pre-training, is composed of two encoders: a vision encoder $G_{\mathtt{V}}(\cdot)$ and a language encoder $G_{\mathtt{L}}(\cdot)$ (see fig. 2.6b). To perform classification, CLIP aligns visual inputs with corresponding textual descriptions by computing their similarity. Given the class labels in a dataset (*i.e.*, *drinking*, *walking*, etc.), we construct prompts for each class, such as “`a video of a person` [CLS]", where `[CLS]` is replaced with the class name.

The language encoder $G_{\mathtt{L}}(\cdot)$ maps these prompts to a set of text embeddings, denoted by $\{\boldsymbol{z}_c^{\mathtt{L}}\}_{c=1}^{C}$. Simultaneously, given a test video with K frames $\mathbf{X} = \{\mathbf{x}_k\}_{k=1}^{K}$, the vision encoder $G\mathtt{V}(\cdot)$ generates visual features for each frame, resulting in $\mathbf{Z}^{\mathtt{V}} = \{\boldsymbol{z}_k^{\mathtt{V}}\}_{k=1}^{K}$.

The probability of the k-th frame belonging to the c-th class is computed as:

$$p_c(\mathbf{x}_k) = \frac{\exp(< \boldsymbol{z}_c^{\mathtt{L}}, \boldsymbol{z}_k^{\mathtt{V}} > /\tau)}{\sum_{c'=1}^{C} \exp(< \boldsymbol{z}_{c'}^{\mathtt{L}}, \boldsymbol{z}_k^{\mathtt{V}} > /\tau)} \tag{2.5}$$

where τ is a temperature parameter, and $< \cdot, \cdot >$ denotes the inner product (or cosine similarity). Frame-level predictions are aggregated by averaging the class probabilities across all frames.



(a) Source/target adapter ($A^{\mathtt{S}}/A^{\mathtt{T}}$) training. The q for the target domain is constructed with CLIP (ViT-B/32) via pseudo-labeling.

(b) Ensemble distillation. We use CLIP (ViT-B/32), $A^{\mathtt{S}}$, and $A^{\mathtt{T}}$ as teachers and train a student adapter $\overline{A}$ on a smaller CLIP model, *i.e.*, RN50.

Figure 2.7: Overview of the DALL-V pipeline. $\overline{bar}$ denotes the student components, and is used to distinguish them from the teachers.

Inference using CLIP is considered zero-shot when the dataset has not been seen during its training. That is, zero-shot inference refers to the ability to generalize to previously unseen datasets [204], which should not be confused with generalization to entirely novel object categories [130].

Domain Adaptation with Large Language-Vision models (DALL-V)

We introduce Domain Adaptation with Large Language-Vision models (DALL-V) as a novel approach to SFUDA. As the name implies, DALL-V leverages the extensive world knowledge captured by VLMs, *i.e.*, CLIP, to reduce the domain gap. While CLIP is trained on image-text pairs rather than video-caption data, the source model complements it by being trained supervisedly on labeled video samples. Our goal is to jointly harness the strengths of these two models in a unified and effective manner. DALL-V consists of two primary stages:

- **Target adaptation**: In this stage, we generate pseudo-labels for the unlabeled target videos using CLIP, and then use these to train a model tailored to the target domain. This enables the new model to benefit from both CLIP's broad knowledge and the unique patterns of the target data.
- **Ensemble distillation**: In the second stage, we ensemble the source and target models and distill their combined knowledge into a compact student network via knowledge distillation [94], which is used at test time.

Before detailing each component of DALL-V, we first describe the training procedure used for the source model.

Source Pre-Training. Unlike prior work such as ATCoN, which fine-tunes all parameters of the source model on the source data, we adopt a more restrained strategy. To avoid overfitting to the source domain, we initialize the source model with CLIP pre-trained weights and fine-tune only a small subset of parameters.

Specifically, we introduce an adapter module [79], *i.e.*, a lightweight multi-layer perceptron—that is appended on top of CLIP's vision encoder. The source training phase consists of updating only the adapter's parameters using a supervised objective, while keeping the vision encoder of CLIP frozen.

Formally, we extend the CLIP vision encoder $G_{\mathtt{V}}(\cdot)$ with an adapter $A(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$, where d is the dimensionality of the output features. We construct textual prompts t for each video using a set of 16 handcrafted templates [252], such as `a video of` [CLS]" or `can you recognize the action of` [CLS]?", where y denotes the class label.

To train the source model, we adopt the objective proposed by ActionCLIP [252]. We first compute ground-truth similarity scores $q(\mathbf{x})$ for each frame $\mathbf{x}$, assigning a value of 1 to positive frame-prompt pairs and 0 to negatives within each batch. The adapter A^{S} is then trained using the Kullback-Leibler (KL) divergence loss between the predicted probabilities $\mathbf{p}(\mathbf{X})$ (as defined in eq. (2.5)) and the target similarity distribution $\mathbf{q}(\mathbf{X})$:

$$\mathcal{L}^{\text{S}}_{\text{adapter}} = \mathbb{E}_{(\mathbf{X},t)\sim\mathcal{D}^{\text{S}}}[\text{KL}(\mathbf{p}(\mathbf{X}), \mathbf{q}(\mathbf{X}))] \tag{2.6}$$

Target adaptation While many source-free methods use the source model to generate pseudo-labels for the unlabeled target data [153], we choose a different route. To adapt CLIP to the target domain, we employ a strategy similar to the source training pipeline, with a key distinction: the absence of labeled data in the target domain. Instead of relying on the potentially biased source model—which includes the source adapter A^{S}—we use the original, unmodified CLIP model to perform pseudo-labeling.

Although the source model only fine-tunes a small number of parameters via the adapter, it can still retain a bias toward the source distribution, possibly resulting in noisy pseudo-labels. To avoid this, we rely on zero-shot inference from CLIP (as discussed in section 2.2.2), using the same set of handcrafted textual prompts employed during source training, to assign pseudo-labels to samples in the target domain.

To ensure label quality, we apply a confidence threshold to filter out uncertain predictions. This results in a subset $\tilde{\mathcal{D}}^{\mathtt{T}} = \mathbf{X}^{\mathtt{T}} ii = 1^{m'} \subset \mathcal{D}^{\mathtt{T}}$, where each sample has an associated pseudo-label $\tilde{y}ii = 1^{m'}$, and m' denotes the number of retained samples.

Following the same setup as in source training, we append an adapter $A^{\mathtt{T}}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ to the CLIP vision encoder and optimize the adapter's weights using the pseudo-labeled subset, while keeping the CLIP encoder itself frozen.

$$\mathcal{L}^{\mathtt{T}}_{\text{adapter}} = \mathbb{E}_{(\mathbf{X},t)\sim\tilde{\mathcal{D}}^{\mathtt{T}}}[\text{KL}(\mathbf{p}(\mathbf{X}), \mathbf{q}(\mathbf{X}))] \tag{2.7}$$

Ensemble distillation With access to three different models, *i.e.*, the source model, the target model, and the original CLIP, we perform knowledge distillation to transfer their combined insights into a single compact network. Knowledge Distillation [94] offers two main benefits: (i) it facilitates the generation of more robust and informative pseudo-labels, and (ii) distilling into a smaller student model reduces inference latency with minimal loss in accuracy.

To achieve this, we designate CLIP ViT-B/32 (also used as the backbone for both source and target adapters) as the teacher ensemble, and CLIP RN50 as the student model. Since we integrate knowledge from three different sources, we refer to this stage as ensemble distillation.

To unify the outputs from the three models, we consider two aggregation strategies. A straightforward method is to average their probability distributions:

$$\mathbf{p}^{\text{ens}}(\mathbf{X}) = \frac{1}{3}(\underbrace{\mathbf{p}(\mathbf{X})}_{\text{ZS CLIP}} + \underbrace{\mathbf{p}^{\text{S}}(\mathbf{X})}_{\text{Source adapter}} + \underbrace{\mathbf{p}^{\text{T}}(\mathbf{X})}_{\text{Target adapter}}) \tag{2.8}$$

Alternatively, a majority voting mechanism can be used, where each model casts a vote for a class, and the label with at least two votes is selected as the pseudo-label.

Following [94], we incorporate both the ensembled soft labels and the majority-voted hard labels in our training objective. We train the student network using a standard cross-entropy loss $\mathcal{L}_{\text{ce}}$ for classification, and a soft distillation loss that minimizes the divergence between the student's predictions and the ensembled teacher outputs:

$$\mathcal{L}_{\text{discrepancy}} = \mathbb{E}_{(\mathbf{X},t)\sim\mathcal{D}^{\text{T}}}[\text{KL}(\frac{\mathbf{p}^{\text{student}}(\mathbf{X})}{\tau}, \frac{\mathbf{p}^{\text{ens}}(\mathbf{X})}{\tau})] \tag{2.9}$$

where $\mathbf{p}^{\text{student}}$ is the predicted distribution of the student network and τ is the temperature.

The final distillation loss combines the two components as:

$$\mathcal{L}_{distill} = \alpha * \mathcal{L}_{ce} + (1-\alpha) * \mathcal{L}_d \tag{2.10}$$

where α balances the contribution of each loss term and is set following the recommendations in [94].

Importantly, we do not fine-tune the entire student model. Instead, we freeze the CLIP RN50 backbone and train only an adapter $\overline{A}$ appended on top, using the loss defined in eq. (2.10).

2.2.3 Experimental results

Datasets and settings. We conduct a comprehensive experimental evaluation using three standard benchmarks tailored for SFUDA. First, we assess our method on Daily-DA[268], which includes 18,949 video samples across 8 action classes, aggregated from four diverse datasets: HMDB51 [124], ARID [267], MIT [184], and Kinetics [113]. Next, we evaluate on UCF-HMDB [29], consisting of 3,209 videos categorized into 12 actions, drawn from HMDB51 [124] and UCF101 [233]. Among the two, Daily-DA poses greater challenges due to significant domain variations, particularly in lighting conditions. Finally, we test on Sports-DA [269], which comprises 40,718 videos across 23 categories from UCF101 [233], Sports-1M [112], and Kinetics [113].

Implementation details. Our adapter modules are implemented as two-layer perceptrons, as suggested by [79]. We use the AdamW optimizer [171] with a learning rate of 0.01 and weight decay of 0.2, consistent with [252]. The temperature parameter τ is fixed at 2.0 across all benchmarks, and each model is trained for 30 epochs. All experiments are conducted using either 4 Tesla V100 GPUs or 2 RTX A6000 GPUs.

For both source pre-training and target adaptation phases, we adopt CLIP pre-trained ViT-B/32 [204] as the vision backbone. In the ensemble distillation phase (section 2.2.2), we employ the CLIP ResNet-50 variant as the student model to ensure fair comparison with prior SFUDA approaches. Throughout all training stages, the CLIP vision encoders remain frozen to preserve their representational capabilities.

In line with recent efforts in parameter-efficient fine-tuning [96, 79], we append lightweight trainable adapters $A(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ on top of the vision encoder in every phase of our DALL-V framework, where d is the input dimensionality. Each adapter consists of a down-projection layer, followed by a ReLU activation, then an up-projection layer, and a final ReLU. The hidden dimension of the adapter is set to one fourth of the input size.

Unlike [79], we do not apply adapters to the language encoder; the text embeddings are used as-is to compute prediction probabilities, following eq. (2.5). Similar to the vision module, the language encoder remains frozen.

Baselines and competitors. We compare our method against a comprehensive suite of baselines in both video-based UDA and SFUDA categories, including TRN [303], DANN [78], MK-MDD [168], TA^{3}N [29], SFDA [115], SHOT [153], SHOT++[155], MA[146], BAIT [273], and CPGA [203]. Additionally, we report results from the most recent state-of-the-art methods: ATCoN [268] and EXTERN [270].

To further contextualize our performance, we introduce CLIP-based baselines that perform zero-shot inference using CLIP with different backbones: CLIP (RN50) and CLIP (ViT-B/32). We also include the lower-bound performance of the source-only model and the upper-bound of a target-supervised model for reference.

Comparison with the State-of-the-Art

We present our evaluation results in table 2.5, table 2.6, and table 2.7, corresponding to the Daily-DA, UCF-HMDB, and Sports-DA benchmarks, respectively. For each benchmark, the lower bound refers to the performance of the source model, *i.e.*, CLIP with the source adapter A^{S}, evaluated directly on the target data. The upper bound, instead, represents an oracle model trained with full supervision on the target data.

On the more challenging Daily-DA benchmark, results in table 2.5 show that our distillation-based approach, DALL-V (RN50), achieves the best average accuracy across 12 domain shifts compared to other methods using the same backbone. Notably, DALL-V outperforms the second-best method (CLIP RN50 zero-shot) by 5% and records the top performance in 9 out of 12 adaptation settings. Among these, performance on the ARID domain is consistently lower—likely due to its difficult visual conditions, including low-light environments.

On the smaller UCF-HMDB benchmark, results show overall higher accuracy levels, indicating a simpler transfer task. Nevertheless, DALL-V achieves the best average performance, outperforming the closest competitor by +0.6% and leading in the HMDB→UCF direction by +1.2%.

For the Sports-DA dataset, as shown in table 2.7, our method performs on par with EXTERN [270], surpassing it in three out of six settings and equaling it in one. In the two scenarios with UCF101 as the target (K→U and S→U), our performance lags by approximately 6 points, resulting in a slightly lower average score (*i.e.*, -0.9%). However, DALL-V still delivers a substantial improvement over ATCoN, with an average gain of +8.5%.

Interestingly, DALL-V's source-only performance already outperforms ATCoN by +2.7% on average. In addition, the zero-shot CLIP baseline proves remarkably strong

	Method	**Accuracy (%)** K→A	K→H	K→M	M→A	M→H	M→K
	TRN [303]	20.9	36.7	29.0	22.1	43.7	53.1
	Lower bound	15.6	47.9	35.7	34.7	44.6	61.6
ZS	CLIP (RN50) [204]	30.5	50.0	42.2	30.5	50.0	62.9
	CLIP (ViT-B/32) [204]	**31.3**	49.6	46.0	**31.3**	49.6	65.4
UDA	DANN [78]	21.2	37.5	21.7	22.8	43.3	58.8
	MK-MDD [168]	21.7	36.2	24.0	21.0	50.4	58.5
	TA^{3}N [29]	19.9	37.7	31.5	21.6	43.0	55.5
SFUDA	SFDA [115]	12.6	44.9	27.5	16.0	35.2	49.2
	SHOT [153]	12.0	44.6	29.5	15.3	36.7	51.0
	SHOT++ [155]	12.6	40.8	28.7	14.9	41.7	46.3
	MA [146]	12.8	45.8	30.0	17.7	37.4	53.5
	BAIT [273]	12.7	45.7	30.0	16.9	39.6	53.0
	CPGA [203]	13.1	46.0	30.7	18.1	39.2	55.1
SFVUDA	ATCoN [268]	17.2	48.2	32.5	27.2	47.3	57.7
	EXTERN [270]	23.9	**55.8**	35.2	18.1	53.7	68.1
	DALL-V (ours)	24.0	52.5	**47.0**	24.0	**65.4**	**78.1**
	Upper bound	26.9	70.4	61.5	26.9	70.4	88.9

	Method	**Accuracy (%)** H→A	H→M	H→K	A→H	A→M	A→K
	TRN [303]	13.8	22.0	37.1	17.2	14.7	24.4
	Lower bound	17.5	25.5	45.1	14.6	15.5	17.8
ZS	CLIP (RN50) [204]	30.5	42.2	62.9	50.0	42.2	62.9
	CLIP (ViT-B/32) [204]	**31.3**	46.0	65.4	49.6	**46.0**	65.4
UDA	DANN [78]	14.2	29.5	38.2	20.1	19.7	27.0
	MK-MDD [168]	20.3	25.7	33.8	18.7	18.0	26.1
	TA^{3}N [29]	14.4	25.7	38.4	14.9	15.6	23.4
SFUDA	SFDA [115]	13.1	24.2	24.9	16.3	13.2	25.2
	SHOT [153]	13.6	24.2	21.2	17.1	14.0	24.3
	SHOT++ [155]	16.0	22.2	33.1	15.4	12.5	21.8
	MA [146]	12.9	25.0	22.2	16.7	15.2	24.3
	BAIT [273]	13.6	25.5	21.2	15.7	14.5	25.5
	CPGA [203]	13.1	26.2	25.5	19.2	16.5	26.7
SFVUDA	ATCoN [268]	17.9	30.7	48.5	26.7	17.2	31.0
	EXTERN [270]	26.2	40.7	57.6	26.2	18.2	51.4
	DALL-V (ours)	24.0	**47.0**	**76.7**	**57.9**	45.7	**75.0**
	Upper bound	26.9	61.5	88.9	70.4	61.5	88.9

Table 2.5: Results on Daily-DA. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50).

	Method	**Accuracy (%)**		
		H→U	U→H	**Avg.**
	TRN [303]	72.8	72.1	72.4
	Lower bound	71.6	76.1	73.8
ZS	CLIP (RN50) [204]	81.0	86.0	83.5
	CLIP (ViT-B/32) [204]	90.3	**89.1**	89.7
UDA	DANN [78]	74.4	75.1	74.8
	MK-MMD [168]	74.7	79.7	77.2
	TA^{3}N [29]	78.1	84.8	81.5
SFUDA	SFDA [115]	69.8	75.0	72.4
	SHOT [153]	74.4	74.4	74.4
	SHOT++ [155]	71.1	68.1	69.6
	MA [146]	74.4	67.3	70.9
	BAIT [273]	75.3	76.3	75.8
	CPGA [203]	75.8	68.1	72.0
SFVUDA	ATCoN [268]	85.3	79.7	82.5
	EXTERN [270]	91.9	88.9	90.4
	DALL-V	**93.1**	88.9	**91.0**
	Upper bound	93.7	91.4	92.6

Table 2.6: Results on UCF-HMDB$_{full}$. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50).

	Method	**Accuracy (%)**						
		K→U	K→S	S→U	S→K	U→K	U→S	**Avg.**
	TRN [303]	86.4	66.9	85.3	71.0	49.3	43.3	67.0
	Lower bound	85.4	79.5	84.4	78.2	67.2	64.3	76.5
ZS	CLIP (RN50) [204]	83.4	79.9	83.4	80.4	80.4	79.9	81.2
	CLIP (ViT-B/32) [204]	90.0	**82.4**	90.0	**85.1**	**85.1**	**82.4**	**85.8**
UDA	DANN [78]	88.0	75.0	85.7	73.4	65.9	55.1	73.8
	MK-MMD [168]	90.2	67.9	90.9	73.6	66.1	55.6	74.0
	TA^{3}N [29]	90.3	68.6	93.0	72.6	63.6	54.1	73.7
SFUDA	SFDA [115]	86.1	60.0	85.4	68.0	55.8	43.6	66.5
	SHOT [153]	91.2	64.9	88.8	72.0	53.9	43.6	69.1
	SHOT++ [155]	90.0	63.1	88.0	70.3	44.7	40.9	66.2
	MA [146]	91.0	65.9	87.8	71.9	60.7	39.4	69.5
	BAIT [273]	92.3	66.6	88.3	72.8	57.2	44.7	70.3
	CPGA [203]	89.4	66.3	86.5	72.5	55.2	44.5	69.1
SFVUDA	ATCoN [268]	93.6	69.7	90.6	76.0	65.2	47.9	73.8
	EXTERN [270]	**93.7**	73.8	**95.4**	82.2	81.2	72.7	83.2
	DALL-V	88.0	77.7	88.8	82.3	81.2	75.9	82.3
	Upper bound	93.4	88.3	93.4	85.6	85.6	88.3	89.1

Table 2.7: Results on Sports-DA. **Bold** indicates best, underline represents best with same backbone as baseline (*i.e.*ResNet50). Lower bound indicates a source adapter and Upper bound a target adapter, both trained supervised on CLIP (RN50).

Method	**Accuracy (%) (Daily-DA)**				
	K→Any	M→Any	H→Any	A→Any	**Avg.**
CLIP	42.3	48.7	47.6	53.7	48.1
A^S	40.0 [-2.3]	55.7 [+7.0]	35.3 [-12.4]	35.0 [-18.7]	41.5 [-6.6]
A^T	44.4 [+2.1]	51.6 [+2.9]	48.5 [+0.9]	58.4 [+4.7]	50.8 [+2.7]
CLIP + A^S	45.1 [+2.8]	**57.8** [+9.1]	40.0 [-7.6]	50.8 [-2.9]	48.4 [+0.3]
CLIP + A^T	43.8 [+1.5]	50.9 [+2.2]	48.2 [+0.6]	57.4 [+3.7]	51.7 [+3.6]
CLIP + A^S + A^T	**46.2** [+3.9]	57.3 [+8.6]	**49.5** [+1.9]	**62.1** [+8.4]	**53.8** [+5.7]

Method	**Accuracy (%) (UCF-HMDB$_{full}$)**		
	H→U	U→H	**Avg.**
CLIP	90.3	89.1	89.7
A^S	91.0 [+0.7]	80.5 [-8.6]	85.8 [-3.9]
A^T	90.5 [+0.2]	91.1 [+2.0]	90.8 [+1.1]
CLIP + A^S	93.1 [+2.8]	87.2 [-1.9]	90.2 [+0.5]
CLIP + A^T	89.3 [-1.0]	**91.9** [+2.8]	90.6 [+0.9]
CLIP + A^S + A^T	**94.9** [+4.6]	90.3 [+1.2]	**92.6** [+2.9]

Table 2.8: Ablation on the relative improvement of the different modules of our proposed DALL-V framework *w.r.t.* CLIP (ViT-B/32). **Bold** indicates best. The + operator indicates ensembling. Results are aggregated by source dataset.

on this dataset, performing nearly on par with EXTERN (only -2.0% with RN50 and +2.6% with ViT-B/32).

Ablation analysis

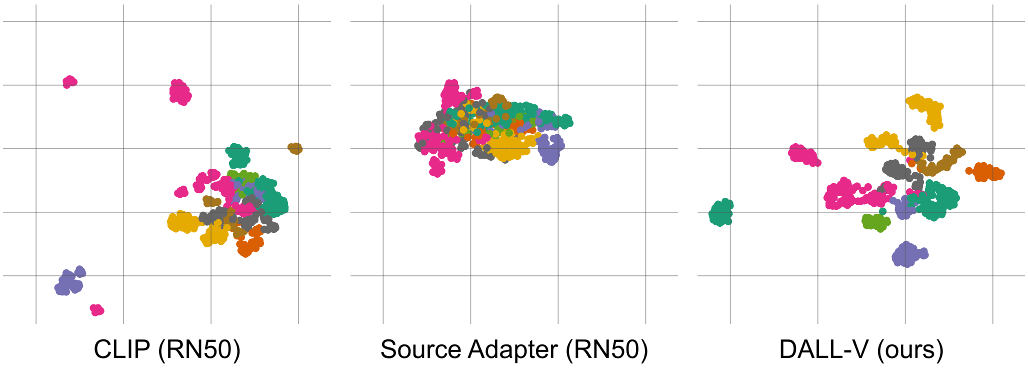
We report in table 2.8 the intermediate performance obtained at various stages of our proposed source-free pipeline, evaluated on both the Daily-DA and UCF-HMDB benchmarks. Specifically, we show results for the supervised source model, the unsupervised target model, the final distilled model, and the zero-shot CLIP model for comparison. We also report the performance of an ensemble combining the predictions of the three models. In this context, $A^{\mathtt{S}}$ and $A^{\mathtt{T}}$ refer to the source and target adapters appended to the frozen CLIP encoder and trained on their respective domains, as detailed in section 2.2.2.

Effectiveness of fine-tuning VLM on the target. As shown in table 2.8, fine-tuning the model on the target domain, even in an unsupervised manner, leads to a notable improvement in validation accuracy. On Daily-DA, the gain is +2.0%, while on UCF-HMDB the improvement is +1.1%. Moreover, when the predictions of the target model are combined with those from CLIP, the accuracy on Daily-DA increases by an additional +3.6%, confirming the benefit of unsupervised adaptation for SFUDA.

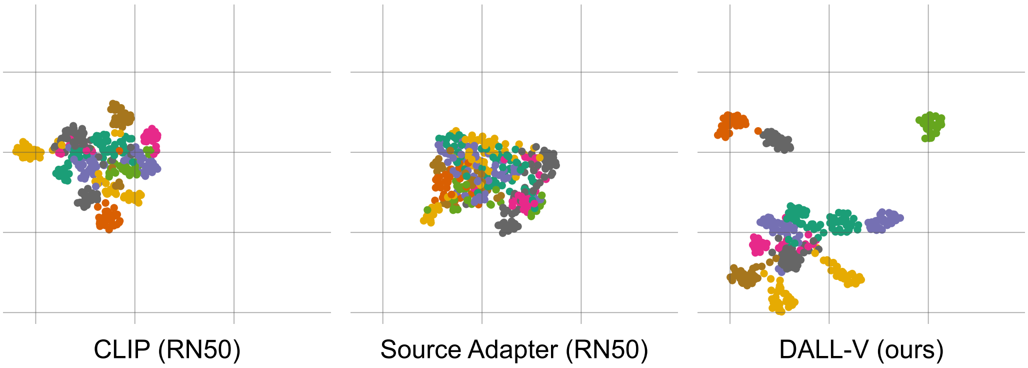
Effectiveness of ensembles combining VLM and domain-specific models. The final row of table 2.8 highlights that ensembling predictions from CLIP, the source model, and the target model yields the best average accuracy on both benchmarks. Specifically, the ensemble delivers a gain of +5.7% on Daily-DA and +2.9% on UCF-HMDB relative to the CLIP baseline. These findings indicate that the three models offer complementary strengths: CLIP brings generalization to unseen domains, while the domain-specific adapters capture nuances of the respective source and target distributions. Although the source and target models individually do not outperform CLIP, they still contribute meaningful discriminative information when combined.

Finally, we distill the ensemble knowledge into a ResNet-based model to ensure direct comparability with prior works that use similar architectures.

Effectiveness of using multiple templates. To better understand the contribution of individual components of our framework, we analyze the impact of the number of textual templates used during inference. Figure 2.9 illustrates how increasing the number of templates—from a single prompt to 12, as proposed in ActionCLIP [252]—significantly improves the model's accuracy on the target validation sets for both considered benchmarks. Beyond 12 templates, however, the performance begins to plateau. We therefore



(a) HMDB→UCF



(b) UCF→HMDB

Figure 2.8: UMAPs of the feature space on ***UCF-HMDB***$_{full}$ for CLIP (RN50), the source model, and the final DALL-V model. DALL-V efficiently clusters the action classes in the feature space by combining knowledge from the other two models.

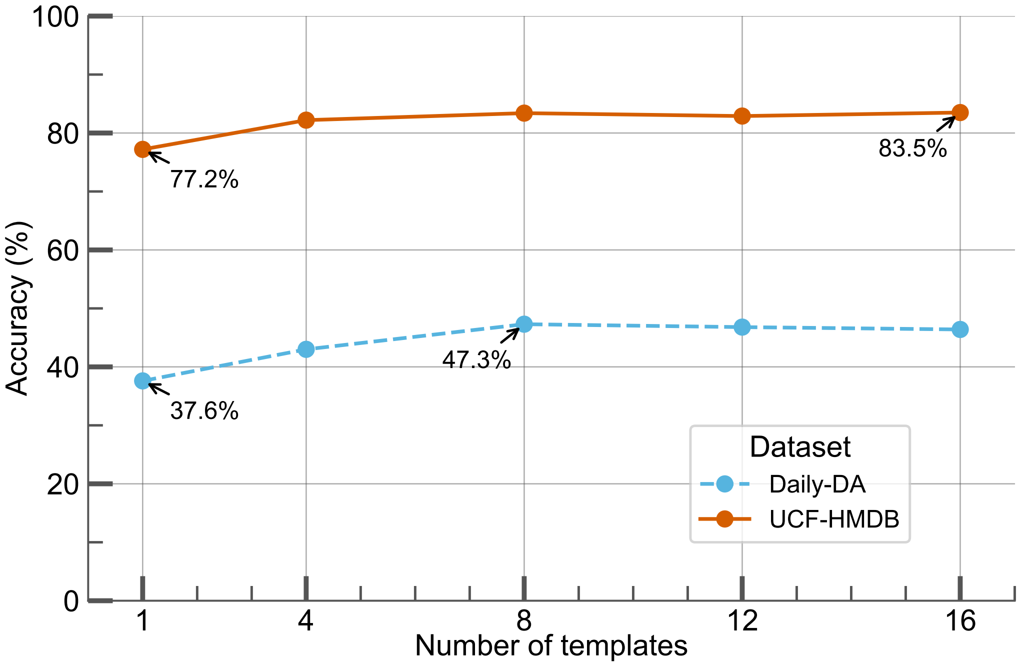


Figure 2.9: Analysis on the number of templates used for the text input *w.r.t.* the accuracy score, on ***Daily-DA*** and ***UCF-HMDB***$_{full}$.

follow the configuration in [252] and adopt 12 templates in our experiments.

Modeling the feature space. In fig. 2.8, we present a UMAP [179] visualization of the feature representations produced by three different models—zero-shot CLIP (RN50), the source model, and our DALL-V model—on the UCF-HMDB benchmark. Each color corresponds to a different action class. The visualization reveals that DALL-V generates more distinct and well-separated clusters compared to the individual models, highlighting its ability to effectively integrate complementary information and improve feature discriminability across action categories.

2.2.4 Related work

Video-based Unsupervised Domain Adaptation. While Unsupervised Domain Adaptation (UDA) has been extensively studied for image-level tasks [83, 271, 167, 237, 29, 266], video-based UDA introduces additional complexity due to temporal dynamics. Several strategies have been proposed to address this challenge. For example, TA^{3}N [29] introduces a discrepancy-based approach to align source and target domains by learning temporal relationships within video sequences. TCoN [192] employs a cross-domain module that aligns temporally structured features. CO2A [56] uses contrastive learning to bridge the domain gap between source and target video representations. Despite these advances, a key limitation of UDA and video-based UDA methods is their reliance on access to source data—an assumption that is often impractical due to privacy or legal constraints. In contrast, our method operates in the more realistic setting where only the source-trained model is available, without requiring access to source data.

Video-based Source-Free Unsupervised Domain Adaptation. The Source-Free Unsupervised Domain Adaptation (SFUDA) paradigm has recently emerged as a response to growing concerns around data sharing and privacy. One of the early contributions in this space is SHOT [153], which leverages an entropy minimization objective and pseudo-labels to adapt a source-pretrained model using target data only. An extension of this method [155] introduces an auxiliary head to solve a relative rotation task, improving performance. Other approaches such as 3C-GAN [145], SFIT [95], and SDDA [128] reframe the SFUDA task as an image translation problem.

Within the specific context of video-based SFUDA, only a few contributions exist. ATCoN [268] proposes a consistency-based adaptation technique leveraging temporal coherence. EXTERN [270] introduces mixup-style augmentations and video-specific regularization to adapt models in a source-free setting. Our work departs from these strategies by integrating Vision Language Models (VLMs) as a core component for domain adaptation.

Vision Language Models. Recent access to large-scale image-text datasets [219, 217, 229] has driven the development of multi-modal models capable of learning joint visual and textual embeddings [204, 107, 229]. These Vision Language Models typically use separate encoders for visual and textual modalities, aligning their embeddings through a contrastive loss [204, 107]. CLIP [204] is a leading example, known for its strong zero-shot recognition capabilities across a variety of datasets. More recently, VLMs have been extended to the video domain, showing promising results in action recognition tasks [252]. However, to the best of our knowledge, our work is the first to employ VLMs for tackling domain shift in the context of video-based action recognition.

Chapter 3

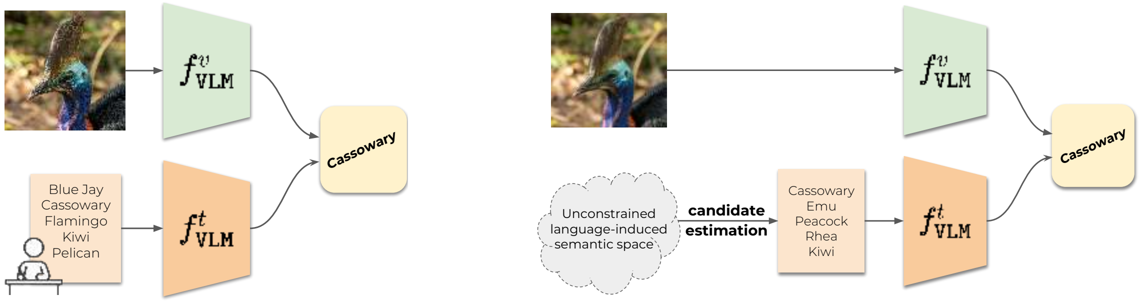
Unsupervised inference

This chapter investigates how the need for predefined supervision can be removed at inference time, a setting where models are typically expected to select outputs from a fixed, human-specified vocabulary. We begin by formalizing the task of Vocabulary-free Image Classification (VIC), additionally introducing a method that uses a Vision Language Model (VLM) to assign meaningful labels to images by querying external databases, without requiring pre-specified categories. We then extend this paradigm to semantic segmentation, showing how dense pixel-wise predictions can be produced under the same open-ended constraints. Finally, we explore unsupervised inference with Large Multimodal Models (LMMs), where classification is cast as an open-ended language generation task. Through these three perspectives, the chapter aims to broaden the understanding of inference as a flexible and adaptive process, no longer constrained by fixed label sets or rigid task formulations.

3.1 Classification with Vision Language Models

3.1.1 The problem

Large-scale Vision Language Models (VLMs) [206, 276, 142] enabled astonishing progress in computer vision by aligning multimodal semantics in a shared embedding space. This chapter focuses on their use for image classification, where models such as CLIP [206] demonstrated strength in zero-shot transfer. While we witnessed advances in VLM-based classification in many directions, *e.g.*, prompt learning [227, 306], scaling up to larger models and datasets [106, 198, 41], or by jointly considering captioning task [276, 142], they all assume a finite set of target categories, *i.e.*, the *vocabulary*, to be pre-defined and static (as shown in fig. 3.1a). However, this assumption is fragile, as



(a) VLM-based classification (b) Vocabulary-free Image Classification

Figure 3.1: Vision-Language Model (VLM)-based classification (a) assumes a pre-defined set of target categories, *i.e.*, the vocabulary, while our novel task (b) lifts this assumption by directly operating on the unconstrained language-induced semantic space, without a known vocabulary. f^v_{VLM} and f^t_{VLM} denote the pre-trained vision and text models of a VLM.

it is often violated in practical applications (*e.g.*, robotics, autonomous driving) where semantic categories can either differ from the development/training to the deployment/testing or evolve dynamically over time.

In this work, we remove this assumption and study the new task of Vocabulary-free Image Classification (VIC). The objective of VIC is to assign an image to a class that belongs to an unconstrained language-induced semantic space at test time, *without a vocabulary*, *i.e.*, without a pre-defined set of categories (as shown in fig. 3.1b). The unconstrained nature of the semantic space makes VIC a challenging problem. First, the search space is extremely large, with a cardinality on the order of millions of semantic concepts[1], way larger than any existing image classification benchmark (*e.g.*, ImageNet-21k [60]). Second, it also includes very fine-grained concepts that might be hard to discriminate by the model. Third, as the categories encountered at test time are undefined beforehand, VIC calls for classification methods that do not rely on any vocabulary-aware supervision.

Recent web-scale Vision Language Databasess (VLDs) [218, 230], offer a unique opportunity to address VIC, as they cover a wide set of semantic concepts, ranging from general to highly specific ones. We empirically show that such external databases allow identifying semantic content that is more relevant to the target category than the captioning-enhanced VLMs supporting visual queries (*e.g.*, BLIP-2 [140]). Motivated by this observation, we propose a training-free method for VIC, Category Search from External Databases (CaSED), which jointly leverages the discriminative multi-

---

[1] We estimate the number of concepts using BabelNet [187], which is close to 4 million for English.

modal representations derived from CLIP [206] and the information provided by recent VLDs (*e.g.*, PMD [230]). CaSED operates in two steps: it first coarsely estimates a set of candidate categories for the test image, and then it predicts the final category via multimodal matching. Specifically, we first retrieve the captions from the database that are semantically closer to the input image, from which we extract candidate categories via text parsing and filtering. We then estimate the similarity score between the input image and each candidate category via CLIP, using both visual and textual information, predicting as output the best matching candidate. CaSED exploits the pre-trained CLIP without further training, thus being flexible and computationally efficient.

We experiment on several datasets, considering both coarse- (*e.g.*, Caltech-101 [70], UCF101 [233]) and fine-grained (*e.g.*, FGVC-Aircraft [176], Flowers-102 [188]) classification tasks. To quantitatively assess the performance of methods addressing VIC, we also propose a set of metrics to measure how well the predicted class matches the semantics of the ground-truth label. Across all tasks and metrics, CaSED consistently outperforms all baselines, including VLMs as complex as BLIP-2 [140]. We believe that thanks to its simplicity and effectiveness, our training-free CaSED can serve as a competitive baseline for future works aiming to address the challenging VIC task.

To summarize, this work provides the following contributions:

- We explore the task of Vocabulary-free Image Classification, where the goal is to assign a class to an image over an unconstrained set of semantic concepts, overcoming the fundamental assumption of existing VLM-based methods for image classification. We formalize this task and suggest specific evaluation metrics that can be used as a reference for future works.

- We propose CaSED, the first method to address VIC thanks to the adoption of large captioning databases. Notably, CaSED is training-free, not requiring any additional parameter nor finetuning of the network's textual and visual encoders.

- Our large-scale evaluation demonstrates that CaSED consistently outperforms a more complex VLM such as BLIP-2 [140] on VIC, while requiring much fewer parameters.

3.1.2 The proposed setting

Preliminaries. Given the image space $\mathcal{X}$ and a set of class labels C, a classification model $f : \mathcal{X} \rightarrow C$ is a function mapping an image $\boldsymbol{x} \in \mathcal{X}$ to its corresponding class $\boldsymbol{c} \in C$. While C may be represented by indices that refer to semantic classes, here we focus on cases where C consists of a set of concept names that correspond to real entities. Note that C is a finite set whose elements belong to the semantic space $\mathcal{S}$, *i.e.*, $C \subset \mathcal{S}$. In standard image classification, C is given a priori, and f is learned on a dataset of image-label pairs.

Recently, the introduction of contrastive-based VLMs [206, 106], which learn a cross-modal aligned feature space, revised this paradigm by defining a function $f_{\texttt{VLM}}$ that infers the similarity between an image and a textual description $\boldsymbol{t} \in \mathcal{T}$, *i.e.*, $f_{\texttt{VLM}} : \mathcal{X} \times \mathcal{T} \rightarrow \mathbb{R}$, with $\mathcal{T}$ being the language space. Given this function, classification can be performed with:

$$f(\boldsymbol{x}) = \arg\max_{\boldsymbol{c} \in C} f_{\texttt{VLM}}(\boldsymbol{x}, \phi(\boldsymbol{c})) \tag{3.1}$$

where $\phi(\boldsymbol{c})$ is a string concatenation operation, combining a fixed text template, *i.e.*, a *prompt*, with a class name. This definition allows for zero-shot transfer, *i.e.*, performing arbitrary classification tasks by re-defining the set C at test time, without re-training the model. However, such zero-shot transfer setup still assumes that the set C is provided. In this work, Vocabulary-free Image Classification (VIC) is formulated to surpass this assumption.

Task definition. VIC aims to assign a class $\boldsymbol{c}$ to an image $\boldsymbol{x}$ *without* prior knowledge on C, thus operating on the semantic class space $\mathcal{S}$ that contains all the possible concepts. Formally, we want to produce a function f mapping an image to a semantic label in $\mathcal{S}$, *i.e.*, $f : \mathcal{X} \rightarrow \mathcal{S}$. Our task definition implies that at test time, the function f has only access to an input image $\boldsymbol{x}$ and a large source of semantic concepts that approximates $\mathcal{S}$. VIC is a challenging classification task by definition due to the extremely large cardinality of the semantic classes in $\mathcal{S}$. As an example, ImageNet-21k [60], one of the largest classification benchmarks, is 200 times smaller than the semantic classes in BabelNet [187]. This large search space poses a prime challenge for distinguishing fine-grained concepts across multiple domains as well as ones that naturally follow a long-tailed distribution.

Semantic space representation. As the main challenge of VIC, how to represent the large semantic space plays a fundamental role in the method design. We can either model the multimodal semantic space directly with a VLM equipped with an autoregressive language decoder [142] or via image-text retrieval from VLDs. Consequently, we can approach VIC either via VQA-enabled VLMs by querying for the candidate class given the input image, or by retrieving and processing data from an external VLD to obtain the candidate class.

To investigate the two potential strategies, we perform a preliminary experimental

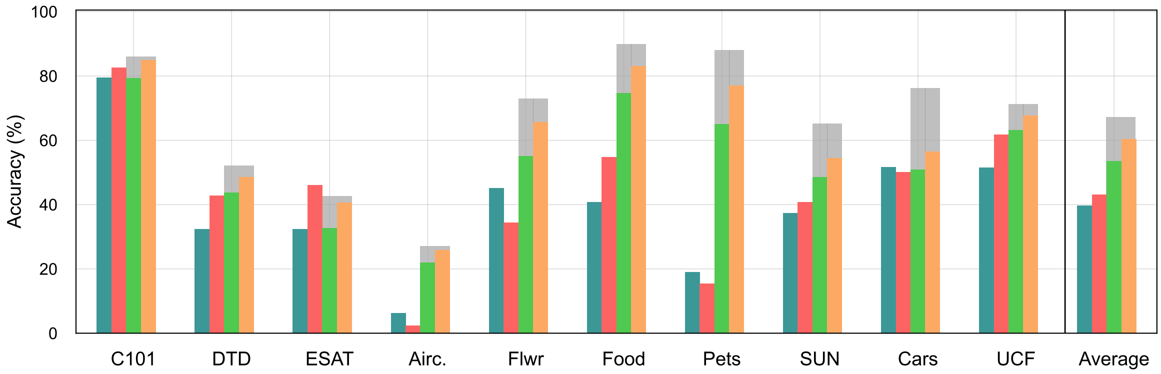


Figure 3.2: Results of our preliminary study, showing the top-1 accuracy when matching semantic descriptions to ground-truth class names in ten different datasets. We compare BLIP-2 (VQA) and BLIP-2 (Captioning) with Closest Caption and Captions Centroid, *i.e.*the average representation of the retrieved captions. We additionally highlight the Upper bound for zero-shot CLIP. Representing the large semantic space as VLDs and retrieving captions from it produces semantically more similar outputs to ground-truth labels *w.r.t.* querying outputs from VQA-enabled VLMs, while requiring 10 times fewer parameters compared to the latter.

analysis to understand how well the output of a method semantically captures the image category, or in other words, to assess the alignment of class boundaries in the visual and textual representations. Specifically, we compare the semantic accuracy of querying VQA VLMs and of retrieving from VLDs *w.r.t.* the ground-truth class labels. We consider the output of a method as correct if its closest textual embedding among the target classes of the dataset corresponds to the ground-truth class of the test sample[2]. We exploit the text encoder of CLIP (ViT-L) [206] to obtain textual embeddings.

Regarding experimented methods, we select BLIP-2 [140] to represent VQA-enabled VLMs for its state-of-the-art performance in VQA benchmarks, while we use a subset of PMD [230] as the VLD. In particular, we compare the following methods: i) *BLIP-2 VQA*, which directly queries BLIP-2 for the image category; ii) *BLIP-2 Captioning*, which queries BLIP-2 for the image caption; iii) *Closest Caption*, which is the closest caption to the image, as retrieved from the database; iv) *Caption Centroid*, which averages the textual embeddings of the 10 most similar captions to the input image. As we use CLIP embeddings, if visual and textual representations perfectly align, the performance would be the same as zero-shot CLIP with given target classes. We thus report zero-shot CLIP to serve as the upper bound for retrieval accuracy.

---

[2] Note that this metric is not the standard accuracy in image classification as we use distances in the embedding space to ground predictions from the unconstrained semantic space to the set of classes in a specific dataset.

We experiment on a variety of test datasets for both coarse- and fine-grained classification (see details in section 3.1.4), and report the results in fig. 3.2. The average textual embedding of the retrieved captions (*i.e.*, Caption Centroid) achieves the best semantic accuracy for 9 datasets out of 10, consistently surpassing methods based on BLIP-2. On average, the accuracy achieved by Caption Centroid is 60.47%, which is +17.36% higher than the one achieved by BLIP-2 Captioning (43.11%). Moreover, Captions Centroid achieves results much closer to the CLIP upper bound (67.17%) than the other approaches. Notably, such VLD-based retrieval is also computationally more efficient, faster (∼4 second for a batch size of 64 on a single A6000 GPU), and requires fewer parameters (approximately 10 times less) than BLIP-2 (see table 3.13).

The results of this preliminary study clearly suggest that representing the large semantic space with VLDs can produce (via retrieval) semantically more relevant content to the input image, in comparison to querying VQA-enabled VLMs, while being computationally efficient. Based on this conclusion, we develop an approach, Category Search from External Databases (CaSED), that searches for the semantic class from the large semantic space represented in the captions of VLDs.

3.1.3 The proposed approach

Our proposed method CaSED finds the best matching category within the unconstrained semantic space by multimodal data from large VLDs. Figure 3.3 provides an overview of our proposed method. We first retrieve the semantically most similar captions from a database, from which we extract a set of candidate categories by applying text parsing and filtering techniques. We further score the candidates using the multimodal aligned representation of the large pre-trained VLM, *i.e.*, CLIP [206], to obtain the best-matching category. We describe in detail each process in the following.

Generating candidate categories

We first restrict the extremely large classification space to a few most probable candidate classes. Let $f_{\texttt{VLM}}$ be the pre-trained VLM and D be the external database of image captions. Given an input $\boldsymbol{x}$, we retrieve the set $D_{\boldsymbol{x}} \subset D$ of K closest captions to the input image via

$$D_{\boldsymbol{x}} = \underset{\boldsymbol{d} \in D}{\text{top-k}}\ f_{\texttt{VLM}}(\boldsymbol{x}, \boldsymbol{d}) = \underset{\boldsymbol{d} \in D}{\text{top-k}}\ \langle f^{v}_{\texttt{VLM}}(\boldsymbol{x}), f^{t}_{\texttt{VLM}}(\boldsymbol{d}) \rangle, \tag{3.2}$$

where $f^v_{\texttt{VLM}} : \mathcal{X} \rightarrow \mathcal{Z}$ is the visual encoder of the VLM, $f^t_{\texttt{VLM}} : \mathcal{T} \rightarrow \mathcal{Z}$ is the textual encoder, and $\mathcal{Z}$ is their shared embedding space. The operation $\langle \cdot, \cdot \rangle$ indicates the computation of the cosine similarity. Note that our approach is agnostic to the particular form of D, and that it can accommodate a flexible database size by including captions from additional resources.

From the set $D_{\boldsymbol{x}}$, we then extract a finite set of candidate classes $C_{\boldsymbol{x}}$ by performing simple text parsing and filtering techniques, *e.g.*, stop-words removal, POS tagging.

Filtering candidate categories

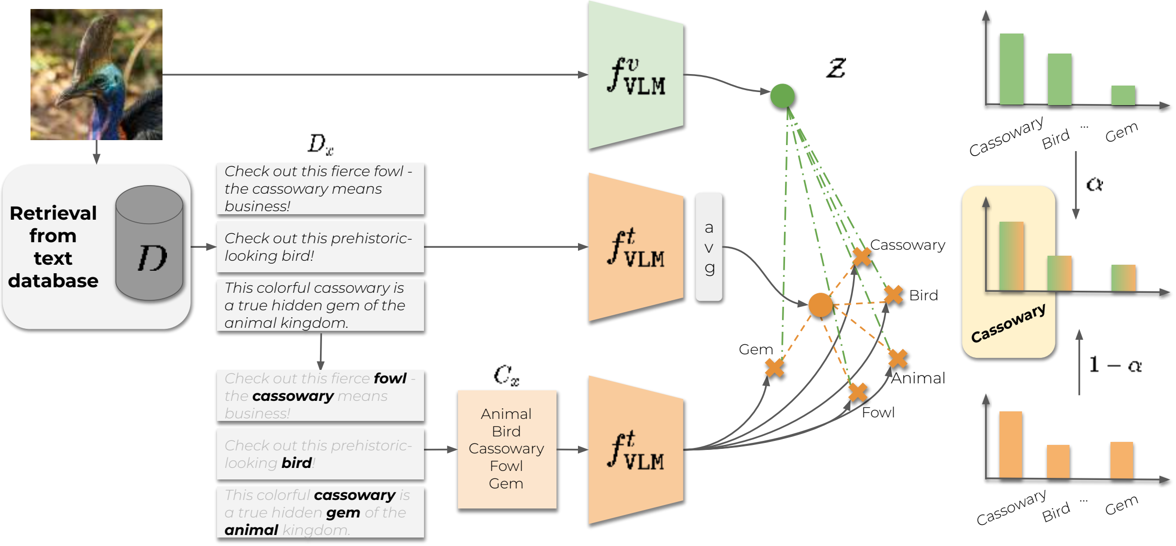
With the closest captions retrieved from the external database given the input image, we post-process them to filter out a set of candidate category names. We first create a set of all words that are contained in the captions. Then we apply sequentially three different groups of operations to the set to (i) remove noisy candidates, (ii) standardize their format, and (iii) filter them.

With the first group of operations, we remove all the irrelevant textual contents, such as tokens (*i.e.*, "<PERSON>"), URLs, or file extensions. Note that, for the file extensions, we remove the extension but retain the file name as it might contain candidate class names. We also remove all the words that are shorter than three characters and split compound words by underscores or dashes. Finally, we remove all those terms containing symbols or numbers and meta words that are irrelevant to the classification task, such as "image", "photo", or "thumbnail". The second group of operations standardizes the candidate names by aligning words that refer to the same semantic class to a standard format, reducing class redundancy. For example, "cassowary" and "Cassowary" will be considered as a single class instead of two. To this end, we perform two operations–lowercase conversion and singular form conversion.

The last group of operations considers two forms of filtering, where the first aims to filter out entire categories of words via Part-Of-Speech (POS) tag, and the second aims to filter out rare and noisy contents based on the word occurrences. We select these two operations since common dataset class names do not contain terms that carry no semantics *e.g.*, articles and pronouns, and since [55] showed that CLIP performs better when exposed to a smaller amount of unique tokens. The POS tagging[3] categorizes words into groups, such as adjectives, articles, nouns, or verbs, enabling us to filter all the terms that are not semantically meaningful for a classification task. Regarding the occurrence filtering, we first count how often a word appears in the retrieved captions

---

[3] We use the NLP library flair (`https://github.com/flairNLP/flair`).



Check out this fierce fowl - the cassowary means business!

Check out this prehistoric-looking bird!

This colorful cassowary is a true hidden gem of the animal kingdom.

Check out this fierce **fowl** - the **cassowary** means business!

Check out this prehistoric-looking **bird**!

This colorful **cassowary** is a true hidden **gem** of the **animal** kingdom.

Figure 3.3: Overview of CaSED. Given an input image, CaSED retrieves the most relevant captions from an external database filtering them to extract candidate categories. We classify image-to-text and text-to-text, using the retrieved captions centroid as the textual counterpart of the input image.

and then we remove words that appear only once to make the candidate list less noisy.

Multimodal candidate scoring

Among the small set $C_{\boldsymbol{x}}$, we score each candidate by accounting for both visual and textual semantic similarities using the VLM encoders, to select the best-matching class for the input image.

Image-to-text score. As the image is the main driver for the semantic class we aim to recognize, we use the visual information to score the candidate categories. We denote $s^{v}_{\boldsymbol{c}}$ as the visual score of each candidate category $\boldsymbol{c}$ and compute it as the similarity between the visual representation of the input image and the textual representation of the candidate name:

$$s^{v}_{\boldsymbol{c}} = \langle f^{v}_{\texttt{VLM}}(\boldsymbol{x}), f^{t}_{\texttt{VLM}}(\boldsymbol{c}) \rangle. \tag{3.3}$$

The higher value of $s^{v}_{\boldsymbol{c}}$ indicates a closer alignment between the target image and the candidate class.

Text-to-text score. While the image-to-text score $s^{v}_{\boldsymbol{c}}$ is effective, there exists a well-known modality gap in the space $\mathcal{Z}$, harming the performance of zero-shot models [156]. As suggested by fig. 3.2 in section 3.1.2, the semantic relevance of the retrieved captions, and their centroid in particular, is high *w.r.t.* the underlying ground-truth label. We are therefore motivated to exploit such attributes and introduce unimodal text-to-text scoring to mitigate the modality gap of cross-modal scoring.

Formally, we define the centroid $\bar{\boldsymbol{d}}_{\boldsymbol{x}}$ of the retrieved captions as:

$$\bar{\boldsymbol{d}}_{\boldsymbol{x}} = \frac{1}{K} \sum_{\boldsymbol{d} \in D_{\boldsymbol{x}}} f^t_{\texttt{VLM}}(\boldsymbol{d}), \tag{3.4}$$

where K is the number of retrieved captions. We then define the text-based matching score $s^t_{\boldsymbol{c}}$ as the similarity between the centroid and the candidate category:

$$s^t_{\boldsymbol{c}} = \langle \bar{\boldsymbol{d}}_{\boldsymbol{x}}, f^t_{\texttt{VLM}}(\boldsymbol{c}) \rangle. \tag{3.5}$$

A higher value of $s^t_{\boldsymbol{c}}$ means a higher alignment between the caption centroid and the candidate embedding. Note that the semantic relevance of the caption centroid is an inherent property of the text encoder of VLMs (*i.e.*, CLIP). Since CLIP is trained with an image-text alignment loss, its text encoder focuses on the visual elements of the caption, discarding parts that are either non-visual or non-relevant to the visual content. This improves the model's robustness to noisy candidates.

Final predicted candidate. To predict the final candidate, we merge the two scores, obtaining the final score $s_{\boldsymbol{c}}$ for each candidate $\boldsymbol{c}$ as:

$$s_{\boldsymbol{c}} = \alpha \; \sigma(s^v_{\boldsymbol{c}}) \; + \; (1 - \alpha) \; \sigma(s^t_{\boldsymbol{c}}) \tag{3.6}$$

where $\sigma(\cdot)$ is the softmax operation on the two scores of each candidate class, and α is a hyperparameter regulating the contribution of the two modalities. Finally we obtain the output category as $f(x) = \arg\max_{\boldsymbol{c} \in C_{\boldsymbol{x}}} s_{\boldsymbol{c}}$. Notably, CaSED respects the VIC task definition, performing classification without known class priors, while being *training-free* with the use of a pre-trained and frozen VLM. This makes the approach flexible and applicable to a variety of architectures and databases.

UpperCaSED

To improve the performance of CaSED, we propose a simple yet effective modification, introducing prompt ensembling [206] after the caption extraction and filtering. Prompt ensembling applies a predefined list of templates to the class names to enhance their contextual information. For instance, the class name "dog" could be expanded to "a photo of a dog". By applying a set of templates rather than a single one and computing their average representation, the resulting features better capture the semantic meaning of the word, leading to consistent performance gains [206].

We apply prompt ensembling to the candidates generated from the retrieved captions. The number of templates used is variable, depending on the specific dataset[4]. The average representation of these ensembled prompts is then used as anchors to compute the image-to-text and text-to-text scores as described above. Formally, for a candidate $\boldsymbol{c}$ and a set of templates T, the ensembled representation $f^t_{\text{ens}}(\boldsymbol{c})$ is computed as:

$$f^t_{\text{ens}}(\boldsymbol{c}) = \frac{1}{|T|} \sum_{t \in T} f^t_{\texttt{VLM}}(t(\boldsymbol{c})), \tag{3.7}$$

where $t(\boldsymbol{c})$ means applying template t to candidate $\boldsymbol{c}$. $f^t_{\text{ens}}(\boldsymbol{c})$ is then used for image-to-text and text-to-text scoring.

3.1.4 Experimental results

We evaluate CaSED in comparison to other VLM-based methods on the novel task Vocabulary-free Image Classification with extensive benchmark datasets covering both coarse-grained and fine-grained classification. We first describe the experimental protocol in terms of the datasets, the proposed evaluation metrics, and the baselines applicable to this task (section 3.1.4). We then discuss the quantitative results regarding the comparison between our method and baselines (section 3.1.4), and showcase qualitative results of predicted categories from multiple datasets in section 3.1.4. Finally, we present a thorough ablation to justify the design choices of CaSED (section 3.1.4). In addition, we provide a cost comparison between the baseline methods and CaSED in section 3.1.4.

Experimental protocol

Datasets. We follow existing works [227, 306] and use ten datasets that feature both coarse-grained and fine-grained classification in different domains: Caltech-101 (C101) [70], DTD [45], EuroSAT (ESAT) [92], FGVC-Aircraft (Airc.) [176], Flowers-102 (Flwr) [188], Food-101 (Food) [18], Oxford Pets (Pets), Stanford Cars (Cars) [121], SUN397 (SUN) [258], and UCF101 (UCF) [233]. Additionally, we used ImageNet [60] for hyperparameters tuning.

Evaluation metrics. Due to the unconstrained nature of the semantic space, evaluating the effectiveness of methods for VIC is not trivial. We propose to use two main criteria, namely *semantic relevance*, *i.e.*, the similarity of the predicted class *w.r.t.* the

---

[4] For each classification dataset, we use the templates defined in CLIP [206], spanning from 1 for, *e.g.*, Flowers-102 [188], Food-101 [18], and Oxford Pets, to 48 for UCF101 [233]

ground-truth label, and *image grouping*, *i.e.*, the quality of the predicted classes for organizing images into clusters. For semantic, we consider i) *Semantic Similarity*, *i.e.*, the similarity of predicted/ground-truth labels in a semantic space, and ii) *Semantic IoU*, *i.e.*, the overlap of words between the prediction and the true label. More formally, given an input $\boldsymbol{x}$ with ground-truth label $\boldsymbol{y}$ and prediction $\hat{\boldsymbol{c}} = f(\boldsymbol{x})$, we compute the *Semantic Similarity* as $\langle g(\hat{\boldsymbol{c}}), g(\boldsymbol{y})\rangle$, where $g : \mathcal{T} \rightarrow \mathcal{Y}$ is a function mapping text to an embedding space $\mathcal{Y}$. Since we want to model free-form text, we use Sentence-BERT [209] as g. For *Semantic IoU*, given a predicted label $\boldsymbol{c}$, and assuming $\boldsymbol{c}$ being a set of words, we compute the Semantic IoU as $|\boldsymbol{c} \cap \boldsymbol{y}|/|\boldsymbol{c} \cup \boldsymbol{y}|$, where $\boldsymbol{y}$ is the set of words in the ground-truth label. To assess grouping, we measure the classic *Cluster Accuracy* by first clustering images according to their predicted label, and then assigning each cluster to a ground-truth label with Hungarian matching. Sometimes, this mapping is resolved with a many-to-one match, where a predicted cluster is assigned to the most present ground-truth label. This evaluation draws inspiration from the protocols used for deep visual clustering [248, 104, 89].

Baselines. We consider three main groups of baselines for our comparisons. The most straightforward baselines consist of using CLIP with large vocabularies, such as WordNet [180] (117k names) or the English Words (234k names [76]). As an upper bound, we also consider CLIP with the perfect vocabulary, *i.e.*, the ground-truth names of the target dataset (CLIP upper bound). The second group of baselines consists of captioning methods, as captions can well describe the semantic content of images. We consider two options: captions retrieved from a database and captions generated by a pre-trained image captioning model. For the former we exploit a large collection of textual descriptions, retrieving the most fitting caption for the given image. For the latter, we exploit BLIP-2 [140]–a VLM with remarkable performance on a variety of tasks, including image captioning — to provide a description for the image. In addition, we also use LLaVA 1.5 (7B) [160], a larger VLM. The last group of baselines consists of using a VQA model to directly predict the class name associated with the image. Again, we consider BLIP-2 [140] and LLaVA 1.5 (7B) [160], since they are highly effective also in VQA. We evaluate BLIP-2 over both ViT-L and ViT-g throughout the experiments [5].

Implementation details. Our experiments were conducted using NVIDIA A6000 GPUs with mixed-bit precision. As database, we use a subset of PMD [230], containing

---

[5] Following the BLIP-2 [140] demo, for captioning, we used the prompt “Question: what’s in the image? Answer:”. For VQA, we used “Question: what’s the name of the object in the image? Answer: a”. For LLaVA 1.5, we use the same prompts of BLIP-2, but append an “Omit any superfluous text.” to avoid introductory or closing statements.

five of its largest datasets: Conceptual Captions (CC3M) [222], Conceptual Captions 12M (CC12M) [26], Wikipedia Image Text (WIT) [235], Redcaps [62], and a subset of [242] used for PMD (YFCC100M*). We speed up the retrieval process by embedding the database via the text encoder $f^t_{\texttt{VLM}}$ and using fast indexing technique, *i.e.*, FAISS [109]. We tuned the α hyperparameter of eq. (3.6) and the number of retrieved captions K of our method on the ImageNet dataset, finding that $\alpha = 0.7$ and $K = 10$ led to the best results. We use these values across all experiments.

	Method	**Cluster Accuracy (%) ↑**										
		C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
CLIP	WordNet	34.0	20.1	16.7	16.7	58.3	40.9	52.0	29.4	18.6	39.5	32.6
	English Words	29.1	19.6	22.1	15.9	64.0	30.9	44.4	24.2	19.3	34.5	30.4
Caption	Closest Caption	12.8	8.9	16.7	13.3	28.5	13.1	15.0	8.6	20.0	17.8	15.5
	BLIP-2 (ViT-L)	26.5	11.7	23.3	5.4	23.6	12.4	11.6	19.5	14.8	25.7	17.4
	BLIP-2 (ViT-g)	37.4	13.0	25.2	10.0	29.5	19.9	15.5	21.5	27.9	32.7	23.3
	LLaVA 1.5 (7B)	41.1	11.1	19.7	10.4	13.4	11.1	12.8	14.0	12.1	29.5	17.5
VQA	BLIP-2 (ViT-L)	60.4	20.4	21.4	8.1	36.7	21.3	14.0	32.6	28.8	44.3	28.8
	BLIP-2 (ViT-g)	62.2	23.8	22.0	15.9	57.8	33.4	23.4	36.4	**57.2**	**55.4**	38.7
	LLaVA 1.5 (7B)	**76.2**	**30.6**	**38.9**	3.0	5.8	22.7	7.7	27.5	2.6	48.0	26.3
	CaSED	51.5	29.1	23.8	22.8	68.7	58.8	60.4	37.4	31.3	47.7	43.1
	UpperCaSED	51.3	29.3	21.0	**24.4**	**70.4**	**61.2**	**60.9**	**37.7**	38.5	46.6	**44.1**
	CLIP upper bound	87.6	52.9	47.4	31.8	78.0	89.9	88.0	65.3	76.5	72.5	69.0

Table 3.1: Cluster Accuracy on the ten datasets. Green is our method, gray shows the upper bound.

Quantitative results

The results of CaSED and the baselines are presented in table 3.1, table 3.2, and table 3.3 for Cluster Accuracy (%), Semantic Similarity, and Semantic IoU (%), respectively. CaSED consistently outperforms all baselines in all metrics on average and in most of the datasets. Notably, CaSED surpasses BLIP-2 (VQA) over ViT-g by +4.4% in Cluster Accuracy and +1.7% on Semantic IoU, while using much fewer parameters (*i.e.*, 102M *vs* 4.1B). The gap is even larger over the same visual backbone, with CaSED outperforming BLIP-2 on ViT-L (VQA) by +14.3% on Cluster Accuracy, +5.7 in Semantic Similarity, and +6.8% on Semantic IoU. These results highlight the effectiveness of CaSED, achieving the best performance both in terms of semantic relevance and image grouping.

An interesting observation from the tables is that simply applying CLIP on top of a pre-defined, large vocabulary, is not effective in VIC. This is due to the challenge of classifying over a huge search space, where class boundaries are hard to model. This is confirmed by the results of CLIP with English Words having a larger search space but

	Method	Semantic Similarity (x100) ↑										
		C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
CLIP	WordNet	48.6	32.7	24.4	18.9	55.9	49.6	53.7	44.9	28.8	44.2	40.2
	English Words	39.3	31.6	19.1	18.6	43.4	38.0	44.2	36.0	19.9	34.7	32.5
Caption	Closest Caption	42.1	23.9	23.4	29.2	40.0	46.9	40.2	39.8	49.2	40.3	37.5
	BLIP-2 (ViT-L)	57.8	31.4	39.9	24.4	36.1	44.6	29.0	45.3	46.4	38.0	39.3
	BLIP-2 (ViT-g)	63.0	33.1	36.2	24.3	45.2	51.6	31.6	48.3	61.0	44.6	43.9
	LLaVA 1.5 (7B)	56.8	29.4	40.5	21.3	31.1	36.9	24.8	42.5	38.1	37.9	35.9
VQA	BLIP-2 (ViT-L)	70.5	34.9	29.7	29.1	48.8	42.0	40.0	50.6	52.4	48.6	44.7
	BLIP-2 (ViT-g)	**73.5**	36.5	31.4	**30.8**	**59.9**	52.1	43.9	**53.3**	**65.1**	**55.1**	50.1
	LLaVA 1.5 (7B)	72.6	36.7	**44.1**	29.4	41.8	41.1	36.0	41.9	35.3	46.6	42.6
	CaSED	65.7	40.0	32.0	30.3	55.5	64.5	62.5	52.5	47.4	54.1	50.4
	UpperCaSED	66.3	**40.3**	34.9	27.0	56.1	**65.0**	**62.9**	53.0	46.5	53.8	**50.6**
	CLIP upper bound	90.8	69.8	67.7	66.7	83.4	93.7	91.8	80.5	92.3	83.3	82.0

Table 3.2: Semantic Similarity (x100) on the ten datasets. Values multiplied by x100 for readability. Green highlights our method and gray the upper bound.

	Method	Semantic IoU (%) ↑										
		C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
CLIP	WordNet	15.0	3.0	1.3	0.5	31.3	7.8	14.7	9.0	4.8	3.8	9.1
	English Words	8.0	2.0	0.0	1.1	16.4	2.0	17.2	8.1	2.7	1.8	5.9
Caption	Closest Caption	4.5	0.8	1.3	1.9	5.9	3.1	3.0	2.3	11.4	1.0	3.5
	BLIP-2 (ViT-L)	13.4	1.4	4.8	0.0	7.5	4.7	1.7	4.7	11.6	1.1	5.1
	BLIP-2 (ViT-g)	16.8	1.8	4.1	0.1	13.9	7.9	2.9	5.7	24.7	1.9	8.0
	LLaVA 1.5 (7B)	13.8	0.87	5.13	0.0	2.9	1.7	0.3	3.5	4.6	1.4	3.4
VQA	BLIP-2 (ViT-L)	36.1	1.8	7.0	0.1	21.5	3.7	5.7	11.5	18.9	2.5	10.9
	BLIP-2 (ViT-g)	**41.5**	2.4	**7.5**	2.0	**38.0**	8.6	10.2	13.8	**33.2**	2.8	16.0
	LLaVA 1.5 (7B)	41.4	0.4	6.1	0.0	5.0	3.0	0.7	6.6	1.3	2.0	6.6
	CaSED	35.4	**5.1**	2.3	**4.8**	33.1	**19.4**	**35.1**	17.2	16.2	**8.4**	17.7
	UpperCaSED	37.8	4.6	5.2	4.4	35.2	18.9	34.9	**17.8**	16.0	7.8	**18.3**
	CLIP upper bound	86.0	52.2	51.5	28.6	75.7	89.9	88.0	66.6	84.5	71.3	69.4

Table 3.3: Semantic IoU on the ten datasets. Green is our method, gray shows the upper bound.

performing consistently worse than CLIP with WordNet across all metrics (*e.g.*, -7.7 on Semantic Similarity, -3.2% on Semantic IoU).

Captioning models, despite their ability to capture image semantics in challenging settings, exhibit high variability across images of the same category, resulting in poor performance on Clustering and Semantic IoU. LLaVA 1.5 (VQA), while generally underperforming compared to BLIP-2 (VQA) and CaSED, does achieve the best results for cluster accuracy on Caltech-101, DTD, and EuroSAT. However, its performance on semantic IoU and semantic similarity is not significantly better than other approaches, and on average, it achieves unsatisfactory results across the ten datasets. These results highlight the efficacy of CaSED in all metrics. Finally, UpperCaSED consistently improves performance across all datasets, with an average gain of +1.0%, +0.6, and +0.2 on the three metrics *w.r.t.* CaSED.

Qualitative results

We report some qualitative results of our method applied on three different datasets, namely Caltech-101 (fig. 3.4), Food101 (fig. 3.5), and SUN397 (fig. 3.6), where the first is coarse, and the last two are fine-grained, focusing on food plates and places respectively. For each, we present a batch of five images, where the first three represent success cases and the last two show interesting failure cases. Each sample shows the image we input to our method with the top-5 candidate classes.

From the results, we can see that for many success cases, our method not only generates the correct class name and selects it as the best matching label, but it also provides valid alternatives for classification. For example, the third image in fig. 3.4 or the second image in fig. 3.5, where CaSED provides the names “dessert” for the cheesecake and the label “bird” for the ibis. This phenomenon also happens in failure cases, where, *e.g.*, the last sample in fig. 3.5 provides both the name “pizza” and the name “margherita” for the dish, despite selecting the wrong name from the set.

Another interesting observation is that our method provides names for different objects in the same scene. For instance, the third and fourth samples in fig. 3.5 contain labels for both “guacamole” and “tortillas” for the first, and for “mozzarella”, “insalata”, and “balsamic” for the second. A further detail on the latter case is the ability of CLIP to reason in multiple languages since “insalata” translates to “salad” from Italian to English.

Regarding failure cases, it is interesting to note that the candidate names and the predicted label often describe well the input image despite being wrong *w.r.t.* the dataset label. For instance, the two failure cases in fig. 3.6 select “stadium” and “dumpsite” when the ground-truth class names are “football” and “garbage site”. In addition, for the first case, the exact name “football” is still available among the best candidate names, but our method considers “stadium” as a better fit. Another example is the last failure case in fig. 3.4, where the model assigns the name “nokia” to a Nokia 3310 cellphone, while the ground-truth class name is “cellphone”. Also in this case, the ground-truth label is present in the candidate list but our method considers “nokia” a more fitting class.

Finally, we notice the discovery of correlations between terms in the reasoning of our model. In the provided examples, it happens multiple times that the candidate class names do not describe objects in the scene but rather a correlated concept to the image. For instance, the third example in fig. 3.4 shows a Dalmatian, and among the candidate names there is “cruella”, which is the name of the villain of the movie “The Hundred and One Dalmatians”. Another instance of this appears in the first example of

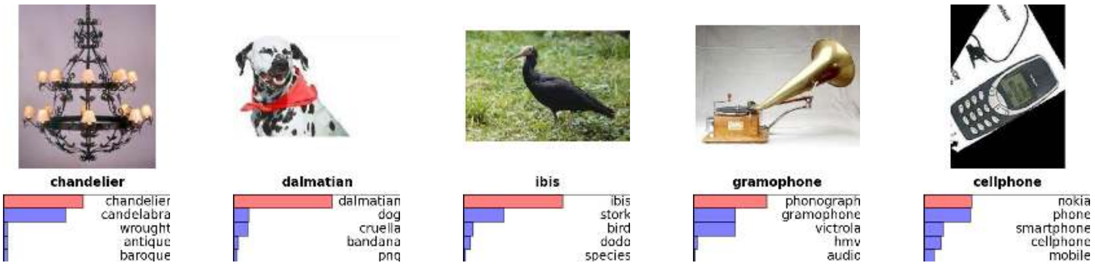


Figure 3.4: Qualitative results on Caltech-101. The first three samples represent success cases, the last two shows failure cases.

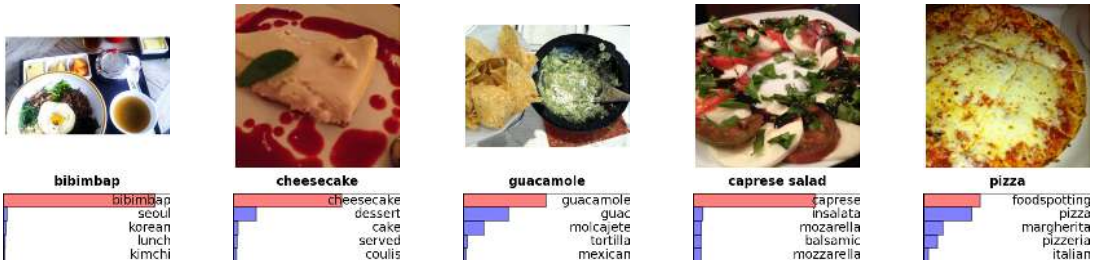


Figure 3.5: Qualitative results on Food101. The first three samples represent success cases, the last two shows failure cases.

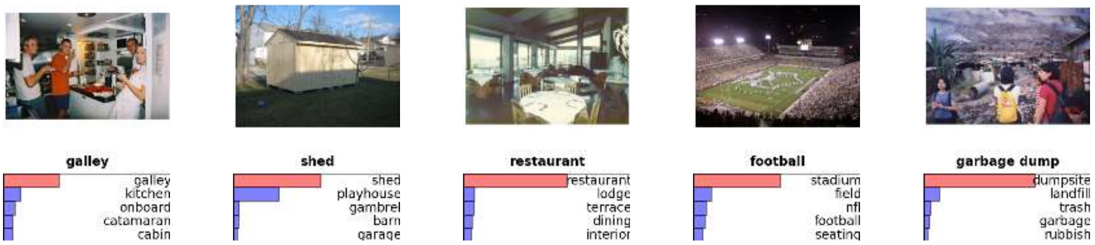


Figure 3.6: Qualitative results on SUN397. The first three samples represent success cases, the last two shows failure cases.

fig. 3.5, where the model correctly associates the “bibimbap” dish to its place of origin, Korea, with the candidate name “Korean”.

Ablation studies

In this section, we present the ablation study associated with different components of our approach. We first analyze the impact of our retrieval-based candidate selection and filtering. Then, we evaluate the impact of the multimodal scoring. We show the results of CaSED for different databases, and how the number of retrieved captions impacts the performance. Last, we show the performance of our model with different backbones and training objectives, and with other commonly used databases for retrieval. In regards

Candidates Generation	**Scoring**		**CA**	**S-Sim.**	**S-IoU**
	Vis.	Lang.			
Generative [140]	✓		23.3	47.1	11.9
Retrieval	✓		41.7	49.3	17.0
		✓	42.7	50.3	17.0
	✓	✓	**43.1**	**50.4**	**17.7**

(a) Ablation on candidate generation and scoring.

Database	**Size**	**CA**	**S-Sim.**	**S-IoU**
CC3M	2.8M	34.2	47.9	13.1
WIT	4.8M	34.6	42.9	12.1
Redcaps	7.9M	42.0	49.5	17.2
CC12M	10.3M	**44.0**	**51.3**	**18.3**
YFCC100M*	29.9M	40.7	48.8	17.1
All	54.8M	43.1	50.4	17.7

(b) Ablation on the database.

Table 3.4: Ablation studies. Metrics are averaged across the ten datasets. Green highlights our setting. **Bold** represents best, underline indicates second best.

to UpperCaSED, we demonstrate how prompt ensemble improves the performance of retrieval-based baselines compared to image-text comparison.

Candidates generation. We consider two options to generate the set of candidate classes. The first uses BLIP-2 (ViT-g), asking for multiple candidate labels for the input image. The second is our caption retrieval and filtering strategy. Table 3.4a shows the results where, for fairness, we score both sets with the same VLM (*i.e.*, CLIP ViT-L). Our approach consistently outperforms BLIP-2 across all metrics (*e.g.*, 41.7 vs 23.3 for Cluster Accuracy). This confirms the preliminary results of our study in fig. 3.2, with retrieved captions providing better semantic priors than directly using a powerful VLM.

Candidate filtering. We test different configurations of candidate transformations in table 3.5. When compared to having no operation for candidate filtering (first row), the "remove" operations eliminates inappropriate content and increases the accuracy of clusters by +6.4% and improves the semantic IoU by +0.3. However, we can observe a drop in semantic similarity by -1.5. This might be due to the removal of unnatural words that could still describe well the content of the image, *i.e.*, underline- or dash-separated words, or URLs since they are longer *w.r.t.* natural words. When we add standardization operations, we observe a sizeable boost in terms of performances when compared to the results obtained by applying only removal-related operations, achieving higher results across all three metrics, and leading to a relative improvement of +7.2%, +0.7, and +1.2 in terms of cluster accuracy, semantic similarity, and semantic IoU, respectively. Last, the inclusion of the "filter" set of operations scores the best among all three metrics when compared to the results obtained when only the previous two groups of operations are applied.

Multimodal scoring. The second ablation studies the effect of our proposed multimodal scoring vs its unimodal counterparts. As table 3.4a shows, multimodal candidate scoring provides the best results across all metrics, with clear improvements over the visual modality alone (*i.e.*, +1.4% of Cluster Accuracy). Notably, scoring via language

Operations			**CA**	**S-Sim.**	**S-IoU**
Remove	**Standardize**	**Filter**			
			27.7	48.8	15.0
✓			34.1	47.3	15.3
✓	✓		41.3	48.0	16.5
✓	✓	✓	**43.1**	**50.4**	**17.7**

Table 3.5: Ablation on the candidate filtering operations. Metrics are averaged across the ten datasets.

Num. captions	**CA**	**S-Sim.**	**S-IoU**	**Candidates**	**Selected**
1	30.9	43.2	12.4	3849	1047
2	35.0	46.4	14.6	6867	854
5	42.3	**50.6**	17.5	14548	775
10	**43.1**	50.4	**17.7**	22475	794
20	42.9	49.3	17.1	28781	802

Table 3.6: Ablation on the number retrieved captions. Green represents our selected configuration. Alongside the metrics, we show the number of unique words extracted from captions (*i.e.*, "candidates") and the number of words selected by CaSED for classification (*i.e.*, "selected"). Results are averaged across the ten datasets.

only partially fills the gap between visual and multimodal scoring (*e.g.*, +1% on Cluster Accuracy and +1 on Semantic Similarity), confirming that caption centroids contain discriminative semantic information.

Retrieval database. We analyze the impact of retrieving captions from different databases, using five public ones, *i.e.*, CC3M, WIT, Redcaps, CC12M, and a subset of YFCC100M. The databases have different sizes (*e.g.*, from 2.8M captions of CC3M to 29.9M captions of the YFCC100M subset), and different levels of noise. As shown in table 3.4b, the results tend to improve as the size of the database increases (*e.g.*, +8.9% on Cluster Accuracy over CC3M). However, the quality of the captions influences the performance, with CC12M and Redcaps alone achieving either comparable or slightly better results than the full database. These results suggest that while performance improves with the size of the database, the quality of the captions has a higher impact than the mere quantity.

Number of captions. Finally, we check how performance varies *w.r.t.* the number of retrieved captions. As shown in table 3.6, all metrics consistently improve as the number of captions increases from 1 to 10. After that, performance tends to saturate, with a slight decrease in terms of Semantic Similarity for $K = 20$. These results suggest that while a minimum number of captions is needed to fully capture the semantics of the image, the possible interference of less-related (or even noisy) captions may impact

Backbone	**Size (M)**	**CA**	**S-Sim.**	**S-IoU**
ResNet-50	102	32.7	50.1	15.4
ViT-L/14	427	43.5	56.9	21.2
ViT-L/16 (SigLIP-384)	652	**47.8**	**59.4**	**22.5**

Table 3.7: Ablation on CaSED backbone. Results are collected on ImageNet, using CC12M as the text database. Sizes are taken from [103].

the final performance. Future research may further improve the performance on VIC by focusing on how to deal with noisy retrieval results. We additionally report the number of unique candidates extracted by the candidate filtering procedure, averaged over the ten datasets, and with an increasing number of retrieved captions. In the table, we show both the number of candidates extracted and the number of selected words. As the number of retrieved captions increases, the unique number of candidate words also increases, *i.e.*, from 3849 with 1 caption to 28781 with 20. However, the number of selected words stabilizes around 800 as soon as we retrieve more than 1 caption. Having more captions reduces the noises in the selected words, something that might be present when relying on a single caption.

Backbone architecture. To answer this natural question about whether the outcome of our model depends on the backbone architecture, we further extend our main results with a CLIP model with a ResNet50 architecture. We report this additional ablation in table 3.8, table 3.9, and table 3.10 for the cluster accuracy, the semantic IoU, and the semantic similarity, respectively. We can see that the performance with the CLIP ResNet50 is lower across all the metrics compared to CLIP ViT-L/14. This is expected since ResNet50 is a a smaller architecture, thus with a reduced capacity for semantic representation learning as compared to ViT-L/14. Nevertheless, our method with ResNet50 is still competitive against BLIP-2 models while using 40x fewer parameters (note that our ViT-L implementation uses 10x fewer parameters).

We additionally test CaSED considering the more recent CLIP instance trained with a sigmoid loss [291] on WebLI [33]. For this analyses, we use the smaller CC12M as retrieval database instead of PMD, as they achieve comparable performance in table 3.4b. We report the results in table 3.7 Notably, our approach shows positive changes when equipped with an higher number of parameters and better training losses, hinting that larger VLMs can lead to higher VIC performance.

Additional retrieval database. We analyze the impact of retrieving captions from databases of different scales, expanding our main results with the four unused databases of PMD, including COCO [159], SBU Captions [191], Localized Narratives [200], and

	Method		**Cluster Accuracy (%) ↑**										
			C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
CLIP RN50	CLIP	WordNet	30.3	18.3	**22.5**	13.2	47.8	31.4	45.2	26.0	14.2	31.2	28.0
		English Words	24.8	17.5	18.5	13.4	49.5	23.1	36.6	22.2	15.5	27.1	24.8
	Caption	Closest Caption	9.7	7.1	13.3	8.4	21.2	6.2	8.7	6.5	12.8	14.9	10.9
	CaSED		**44.6**	**23.9**	12.5	**15.3**	**58.8**	**48.7**	**50.1**	**32.8**	**24.6**	**33.9**	**34.5**
	CLIP upper bound		82.1	41.5	33.5	19.6	63.1	74.6	78.9	55.6	54.9	58.4	56.2
CLIP ViT-L/14	CLIP	WordNet	34.0	20.1	16.7	16.7	58.3	40.9	52.0	29.4	18.6	39.5	32.6
		English Words	29.1	19.6	22.1	15.9	64.0	30.9	44.4	24.2	19.3	34.5	30.4
	Caption	Closest Caption	12.8	8.9	16.7	13.3	28.5	13.1	15.0	8.6	20.0	17.8	15.5
	CaSED		**51.5**	**29.1**	**23.8**	**22.8**	**68.7**	**58.8**	**60.4**	**37.4**	**31.3**	**47.7**	**43.1**
	CLIP upper bound		87.6	52.9	47.4	31.8	78.0	89.9	88.0	65.3	76.5	72.5	69.0
	Caption	BLIP-2 (ViT-L)	26.5	11.7	23.3	5.4	23.6	12.4	11.6	19.5	14.8	25.7	17.4
		BLIP-2 (ViT-g)	37.4	13.0	25.2	10.0	29.5	19.9	15.5	21.5	27.9	32.7	23.3
	VQA	BLIP-2 (ViT-L)	60.4	20.4	21.4	8.1	36.7	21.3	14.0	32.6	28.8	44.3	28.8
		BLIP-2 (ViT-g)	62.2	23.8	22.0	15.9	57.8	33.4	23.4	36.4	57.2	55.4	38.7

Table 3.8: Cluster Accuracy on the ten datasets. Green is our method, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models.

Visual Genome [123]. Moreover, we evaluate our method on other three databases, namely Ade20K [304], OpenImages [129], and LAION-400M [218]. These databases further extend the range of database sizes, with Ade20k containing only 0.07M captions and LAION-400M having 413M. We report the results in table 3.11, ordering the rows by the database size. Differently from the tables reported in the main results, the results in table 3.11 are obtained on ImageNet. We first discuss the databases belonging to the PMD superset, and then analyze the results obtained with Ade20K, OpenImages, and LAION-400M. In the table, we also show the POS tag distribution (*e.g.*, nouns, adjectives, verbs) of each dataset. Moreover, we report a metric, the semantic similarity to the closest caption in the dataset of each image (C.C. S-Sim) to check how close a database is to the target one (*i.e.*, ImageNet).

By comparing the nine databases of PMD, we can perceive a non-uniform variation in performance across databases, with a remarkable gap between the top-5 (highlighted in blue) and the rest. The performance of the fifth-performing (*i.e.*, WIT) and the sixth database (*i.e.*, SBU Captions) differs by 12.8% on cluster accuracy, while the difference between the best-performing (*i.e.*, CC12M) and the fifth is 10.9%. This observation motivates us to use the top-5 databases from PMD instead of the full superset to be efficient. We expand the best database with the other top-5 to ensure better coverage of class names in the textual descriptions.

With the additional databases, *i.e.*, Ade20K, OpenImages, and LAION-400M, we further notice that there is a correlation between the size of a database and its performance. This was also noted in our results reported in the main manuscript. Ade20K,

	Method		C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
			Semantic IoU (%) ↑										
CLIP RN50	CLIP	WordNet	9.8	1.9	**0.9**	0.0	21.2	5.5	14.1	7.3	3.5	2.6	6.7
		English Words	5.1	0.8	0.0	0.1	12.4	1.8	13.2	7.8	2.5	1.3	4.5
	Caption	Closest Caption	3.4	0.5	0.1	1.5	6.6	2.2	2.2	2.1	9.1	0.7	2.8
	CaSED		**31.1**	**2.9**	0.08	**2.8**	**29.7**	**15.1**	**27.6**	**15.0**	**13.4**	**5.2**	**14.3**
	CLIP upper bound		81.5	41.4	33.9	16.5	60.2	74.6	78.9	56.9	66.5	57.1	56.8
CLIP ViT-L/14	CLIP	WordNet	15.0	3.0	1.3	0.5	31.3	7.8	14.7	9.0	4.8	3.8	9.1
		English Words	8.0	2.0	0.0	1.1	16.4	2.0	17.2	8.1	2.7	1.8	5.9
	Caption	Closest Caption	4.5	0.8	1.3	1.9	5.9	3.1	3.0	2.3	11.4	1.0	3.5
	CaSED		**35.4**	**5.1**	**2.3**	**4.8**	**33.1**	**19.4**	**35.1**	**17.2**	**16.2**	**8.4**	**17.7**
	CLIP upper bound		86.0	52.2	51.5	28.6	75.7	89.9	88.0	66.6	84.5	71.3	69.4
	Caption	BLIP-2 (ViT-L)	13.4	1.4	4.8	0.0	7.5	4.7	1.7	4.7	11.6	1.1	5.1
		BLIP-2 (ViT-g)	16.8	1.8	4.1	0.1	13.9	7.9	2.9	5.7	24.7	1.9	8.0
	VQA	BLIP-2 (ViT-L)	36.1	1.8	7.0	0.1	21.5	3.7	5.7	11.5	18.9	2.5	10.9
		BLIP-2 (ViT-g)	41.5	2.4	7.5	2.0	38.0	8.6	10.2	13.8	33.2	2.8	16.0

Table 3.9: Semantic IoU on the ten datasets. Green is our method, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models.

the smallest dataset, comprises only about 0.07M captions and obtains the lowest performance among all metrics. As the size of databases increase, the performance generally improves. However, it is important to note that dataset size alone is not the only requirement for good results. LAION-400M is of a much larger size than our split of PMD, yet the performance with LAION-400M is worse, which we attributes to the impact of much noisier retrieval from database of such scale. Instead, datasets such as CC12M and Redcaps perform better than LAION-400M despite being approximately 40x smaller. This finding suggests that the quality of the databases could be of a higher importance than its size.

Another observation pertains to the differences between captioning and visual question answering (VQA) datasets (*e.g.*, COCO, Localized Narratives, and Visual Genome) *w.r.t.* image-text datasets collected from the web (*e.g.*, CC12M, Redcaps, and LAION-400M). In general, it appears that textual descriptions from captioning and VQA datasets perform worse than datasets scraped from the web. This difference in performance may be due to the type of description provided, as the former often describes specific regions of the image, while the latter provides a general description of the image as a whole. Since our prime objective is to understand the class name of the subject in the image (whether it be a location, a pet, or a car model), captions describing secondary objects can lead to sub-optimal results.

Interestingly, we found that the average performance on the 10 datasets of a chosen textual database generally correlates with the accuracy on ImageNet and with the closest caption semantic similarity (C.C. S-Sim in table 3.11). Examples are the subset

Method			Semantic Similarity (x100) ↑										
			C101	DTD	ESAT	Airc.	Flwr	Food	Pets	SUN	Cars	UCF	**Avg.**
CLIP RN50	CLIP	WordNet	43.2	29.0	18.5	21.6	46.7	44.6	50.3	42.8	26.4	40.0	36.3
		English Words	36.0	29.5	14.9	20.0	38.1	34.2	40.7	35.4	18.3	32.4	29.9
	Caption	Closest Caption	37.2	22.8	14.2	26.8	38.9	41.2	32.6	37.4	**44.3**	32.4	32.8
	CaSED		**62.3**	**36.4**	**22.6**	**28.7**	**52.8**	**59.0**	**57.0**	**50.2**	42.9	**46.2**	**45.8**
	CLIP upper bound		88.1	62.4	52.8	53.9	72.0	83.7	85.8	73.9	81.0	73.9	72.7
CLIP ViT-L/14	CLIP	WordNet	48.6	32.7	24.4	18.9	**55.9**	49.6	53.7	44.9	28.8	44.2	40.2
		English Words	39.3	31.6	19.1	18.6	43.4	38.0	44.2	36.0	19.9	34.7	32.5
	Caption	Closest Caption	42.1	23.9	23.4	29.2	40.0	46.9	40.2	39.8	**49.2**	40.3	37.5
	CaSED		**65.7**	**40.0**	**32.0**	**30.3**	55.5	**64.5**	**62.5**	**52.5**	47.4	**54.1**	**50.4**
	CLIP upper bound		90.8	69.8	67.7	66.7	83.4	93.7	91.8	80.5	92.3	83.3	82.0
	Caption	BLIP-2 (ViT-L)	57.8	31.4	39.9	24.4	36.1	44.6	29.0	45.3	46.4	38.0	39.3
		BLIP-2 (ViT-g)	63.0	33.1	36.2	24.3	45.2	51.6	31.6	48.3	61.0	44.6	43.9
	VQA	BLIP-2 (ViT-L)	70.5	34.9	29.7	29.1	48.8	42.0	40.0	50.6	52.4	48.6	44.7
		BLIP-2 (ViT-g)	73.5	36.5	31.4	30.8	59.9	52.1	43.9	53.3	65.1	55.1	50.1

Table 3.10: Semantic similarity on the ten datasets. Green is ours, gray shows the upper bound. **Bold** represents best, underline indicates best considering also image captioning and VQA models.

of PMD and CC12M, achieving the best average results on the 10 datasets according to all metrics, while being the best on the ImageNet validation set *w.r.t.* both the closest caption semantic similarity and the ViC metrics. The ImageNet validation set can thus be used as a proxy to pick the textual database in case of a lack of priors on the test set.

For what concerns the POS tag distribution, it is hard to extract any pattern that justifies the performance of different databases. For instance, CC3M is the only database without an extreme imbalance for nouns (*i.e.*, 57.9% compared to >96% for the other in the top-5) while still achieving good performance. Moreover, while the number of concepts depends on the size of the database, there is no clear correlation between this value and the achieved scores. As an example, Localized Narratives has a comparable amount of concepts *w.r.t.* CC3M but a gap of >20% in cluster accuracy, and of 10 points in semantic similarity and IoU. Finally, it is worth noting that most of the datasets have an average of 12.6 words per caption, while Visual Genome has an average of only 5.1 words per caption. This characteristic may explain why Visual Genome behaves differently in terms of the database scaling rule of increasing performance with the database size.

Prompt ensemble. We report the performance of prompt ensembling on all the retrieval-based baselines, *i.e.*, English Words, WordNet, and CaSED in table 3.12. The results show consistent improvements across all baselines when using multiple templates. This outcome is consistent across the three metrics, except for one configuration, *i.e.*, cluster accuracy for English Words. On average, prompt ensemble improves

Database	**Size**	**CA**	**S-Sim.**	**S-IoU**	**Nouns** (%)	**Adjs** (%)	**Verbs** (%)	**Concepts**	C.C. S-Sim.
Ade20K	0.07M	11.0	31.8	1.7	82.7	10.6	6.7	3.1K	19.0
COCO	0.9M	13.8	35.6	2.6	86.1	9.5	4.4	14.6K	22.6
SBU Captions	1.0M	20.1	40.8	5.7	98.0	1.0	0.9	29.3K	26.9
OpenImages	1.4M	16.8	37.8	4.2	85.4	10.0	4.5	21.4K	24.8
Loc. Narr.	1.9M	15.7	37.1	4.1	86.9	9.8	3.2	30.9K	24.9
CC3M	2.8M	37.9	53.9	16.2	57.9	23.6	18.5	28.4K	35.2
WIT	4.8M	32.9	47.9	14.5	99.8	0.05	0.07	163.7K	30.0
Visual Genome	5.4M	14.6	35.2	3.8	75.1	15.6	9.2	21.1K	26.7
Redcaps	7.9M	41.1	54.7	19.6	98.6	0.7	0.6	50.5K	37.5
CC12M	10.3M	**43.8**	**57.4**	**21.2**	96.6	1.8	1.5	36.7K	**38.7**
YFCC100M*	29.9M	39.1	53.5	18.9	99.7	0.2	0.1	179.6K	37.8
Ours	54.8M	41.5	55.7	20.4	99.8	0.05	0.05	334.1K	**38.7**
LAION-400M	413.8M	37.7	52.7	18.7	99.4	0.5	0.1	1.68M	37.3

Table 3.11: Ablation on the databases on ImageNet. Green is our database, Blue shows the top-5 databases of PMD, which we used to create Ours. We also evaluate the semantic similarity of the closest caption to each image in the dataset (*i.e.*, C.C. S-Sim.). **Bold** represents best, while underline second best. YFCC100M* indicates we use the subset of the dataset used in PMD.

Vocabulary	**Prompt ensemble**	**CA**	**S-Sim.**	**S-IoU**
WordNet		32.6	40.2	9.1
	✓	33.3	41.8	10.1
English Words		30.4	32.5	5.9
	✓	29.6	34.3	7.0
Ours		43.1	50.4	17.7
	✓	44.1	50.6	18.3

Table 3.12: Ablation on prompt ensembling. Results are averaged on the ten classification datasets. Green is our configuration.

performance by +0.3% on cluster accuracy, +1.2 on semantic similarity, and +0.9 on semantic IoU.

Computational cost

We analyze the computational efficiency of CaSED *vs* BLIP-2 and LLaVA 1.5 performing VQA and captioning, and report their respective number of parameters and inference time in table 3.13. Notably, the methods using external databases are consistently faster than BLIP-2 and LLaVA. For instance, CaSED achieves a speed-up of 2x with respect to both the largest captioning model and the largest VQA model, while also achieving better performance. In a similar fashion, our method is approximately 3.0x faster than LLaVA 1.5 for VQA and 3.5x for captioning. Overall, the fastest method is Closest Caption, which exploits the external database to retrieve a single caption and does not consider any candidate extraction pipeline. Conversely, our method retrieves the ten most similar captions and post-processes them to extract class name, resulting in a increase in inference time of approximately 2 times. Compared with the CLIP up-

Method		**Num. Params.**	**Inference time (ms) ↓**
Caption	Closest Caption	0.43B	1390 ± 10
	BLIP-2 (ViT-L)	3.46B	5710 ± 153
	BLIP-2 (ViT-g)	4.37B	6870 ± 177
	LLaVA 1.5 (7B)	7.49B	8260 ± 20
VQA	BLIP-2 (ViT-L)	3.46B	5670 ± 135
	BLIP-2 (ViT-g)	4.37B	6650 ± 117
	LLaVA 1.5 (7B)	7.49B	7040 ± 28
CaSED		0.43B	2630 ± 14
CaSED (LAION-400M)		0.43B	2640 ± 11
CLIP upper bound		0.43B	645 ± 77

Table 3.13: Computational cost of different methods. Green is our method, gray shows the upper bound. Inference time is reported on batches of size 64, as the average over multiple runs.

per bound, our method considers multiple additional steps, each adding extra inference time. First, our method retrieves candidates from the external database to then extract the class names. Second, we have to forward the class names through the text encoder for each sample, while CLIP can forward them once and cache the features for later reuse.

When using an external database, note that an increase in database size implies a minimal variation in retrieval time. This is demonstrated by the computational cost required by retrieving from the large LAION-400M [218] database. As the results show, inference time is comparable between CaSED and CaSED (LAION-400M) despite the latter being approximately 8 times larger.

3.1.5 Related work

Vision-Language Models. Leveraging large-scale datasets with image-text pairs [218, 217, 230], recent works train models by mapping the two modalities into a shared representation space [110, 84, 61, 206, 107, 143, 74]. A notable example is CLIP [206] which, using modality-specific encoders and a contrastive objective to align their output representations, showed remarkable performance on zero-shot classification. Subsequent works improved CLIP by *e.g.*, connecting the two modalities via cross-modal attention [143], multi-object representation alignment [289], learning from weak-supervision [254] or unaligned data [230].

Another line of works improved vision-language pre-training for complex vision-language tasks, such as image captioning and visual question answering (VQA) [276, 99, 142, 3]. In this context, BLIP [142] exploits web data and generated captions to supervise pre-training of a multimodal architecture, outperforming existing VLMs on both captioning and VQA. The current state-of-the-art method BLIP-2 [140] trains a module connecting the two modalities on top of a frozen visual encoder and a frozen large-language model, enabling instructed zero-shot image-to-text generation.

In this work, we challenge a fundamental assumption of zero-shot classification with VLMs: the set of target classes is known a priori. We propose a new task, VIC, which sidesteps this assumption, performing classification in a language-induced open-ended space of semantic categories. We show that even BLIP-2 struggles in this scenario while external multimodal databases provide valuable priors for inferring the semantic category of an image. As a final note, VIC differs from open-vocabulary recognition (*e.g.*, [286, 82]) since the latter assumes that the list of target classes is known and available to the model during inference.

Retrieval augmented models. In natural language processing, multiple works showed the benefit of retrieving information from external databases, improving the performance of large language models [88, 134, 17]. In computer vision, such a paradigm has been used mostly to deal with the imbalanced distribution of classes. Examples are [165, 166], addressing long-tail recognition by learning to retrieve training samples [165] or image-text pairs from an external database [166]. Similarly, [245] retrieves images from a given dataset to learn fine-grained visual representations. More recently, retrieval-augmentation has been extended to various types of sources for visual question answering [100], as well as to condition the generative process in diffusion models [16], or image captioning [208]. Our work is close in spirit to [166], as we exploit an external database. However, [166] assumes a pre-defined set of classes (and data) available for training a retrieval module, something we cannot have for the extremely large semantic space of VIC. In CaSED, retrieval is leveraged to first create a set of candidate classes, and to then perform the final class prediction. Moreover, we assume the database to contain only captions, and not necessarily paired image-text data, thus being less memory-demanding.

Finally, the performance of retrieval models is largely affected by the employed database. In the context of VLMs, researchers collected multiple open datasets to study vision-language pre-training. Two notable examples are LAION-5B [218], collected by filtering Common Crawl [53] via CLIP, and the Public Multimodal Datasets (PMD) [230], collecting image-text pairs from different public datasets, such as Conceptual Captions [222, 26], YFCC100M [242], Wikipedia Image Text [235], and Redcaps [62]. In our experiments, we use a subset of PMD as database and investigate how classification performance varies based on the size and quality of the database.

3.2 Semantic segmentation with Vision Language Models

3.2.1 The problem

Large-scale Vision Language Models (VLMs) [206, 276, 142] have revolutionized the field of computer vision, connecting multimodal information in an unprecedented manner. One peculiar aspect of these models is their zero-shot transfer capabilities: for instance, CLIP [206] showed outstanding classification results in multiple datasets, even if not being explicitly trained for the task at hand. This lead to extending VLMs to other discriminative tasks, such as semantic segmentation [151, 264, 277] or object detection [173, 127], where their multimodal nature allows to perform such tasks in an "open-vocabulary" manner, *i.e.*, where the (finite) set of categories can be dynamically defined by the user.

In the previous chapter, we aimed to challenge the latter assumption and perform classification tasks with VLMs *without* a set of target categories (*i.e.*, the vocabulary) pre-defined by the user (see fig. 3.1). This has many practical advantages as this lack of priors often arises when working with autonomous agents in unconstrained environments. At the same time, it inherits various challenges as (i) the search space encompasses all possible existing semantic concepts, even very fine-grained ones that are difficult to discriminate and possibly ambiguous; (ii) we need classification models that do not rely on vocabulary-aware supervision, to avoid potential biases on the subpart of the vocabulary within the training data. We named this task Vocabulary-free Image Classification (VIC).

Our approach exploits two core elements: multimodal representations from a contrastive VLM, *i.e.*, CLIP [206], and the information included in large scale Vision Language Databases (VLD), *e.g.*, PMD [230]. For classification, given an image we retrieve its closest captions in a VLD, encoding both input and captions via the CLIP encoders. We then parse these captions to obtain a set of candidate class names for the input. We encode the candidates via the text encoder of CLIP, scoring them according to their similarity with the visual input and the centroid of the retrieved captions, performing multimodal matching. We named this approach Category Search from External Databases (CaSED). On a variety of VIC benchmarks, CaSED achieves higher performance than computationally more expensive VLMs for VQA.

To further demonstrate the effectiveness of our proposed approach, in this chapter we extend CaSED for the task of Vocabulary-free Semantic Segmentation (VSS). In contrast to the limited nature of image-level classification, often overlooking objects in the background, semantic segmentation aligns with the challenges of unconstrained environments, containing unforeseen objects that cause ambiguities in defining a fixed set of classes. In particular, when faced with the absence of predefined categories, the segmentation task becomes more complex, prompting questions about the appropriate granularity, *e.g.*, whether to segment object parts or the entire object itself. Moreover, segmentation poses new challenges for CaSED, given the tendency of Vision Language Models to recognize foreground objects and ignore the background, and the object-centric nature of internet-sourced captions.

In this context, we explore different strategies: the first is to use an off-the-shelf segmenter to obtain an initial set of masks. CaSED can then assign a semantic label to each mask independently. A second strategy is to do the opposite: CaSED can provide estimates of the semantic categories that can then be processed by an open-vocabulary segmentation model. Finally, we may avoid any external segmentation model and only employ a single pre-trained VLM, without additional fine-tuning. Encoding non-overlapping patches separately and classifying their content may seem a good strategy, but we found that it leads to noisy results because a single patch cannot capture the surrounding visual context. To address this issue, we first encode local information of the image by dividing it into cells of different sizes and processing them via CLIP. These multi-scale representations are then accumulated locally to obtain a more precise dense visual representation. The latter undergoes the same CaSED processing, retrieving a set of captions and candidate categories for each local representation. We then apply multimodal scoring on each cell, obtaining the final, dense semantic prediction. We name this approach DenseCaSED. Experiments show that DenseCaSED and CaSED variants outperform various semantic segmentation models in multiple benchmarks, *without* requiring any training procedure.

To summarize, the contributions of this work are:

- We extend the tasks of VIC to VSS, where the goal is to segment images *without* any pre-defined set of target categories, operating directly on an unconstrained semantic space. We define specific metrics for these tasks, capturing the semantic between predictions and ground-truth labels, providing a reference for future research.

- We extend Category Search from External Databases, a training-free method for VIC, for VSS, presenting three variants. While two of them couple CaSED with pretrained segmentation models, the third, DenseCaSED directly exploits a VLM and multi-scale image processing to obtain local visual representations, that are used to retrieve and score candidates at a local level, providing a dense semantic map of the input without any class priors or training.

- Our extensive evaluation on different benchmarks (*i.e.*, PascalVOC-20 [66], PASCAL Context-59 [185], and ADE20K-150 [305]) and different VLM-based models, demonstrate the efficacy of CaSED and DenseCaSED for VSS, highlighting the potential of VLM plus retrieval as a pipeline for semantic categorization tasks with an unconstrained vocabulary.

3.2.2 The proposed setting

Preliminaries. Let us denote as $\mathcal{X} \subset \mathbb{R}^{N \times 3}$ the image space, where N is the number of pixels. Moreover, we can define as $\mathcal{C}$ a set of class labels. These labels are semantic entities in the much larger space of all possible semantic concepts $\mathcal{S}$, *i.e.*, $\mathcal{C} \subset \mathcal{S}$. A segmenter, is a function f mapping pixels to semantic labels in $\mathcal{C}$, with $f : \mathcal{X} \to \mathcal{C}^N$. While f is usually trained on paired samples of images and labels, this approach is costly and does not scale with the cardinality of $\mathcal{C}$, as it may require expensive manual annotation. VLMs [206, 106] removed the need for explicit annotations, measuring similarities between image and text descriptions, *i.e.*, $f_{\texttt{VLM}} : \mathcal{X} \times \mathcal{T} \to \mathbb{R}$, with $\mathcal{T}$ the textual space. In this way, we can perform classification by:

$$f(\boldsymbol{x}) = \arg\max_{\boldsymbol{c} \in C} f_{\texttt{VLM}}(\boldsymbol{x}, \phi(\boldsymbol{c})) \tag{3.8}$$

where $\phi(\boldsymbol{c})$ denotes a text concatenation, merging a static text template, or *prompt*, with a class name. Segmentation is achieved in a similar manner, computing the similarity between text and local image representations [151]. Note that this does not require any training and the model can classify/segment images into new categories defined at test time without retraining, performing zero-shot transfer. Nevertheless, this approach assumes that the set of categories $\mathcal{C}$ is given a priori by a user. In the previous chapter, we described Vocabulary-free Image Classification (VIC) to overcome this limitation in classification. Here, we introduce its counterpart Vocabulary-free Semantic Segmentation (VSS) for segmentation.

Task definition. The goal of the vocabulary-free settings is to either classify (VIC) or segment (VSS) an image $\boldsymbol{x}$ without any prior knowledge about $\mathcal{C}$. Specifically, this means operating directly on the vast semantic space $\mathcal{S}$, which encompasses all semantic concepts. For VIC, we aim to devise a function $f : \mathcal{X} \rightarrow \mathcal{S}$ that maps an image to a semantic label within $\mathcal{S}$. Similarly, for VSS, the target function is $f : \mathcal{X} \rightarrow \mathcal{S}^N$ mapping images to semantic maps. Note that, at test time, f relies solely on the input image $\boldsymbol{x}$ and a broad repository of semantic concepts approximating $\mathcal{S}$.

Challenges. Both VIC and VSS share the challenge of identifying which semantic categories in the large set $\mathcal{C}$ are present in the input image [48]. In particular, VSS is hard as $\mathcal{C}$ contains a lot of potential distractors, *i.e.*, concepts related to the ones in the image but not present. Examples are *couch vs sofa*, *tv vs monitor*, but also different animal species of plants. Thus, addressing VSS requires models with fine-grained recognition capabilities. On the other hand, a model may also segment two regions using synonyms (*e.g.*, *lawn vs grass*, *road vs highway*): these cases need to be disambiguated to obtain coherent segmentation masks. This also relates to other issues, such as the granularity of the segmentation masks (*e.g.*, parts *vs* entire objects), or the object-centric focus of VLMs and internet-sourced captions. The latter, tend to ignore objects or elements in the background, making it hard to provide extremely fine-grained segmentation masks.

In the next sections, we describe how we address these challenges by (i) constraining the output space via external captions; (ii) disambiguating semantic concepts via multimodal matching, and (iii) propagating local features.

3.2.3 The proposed approach

In the following, we quickly review Category Search from External Databases (CaSED) [48], for tackling VIC. The method leverages the power of large Vision Language Databasess (VLDs) to find the best matching category within an unconstrained semantic space. Then, we present how CaSED can be extended to semantic segmentation, either by application on top of pretrained segmentation network or by modifying how CLIP processes the input image (DenseCaSED).

Category Search from External Databases

CaSED is built on two modules: (i) candidate categories generation an (ii) multimodal scoring of the list of candidates.

Candidate category generation. We initially narrow down the vast classification space $\mathcal{S}$ to few probable candidate classes. Given an input $\boldsymbol{x}$, we use the pre-trained

VLM $f_{\texttt{VLM}}$ and an external image captions database D to retrieve a subset $D_{\boldsymbol{x}} \subset D$ of K closest captions to the input image as

$$D_{\boldsymbol{x}} = \underset{\boldsymbol{d} \in D}{\text{top-k}}\ f_{\texttt{VLM}}(\boldsymbol{x}, \boldsymbol{d}) = \underset{\boldsymbol{d} \in D}{\text{top-k}}\ \langle f^v_{\texttt{VLM}}(\boldsymbol{x}), f^t_{\texttt{VLM}}(\boldsymbol{d}) \rangle, \tag{3.9}$$

where $f^v_{\texttt{VLM}} : \mathcal{X} \to \mathcal{Z}$ and $f^t_{\texttt{VLM}} : \mathcal{T} \to \mathcal{Z}$ are the visual and textual encoders of the VLM, respectively, and $\mathcal{Z}$ is their shared embedding space. The operation $\langle \cdot, \cdot \rangle$ computes the cosine similarity between the two. Our approach can accommodate varying database sizes and is not dependent on the specific form of D. We then extract a finite set of candidate classes $C_{\boldsymbol{x}}$ from $D_{\boldsymbol{x}}$ using basic text parsing and filtering techniques.

Multimodal candidate scoring. We score each candidate in the set $C_{\boldsymbol{x}}$ using both visual and textual semantic similarities via the VLM encoders to identify the best-matching class for the input image. We denote $s^v_{\boldsymbol{c}}$ as the visual score of each candidate category $\boldsymbol{c}$, computed as the similarity between the visual representation of the input image and the textual representation of the candidate name:

$$s^v_{\boldsymbol{c}} = \langle f^v_{\texttt{VLM}}(\boldsymbol{x}), f^t_{\texttt{VLM}}(\boldsymbol{c}) \rangle. \tag{3.10}$$

To mitigate the modality gap in the space $\mathcal{Z}$, we introduce a unimodal text-to-text scoring. Denoting the centroid $\bar{\boldsymbol{d}}_{\boldsymbol{x}}$ of the retrieved captions, the text-based matching score $s^t_{\boldsymbol{c}}$ is:

$$\bar{\boldsymbol{d}}_{\boldsymbol{x}} = \frac{1}{K} \sum_{\boldsymbol{d} \in D_{\boldsymbol{x}}} f^t_{\texttt{VLM}}(\boldsymbol{d}), \tag{3.11}$$

$$s^t_{\boldsymbol{c}} = \langle \bar{\boldsymbol{d}}_{\boldsymbol{x}}, f^t_{\texttt{VLM}}(\boldsymbol{c}) \rangle. \tag{3.12}$$

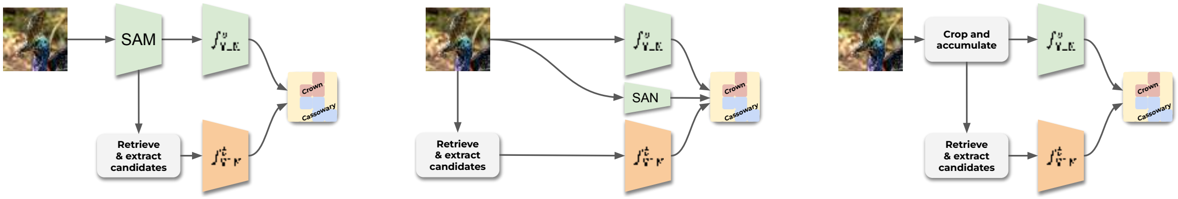
We obtain the final score $s_{\boldsymbol{c}}$ for each candidate $\boldsymbol{c}$ by merging the two scores, as:

$$s_{\boldsymbol{c}} = \alpha\ \sigma(s^v_{\boldsymbol{c}})\ +\ (1 - \alpha)\ \sigma(s^t_{\boldsymbol{c}}) \tag{3.13}$$

where $\sigma(\cdot)$ is the softmax operation on the two scores of each candidate class, and α is a hyperparameter. The output category is $f_{\texttt{CaSED}}(\boldsymbol{x}) = \arg\max_{\boldsymbol{c} \in C_{\boldsymbol{x}}} s_{\boldsymbol{c}}$. CaSED, is *training-free*, uses a pre-trained and frozen VLM, and is flexible for various architectures and databases.

CaSED for Vocabulary-free Semantic Segmentation

To extend CaSED for the task of semantic segmentation, we can follow three strategies. The first is to exploit an available class-agnostic segmentation model [117], extract



(a) SAM-based method (b) SAN-based method (c) DenseCaSED

Figure 3.7: We extend CaSED to Semantic Segmentation following three strategies: (a) a class-agnostic segmenter (SAM) segments all objects, then CaSED labels each mask independently; (b) CaSED provides candidate categories for the image that are fed as input to an open-vocabulary segmentation model (SAN); (c) DenseCaSED, where we directly accumulate visual features from multi-scale patches, and perform CaSED locally.

segmentation masks and then assigning them a label via CaSED, independently. The second does the opposite: the initial set of candidates generated by CaSED is the input to an open-vocabulary segmentation network [264]. While these strategies lead to good results but they require the additional computational cost of using another segmentation network. We thus propose a third approach which generates local visual representations directly from patches of the input image. In the following, we describe the three strategies.

Coupling CaSED with segmentation networks

Assigning semantics to class-agnostic masks with CaSED. Let us assume to have a class-agnostic segmentation network $f_{\texttt{SEG}}$ that, given as input an image $\boldsymbol{x}$ maps it to a set of k segmentation masks $M = \{\boldsymbol{m}_1, \cdots, \boldsymbol{m}_k\}$. Note that these masks have no semantic attached, and the number of masks k may be input dependent [117]. From the masks, we extract a set of image regions $R = \{\boldsymbol{r}_1, \cdots, \boldsymbol{r}_k\}$, *e.g.*, by cropping around the relative mask. We can then attach a semantic to each region using CaSED, propagating the prediction to the pixels of the relative mask. Given a pixel $x \in \boldsymbol{x}$ that assigned to mask $\boldsymbol{m}_i \in M$ by $f_{\texttt{SEG}}$, its semantic label is simply $f_{\texttt{CaSED}}(\boldsymbol{r}_i)$. The rationale behind this approach is that the semantic of a pixel is the one assigned by CaSED to an image extracted from the mask the pixels belong to.

Candidates generation with CaSED for Open-Vocabulary Segmentation. The main limitation of the previous approach is that extracting meaningful image regions from masks may require solutions that exploit VLM priors on object localization (*e.g.*, circle drawing [226]). To sidestep this problem, we can invert the pipeline by first obtaining candidate class-names using CaSED and then segmenting the image using an open-vocabulary segmentation model, *e.g.*, [263]. Specifically, an open-vocabulary segmentation model takes as input an image and a set of possible labels $\mathcal{Y} \subseteq \mathcal{S}$ and maps them to a segmentation mask, assigning pixels to elements of $\mathcal{Y}$, *i.e.*, $f_{\texttt{OV-SEG}} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Y}^N$. As we will show experimentally in section 3.2.4, if we do not take into account the challenges of VSS and set $\mathcal{Y} = \mathcal{S}$, we obtain poor segmentation results. This is mainly due to the extremely large cardinality of $\mathcal{S}$, requiring to distinguish local, fine-grained differences of potentially similar semantic concepts.

To overcome these challenges, we follow the rationale behind CaSED, restricting the search space by estimating a set of candidate classes. Thus, given an image $\boldsymbol{x}$, we define a set of candidates $C_{\boldsymbol{x}}$ by filtering a set of captions $D_{\boldsymbol{x}}$, with the latter obtained as in eq. (3.9). We then feed the set $C_{\boldsymbol{x}}$ as input to the open-vocabulary segmentation network and obtain the relative segmentation mask as $f_{\texttt{OV-SEG}}(\boldsymbol{x}, C_{\boldsymbol{x}})$.

DenseCaSED

The previous approaches rely on the presence of an additional (pretrained) semantic segmentation model, and the assumption that the module is not biased toward particular input distributions. Here we propose a different strategy, directly exploiting the available VLM.

The approach applies CaSED to local image representations, as done with the class-agnostic strategy previously described. As we do not have access to masks, we need to define local image regions that we can feed as input to the VLM. To obtain such representations, We divide the image in multiple grids, where each grid has n^2. For simplicity, we choose n to be powers of 2, *i.e.*, $n \in \{1, 2, 3\}$. We also replicate the grids by shifting them vertically and/or horizontally by a stride equal to half the size of a grid cell. This creates a hierarchy of patches where neighboring patches are likely to belong to the same super patch and therefore their representation (loosely) depends on each other. This helps in embedding contextual information in the aggregated pixel-level representation.

Formally, let us denote the visual representation of a pixel i in $\boldsymbol{x}$ as $\boldsymbol{l}_i$. Moreover, let us denote as $\{\boldsymbol{g}_i^1, \cdots, \boldsymbol{g}_i^N\}$ all patches that contain i. The local representation $\boldsymbol{l}_{i,j}$ is then

$$\boldsymbol{l}_i = \frac{1}{N} \sum_{q=1}^{N} f^v_{\texttt{VLM}}(\boldsymbol{g}_i^q). \tag{3.14}$$

Note that we divide the aggregated value for each pixel by the number of times it was forwarded within a cell. As these local representations are already encoded using a

VLM, we can retrieve the most relevant set of captions to a cell from a VLD D using the cosine similarity of the embeddings, as:

$$D_{\boldsymbol{l}_i} = \underset{\boldsymbol{d} \in D}{\text{top-k}} \; \langle \boldsymbol{l}_i, f^t_{\texttt{VLM}}(\boldsymbol{d}) \rangle, \tag{3.15}$$

For each cell in $\boldsymbol{g} \in G_g$, we follow the pipeline of CaSED, by (i) filtering $D_{\hat{\boldsymbol{l}}}$ to obtain the corresponding set of candidates $\mathcal{C}_{\boldsymbol{g}}$; (ii) computing the visual score as in eq. (3.10) but using $\hat{\boldsymbol{l}}$ as visual representation; (iii) compute the textual scoring as in eq. (3.12) with $D_{\hat{\boldsymbol{l}}}$ as set of captions; (iv) merging the two scores to compute the final multimodal one, as in eq. (3.13). The CaSED predictions on accumulated local visual features are then propagated to the whole cell, producing the final segmentation mask. We name this approach DenseCaSED.

Note that this approach is not only training-free but does not use any segmentation network, relying only on a contrastive-based VLM. Moreover, accumulating local cells representations across scales allows to model the context in which a cell appears while enforcing a consistent vocabulary across neighboring cells. This latter aspect is important in VSS, as modeling cells in isolation may lead to inconsistent choices of labels in the large search space (*e.g.*, “sofa” *vs* “couch”) leading to lower segmentation results.

3.2.4 Experimental results

Experimental protocol

Datasets. We experiment with three datasets: Pascal VOC [66] (VOC-20), PASCAL Context-59 [185] (CTX-59), and ADE20K-150 [305] (ADE-150). As opposed to [263, 262, 152], we do not use COCO Stuff [22], as it is used for training and all the considered baselines and methods are training-free.

Evaluation metrics. To address the openness of VSS, we extend two popular metrics for open-vocabulary semantic segmentation, namely Jaccard Index (JI) and Recall (R). We will refer to the metrics as Hard JI (HJI) and Hard Recall (HR). For their soft variants, we replace the binary “hard” values (*i.e.*, zero and one) with the semantic similarity between predicted and ground-truth word. The Soft Jaccard Index (SJI), directly accounts for this similarity at pixel-level. Formally, for a pixel $\boldsymbol{p}$ with label $\boldsymbol{y}$ and prediction $\hat{\boldsymbol{c}} = f(\boldsymbol{p})$, the textual similarity is computed as $\langle g(\hat{\boldsymbol{c}}), g(\boldsymbol{y}) \rangle$, where the function $g : \mathcal{T} \rightarrow \mathcal{Y}$ maps text to an embedding space $\mathcal{Y}$. As in [48], we use Sentence-BERT [209] as g. Similarly, Soft Recall (SR) expands the recall metric with the semantic proximity of the predicted and ground-truth classes in the image.

Furthermore, we introduce two variants of the JI, *i.e.*, Nearest Jaccard Index (NJI) and Overlap Jaccard Index (OJI) to account for cases where proposed words may not perfectly align with the annotations due to linguistic ambiguities or to specificity of the proposed segmentation masks, *i.e.*, part *vs* whole cases (*e.g.*, predicting "head" and "shirt" on two parts of a "person"). We propose to map predicted names to ground-truth ones, evaluating the traditional JI on the projected predicted mask. More formally, given the predicted segmentation mask $\boldsymbol{C}$ and its ground-truth mask $\boldsymbol{Y}$, we extract the lists of predicted names $L^{\boldsymbol{C}} \in \boldsymbol{C}$ and the ground-truth names $L^{\boldsymbol{Y}} \in \boldsymbol{Y}$. We then create a mapping $M : \mathcal{L}^{\boldsymbol{C}} \rightarrow \mathcal{L}^{\boldsymbol{Y}}$, so that each predicted word is mapped to one ground-truth word. The criteria behind this mapping is the main difference between NJI and OJI. In the former, we use textual similarity between predictions and the list of annotated words. In the latter, we directly evaluate the co-occurrence of predictions and annotations in the pixel space.

Baselines. We consider two groups of baseline methods for comparison. The first exploits SAM [117] to first extract regions and then propose region-based candidate names via retrieval or generation with, *e.g.*, English Words or BLIP-2. The second uses a open-vocabulary semantic segmentation model in the absence of the pre-defined list of class names for the dataset, requiring the ad-hoc generation with traditional vocabulary-free methods. In this context, we employ SAN [263] as it offers a minor variation from the CLIP architecture, incorporating only an auxiliary network to address the task on the pre-trained backbone. For both groups, we report results with the same set of baselines of VIC, without the captioning baselines. We also report results when CaSED is used for vocabulary generation. For all main experiments, we use CLIP with the ViT-L/14 backbone.

Implementation details. We conduct our experiments on NVIDIA A6000 GPUs, using mixed-precision for efficiency. We use a subset of PMD [230] as our database, which includes five of its largest datasets: Conceptual Captions (CC3M) [222], Conceptual Captions 12M (CC12M) [26], Wikipedia Image Text (WIT) [235], Redcaps [62], and the portion of YFCC100M* [242] curated for PMD. For retrieval, we embed the database with the CLIP text encoder $f^t_{\texttt{VLM}}$, using a fast indexing technique, *i.e.*, FAISS [109]. The hyperparameter α in eq. (3.13) and the number of retrieved captions K are the same of CaSED. DenseCaSED introduces a single hyperparameter, *i.e.*, the grid sizes to crop the input image. We empirically set $N = \{2, 4, 8\}$ to have a final dense pixel map of 16×16, for a total of 256 sub-regions for each input image. For computational efficiency, we use FastSAM [299] in place of SAM for segmentation.

	Method	**VOC-20**					
		Jaccard Index				**Recall**	
		HJI	NJI	OJI	SJI	HR	SR
SAM	English words	4.2	12.1	48.9	11.1	5.4	30.4
	WordNet	4.9	14.5	48.2	12.5	6.1	35.9
	BLIP-2 (ViT-L)	15.8	19.2	31.7	17.2	33.4	60.6
	BLIP-2 (ViT-g)	15.1	17.5	27.2	16.4	33.1	61.7
	LLaVA 1.5 (7B)	18.3	18.9	29.3	17.5	**41.6**	**63.6**
SAN	English words	5.8	18.6	48.4	13.8	7.2	38.1
	WordNet	7.2	19.3	46.4	15.7	8.7	45.3
	SAM + CaSED	13.7	17.5	44.6	15.5	20.1	48.0
	SAN + CaSED	**26.9**	**30.2**	**53.2**	**20.8**	34.2	61.8
	DenseCaSED	20.5	20.3	32.2	18.5	31.4	61.8

Table 3.14: Semantic segmentation on PascalVOC-20. Green are our methods.

Quantitative results.

We report results on table 3.14, table 3.15, and table 3.16. Our experiments consistently demonstrate that English Words and WordNet are sub-optimal for the task, independently of whether a method first segment the image into regions of interest and then classify (*i.e.*, SAM-based methods), or first generates candidate names and then segment (*i.e.*, SAN-based methods). With SAM, the best approach exploits LLaVA 1.5 (7B) followed by BLIP-2 (both ViT-L and ViT-g), generally achieving comparable results across the six metrics and three datasets, with, *e.g.*, 12.7 and 12.3 on NJI, and 50.8 and 47.3 on SR, respectively. The strongest method consists of using SAN with CaSED to generate a custom list of candidates for each image and then segment with the open-vocabulary segmentation model. When compared with more naive approaches, *i.e.*, English Words or WordNet for retrieval, SAN with CaSED improves by, *e.g.*, about +12.9 on HJI and +17.0 on HR. SAN with CaSED and the best SAM-based approaches generally performs comparably on recall metrics, whether hard or soft. Notably, DenseCaSED, despite not including any component pre-trained for semantic segmentation, achieves and sometimes surpasses all the SAM-based approaches, even achieving the highest scores on hard and soft recall, *i.e.*, +0.9% and +1.4% against SAN with CaSED. Due to the coarse nature of the segmentation mask, however, DenseCaSED falls behind against methods trained for open-vocabulary semantic segmentation, while keeping an edge against SAM-based methods. Specifically, it achieves −2.3 and −1.0 on HJI and SJI against SAN with CaSED, but improves on average by +4.4 and +1.8 against SAM

	Method	**CTX-59**					
		Jaccard Index				**Recall**	
		HJI	NJI	OJI	SJI	HR	SR
SAM	English words	1.0	6.4	**40.2**	8.9	2.5	27.0
	WordNet	1.0	8.7	39.6	9.6	2.5	28.9
	BLIP-2 (ViT-L)	7.0	10.5	28.4	12.1	16.1	42.7
	BLIP-2 (ViT-g)	6.5	10.3	27.4	12.0	15.5	44.4
	LLaVA 1.5 (7B)	9.1	12.0	29.7	12.9	**21.4**	48.3
SAN	English words	1.2	8.0	34.9	10.3	2.7	30.7
	WordNet	2.0	9.4	33.0	11.2	3.6	33.6
	SAM + CaSED	7.5	11.0	38.2	11.5	11.2	39.4
	SAN + CaSED	**15.5**	**16.2**	38.1	**14.7**	20.8	46.9
	DenseCaSED	13.4	13.1	32.6	13.9	20.8	**48.6**

Table 3.15: Segmentation on PASCAL Context-59. Green highlights our methods.

with LLaVA (7B) or BLIP-2.

Qualitative results.

We report some qualitative results of CaSED for semantic segmentation, taking the best models in each group: of SAM with LLaVA 1.5 (7B), SAN with WordNet as vocabulary, and our localizer-free model for VSS, DenseCaSED. We report some output examples in fig. 3.8, fig. 3.9, and fig. 3.10. From a segmentation perspective, it is notable how SAM achieves the best segmentation masks, despite missing some more local objects, *e.g.*, it misses all the objects except the cat in the first example of fig. 3.9, or the horse in the first example of fig. 3.10. Moreover, it sometimes misclassifies one object for another (*e.g.*, it considers the table as a "car" in the second example of fig. 3.10, or the aisle in the airplane as "staircase" in the second example of fig. 3.8, or assigns the label "tree" to the sky and the car in fig. 3.9). In some other cases, the most correct label for a region is assigned to a near segmentation mask, for instance in the second examples in fig. 3.10 where it assigns "woman" to the hair and the wall behind the person, or "bottle" to the arm holding the bottle in the same example. We highlight that, nevertheless, SAM + LLaVA 1.5 (7B) is the model with the highest number of parameters overall, using two large-scale foundation models to address VSS.

Differently from SAM, SAN tends to segment whole objects instead of parts, *e.g.*, first and second example in fig. 3.8, where it divides the image into foreground and background, or completely merge floor and ceiling of the airplane. Moreover, given

	Method	ADE-150					
		Jaccard Index				**Recall**	
		HJI	NJI	OJI	SJI	HR	SR
SAM	English words	2.2	4.5	**31.2**	6.3	3.3	27.4
	WordNet	2.3	6.3	30.9	6.4	3.5	27.45
	BLIP-2 (ViT-L)	5.0	7.7	24.7	7.7	10.5	36.5
	BLIP-2 (ViT-g)	4.9	8.2	23.9	7.7	10.9	37.8
	LLaVA 1.5 (7B)	6.2	7.1	24.2	7.8	14.4	40.4
SAN	English words	2.2	4.5	19.2	7.2	3.6	30.5
	WordNet	2.8	5.1	19.1	7.5	4.8	32.9
	SAM + CaSED	6.1	7.7	29.4	7.4	10.0	35.2
	SAN + CaSED	7.2	7.5	20.8	8.7	11.3	40.9
	DenseCaSED	**8.6**	**9.1**	24.1	**8.8**	**16.8**	**43.6**

Table 3.16: Semantic segmentation on ADE20K-150. Green are our methods.

the labels selected by SAN, it looks like the model tend to assign the most out-of-context words compared with SAM with LLaVA 1.5 (7B) and DenseCaSED. As an example, SAN assigns the word "Erin" to the person in the second example of fig. 3.10, or "Cumberland Gap" to the landline in the second example in fig. 3.9. Other examples include "pantile" or "westbound" assigned to the sky in the first example of fig. 3.8 and in the second example of fig. 3.9. Some of these issues are fixed when using CaSED to generate the vocabulary, as shown by the better quantitative results achieved in Tab. 4-6 by SAN + CaSED. Finally, compared to the SAM segmentation masks, the outputs presents slightly more noise on the edges, where regions are not perfectly segmented but are more approximate.

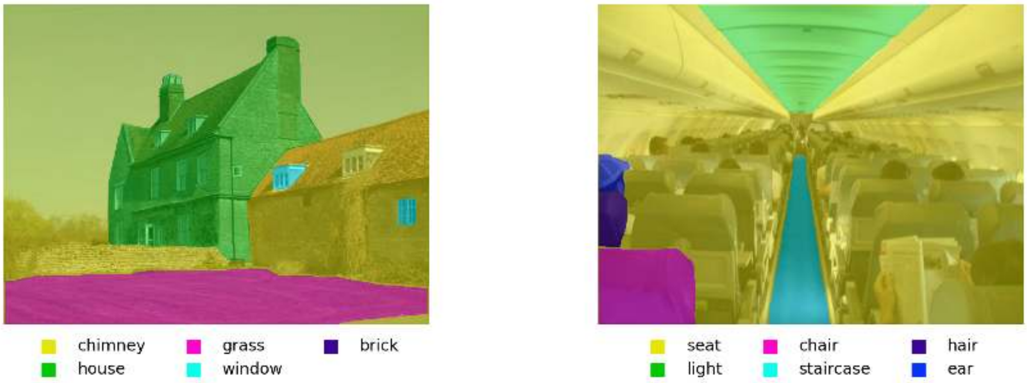
Compared to the previous approaches, our method shows more coarse segmentation masks mainly due to the lack of any segmentation model to generate them. This is noticeable in, *e.g.*, the first example of fig. 3.8, where the chimney and the sky have no clean separation, or in the second example of fig. 3.9, where the sky is not completely segmented as a single object. Moreover, it tends to propose larger amount of regions, *e.g.*, first example of fig. 3.9, where the cat is separated into "cat", "ear", and "whisker", or in the second example of fig. 3.10, where the woman is segmented as "hair" and "shirt". Note, however, that DenseCaSED is the only method to correctly classify and segment subtle details like the "cup" in the left part of the second example of fig. 3.10, or the horse and grass in the first example of the same figure. Despite the coarse segmentation masks, our model provides more contexualised labels, *e.g.*, "plane", "ceiling", "airline", "aisle", and "aircraft" for the airplane aisle in the second example of fig. 3.8 (SAM and SAN selects labels such as "chair", "air travel", and "hair"), or "cat", "whisker", and "ear", in the first example of fig. 3.9 (SAN selects "tabby", "gitana", "Freya", and "soporific" as labels).

To conclude, all methods show their peculiarities, with SAM presenting the cleanest segmentation masks but sometimes undersegmenting objects, SAN having the most efficient implementation but showing sub-optimal performance with vocabularies out-of-distribution (*i.e.*, outside of the common words used for semantic segmentation, as also shown by its weakness against "distractor" words), and DenseCaSED recognising also smaller objects but resulting in the most coarse segmentation masks. These results confirm the challenges of VSS, and the need of developing methods tailored to this task.

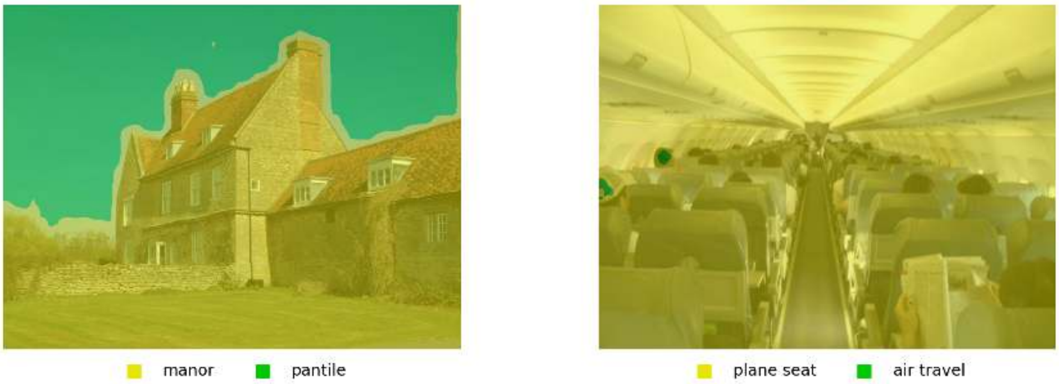
3.2.5 Related work

Vision Language Models. The recent surge in models that map image-text pairs into a shared representation space has been largely driven by the availability of large-scale datasets [218, 217, 230]. These models [110, 84, 61, 206, 107, 143, 74] employ modality-specific encoders and a contrastive objective to align the output representations of the two modalities. A prime example of this approach is CLIP [206], which has demonstrated impressive results in zero-shot classification tasks. Further enhancements to CLIP have been proposed, including the integration of cross-modal attention [143], multi-object representation alignment [289], learning from weak-supervision [254], and leveraging unaligned data [230]. A separate stream of research has focused on enhancing vision-language pre-training for more intricate vision-language tasks, such as image captioning and visual question answering (VQA) [276, 99, 142, 3, 160]. Within this domain, BLIP [142] uses web data and generated captions to guide the pre-training of a multimodal architecture, surpassing existing VLMs in both captioning and VQA tasks, and LLaVA [160] aligns and CLIP vision encoder and LLAMA large-language model [244] to reason on visual inputs.

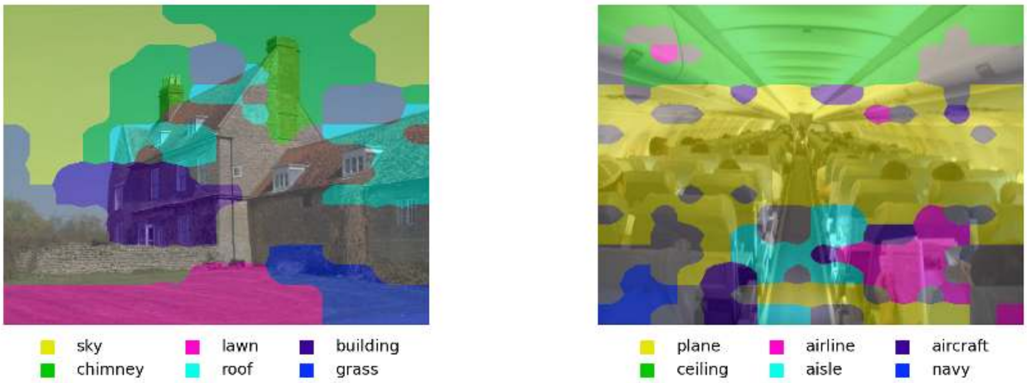
Here, we question a core premise of zero-shot classification with VLMs: the prior knowledge of the target classes. We present VIC, that bypasses this assumption, performing classification within an open-ended, language-induced space of semantic categories. In this setting, even advanced methods like BLIP-2 [140] struggle, whereas caption databases offer valuable priors for deducing the semantic class of an image. It is important to distinguish VIC from open-vocabulary recognition (*e.g.*, [286, 82]), as the latter still operates assuming that the list of target classes is known and accessible



(a) SAM + LLaVA 1.5 (7B)

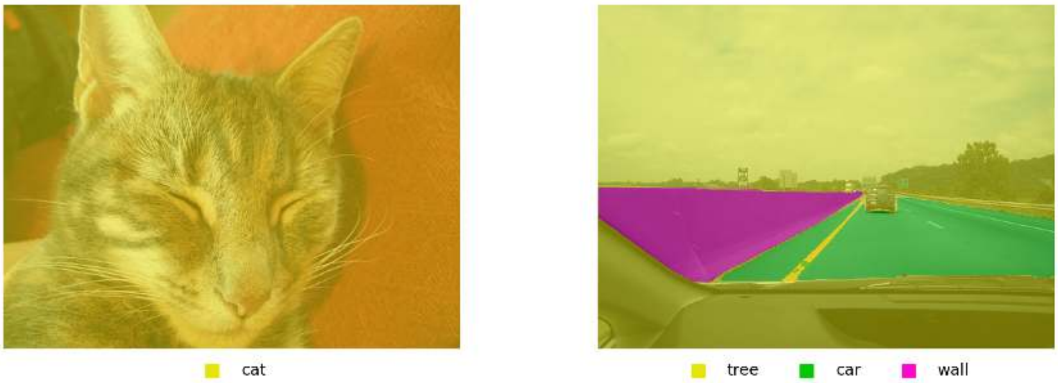


(b) SAN + WordNet

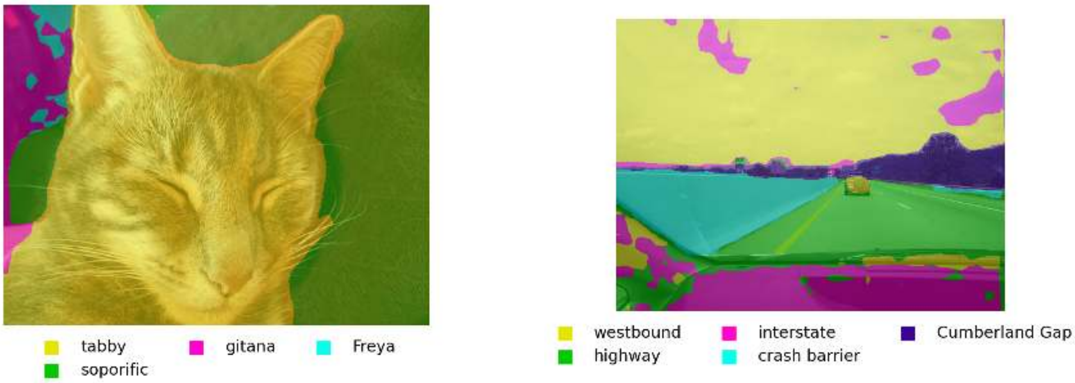


(c) DenseCaSED

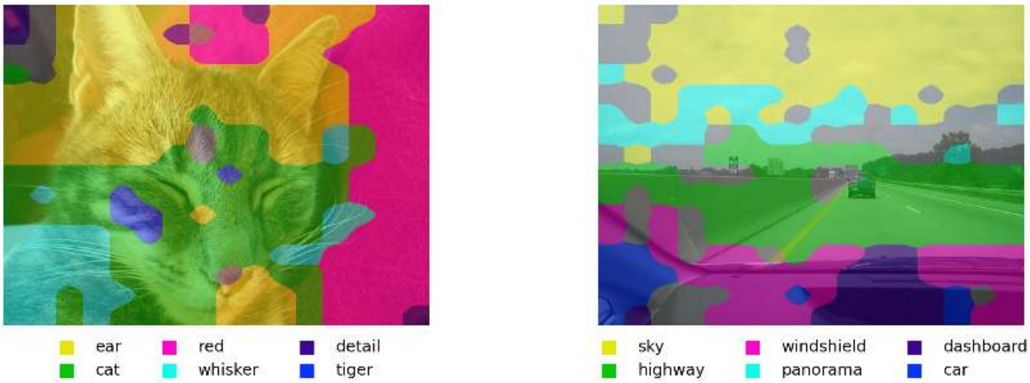
Figure 3.8: Qualitative results of Vocabulary-free Semantic Segmentation methods on ADE20K-150. For visualisation purposes, we visualise only the top-6 most prominent regions.



(a) SAM + LLaVA 1.5 (7B)

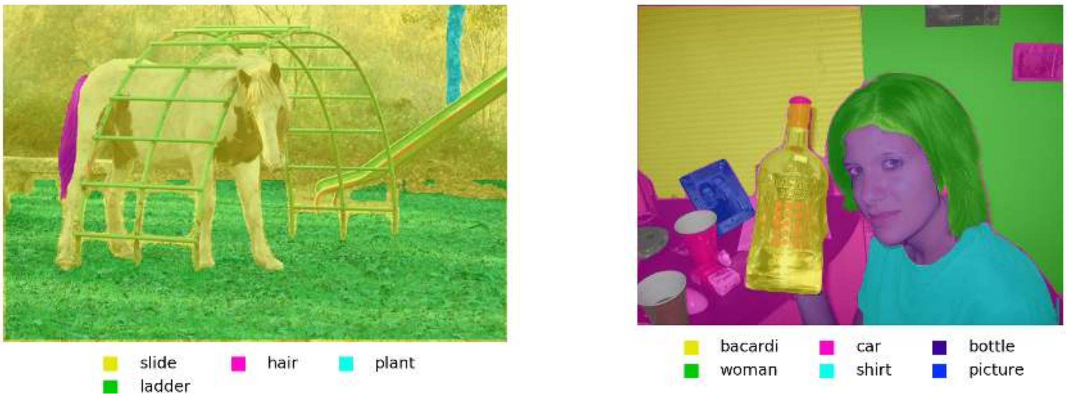


(b) SAN + WordNet

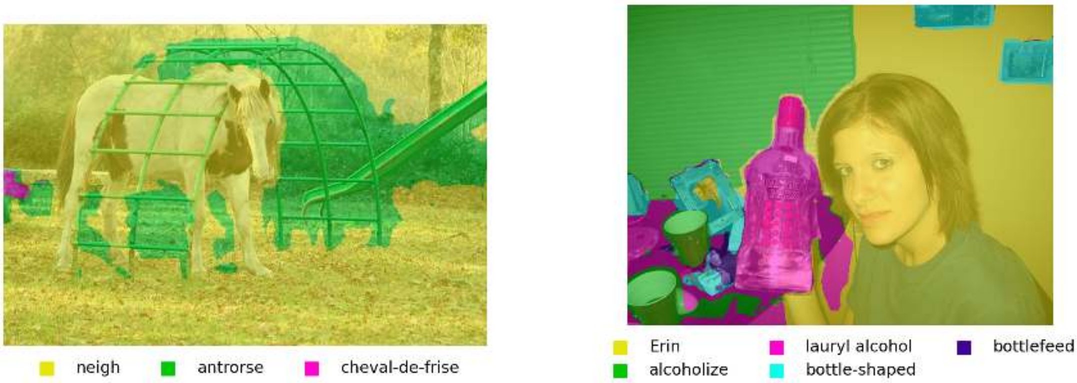


(c) DenseCaSED

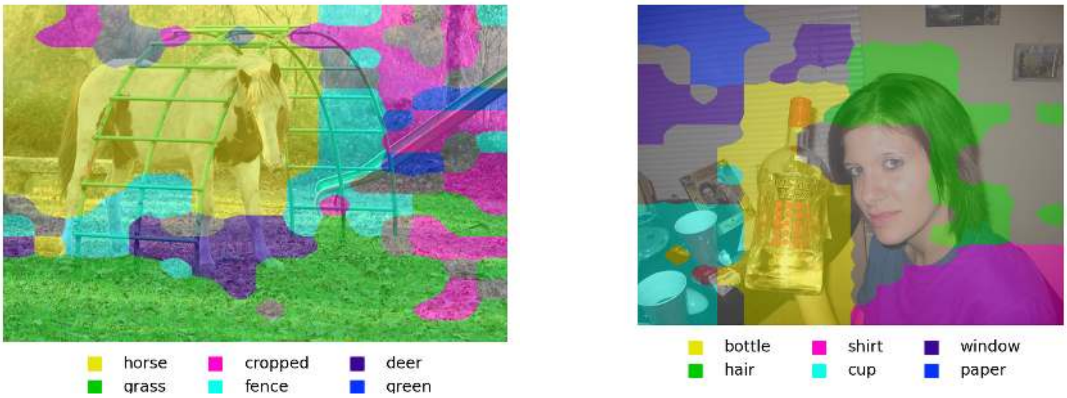
Figure 3.9: Qualitative results of Vocabulary-free Semantic Segmentation methods on PASCAL Context-59. For visualisation purposes, we visualise only the top-6 most prominent regions.



(a) SAM + LLaVA 1.5 (7B)



(b) SAN + WordNet



(c) DenseCaSED

Figure 3.10: Qualitative results of Vocabulary-free Semantic Segmentation methods on PascalVOC-20. For visualisation purposes, we visualise only the top-6 most prominent regions.

to the model during inference.

Retrieval augmented models. In the field of natural language processing, a number of studies have demonstrated the advantages of retrieving information from external databases to enhance the performance of large language models [88, 134, 17]. This approach has also found applications in computer vision, particularly in addressing the issue of class imbalance. For instance, some works have focused on long-tail recognition by learning to retrieve training samples [165] or image-text pairs from an external database [166]. Another study, [245], retrieves images from a specific dataset to learn fine-grained visual representations. More recently, the concept of retrieval-augmentation has been expanded to various types of sources for visual question answering [100], as well as to condition the generative process in diffusion models [16], and even image captioning [208].

Our work shares similarities with [166] in that we also utilize an external database. However, unlike [166] which assumes a pre-defined set of classes (and data) available for training a retrieval module, our approach does not make this assumption due to the vast semantic space of VIC. In our method, CaSED, we use retrieval to first generate a set of candidate classes, and then to perform the final class prediction. Furthermore, we assume the database to contain only captions, and not necessarily paired image-text data, thus making our approach less memory-intensive.

Semantic segmentation. The earliest works tackling semantic segmentation with an open vocabulary learns a joint embedding space between pixels and class names [257, 21, 35, 298]. After the surge in image-text models such as CLIP [206], the paradigm shifted towards exploiting such web-scale pre-trained models as priors to tackle the problem [82, 151, 264, 277]. All prior works on the task assume a fixed pre-defined list of candidate names to address the task of semantic segmentation. Such simplification, however, is unrealistic for real-world applications, where, *e.g.*, a pre-defined list of class names is restrictive and not exhaustive for most scenarios. Different from previous approaches, we aim to tackle a more challenging setup where this list of class names is unavailable and must be inferred from the input.

Recently, we formalized a similar task for image classification [48] and our objective is to expand such scenario to dense classification tasks. The most similar work to our proposed task of Vocabulary-free Semantic Segmentation is zero-guidance segmentation [210]. Differently from them, we formalize the Vocabulary-free Semantic Segmentation, strengthening the evaluation protocol, proposing principled metrics for the vocabulary-free scenario, and reusing traditional semantic segmentation benchmarks to assess the performance of multiple baselines methods.

3.3 Classification with Large Multimodal Models

3.3.1 The problem

Image classification aims to assign a label to an image. This widely studied task relies on a key assumption: the categories are fixed and known in advance, a setting known as the *closed world*. However, the latter is often restrictive in real-world applications where new categories can emerge, requiring to expand the label set [13], recognize unseen concepts [81], or both [14]. Despite its limitations, this assumption has historically been useful, enabling supervised training and straightforward evaluation on labeled data. With the rise of Large Multimodal Models (LMMs) [162, 141, 11] processing images and text, this constraint is no longer necessary. Instead of choosing from a fixed list, LMMs can answer open-ended prompts such as *"What is the object in the image?"*, recognizing virtually any semantic concept. From this perspective, closed-world classification is an artificial limitation that restricts a model's expressive capabilities rather than reflecting its true potential.

While some studies have explored classification with LMMs, they have either focused on the closed-world setting [163] or relied on limited metrics to assess performance: checking whether the ground truth label appears in the model's prediction [296]. However, this metric provides a limited view of classification performance. It fails to account for alternative correct answers (*e.g.*, *sofa* instead of *couch*), while also overlooking real mistakes (*e.g.*, confusing *can* with *trash can*). Evaluating models in the open world presents additional challenges, as predictions may differ in granularity (*e.g.*, *dog vs pug*), or conflict with annotation ambiguities (*e.g.*, *bedroom vs bed*). These issues highlight the need for a more comprehensive evaluation framework to assess the open-world capabilities of LMMs.

In this work, we address this gap by formalizing the Open-World (OW) classification task and introducing four complementary metrics: (i) text inclusion [296], evaluating string matching, (ii) Llama inclusion which leverages Llama [86], to distinguish good and bad responses, as in LLM-as-a-judge [302], (iii) semantic similarity [48] between text embeddings of predictions and ground truth, and (iv) concept similarity, doing the same at the level of sentence parts. Using these metrics, we evaluate 13 models across 10 benchmarks spanning different levels of granularity, from prototypical, coarse categories (*e.g.*, Caltech101 [71]) to fine-grained (*e.g.*, Flowers102 [189]), very fine-grained (*e.g.*, Stanford Cars [122]), and non-prototypical ones (*e.g.*, DTD [46]). Our results (aggregated in fig. 3.11) show that these models often predict semantically related con-

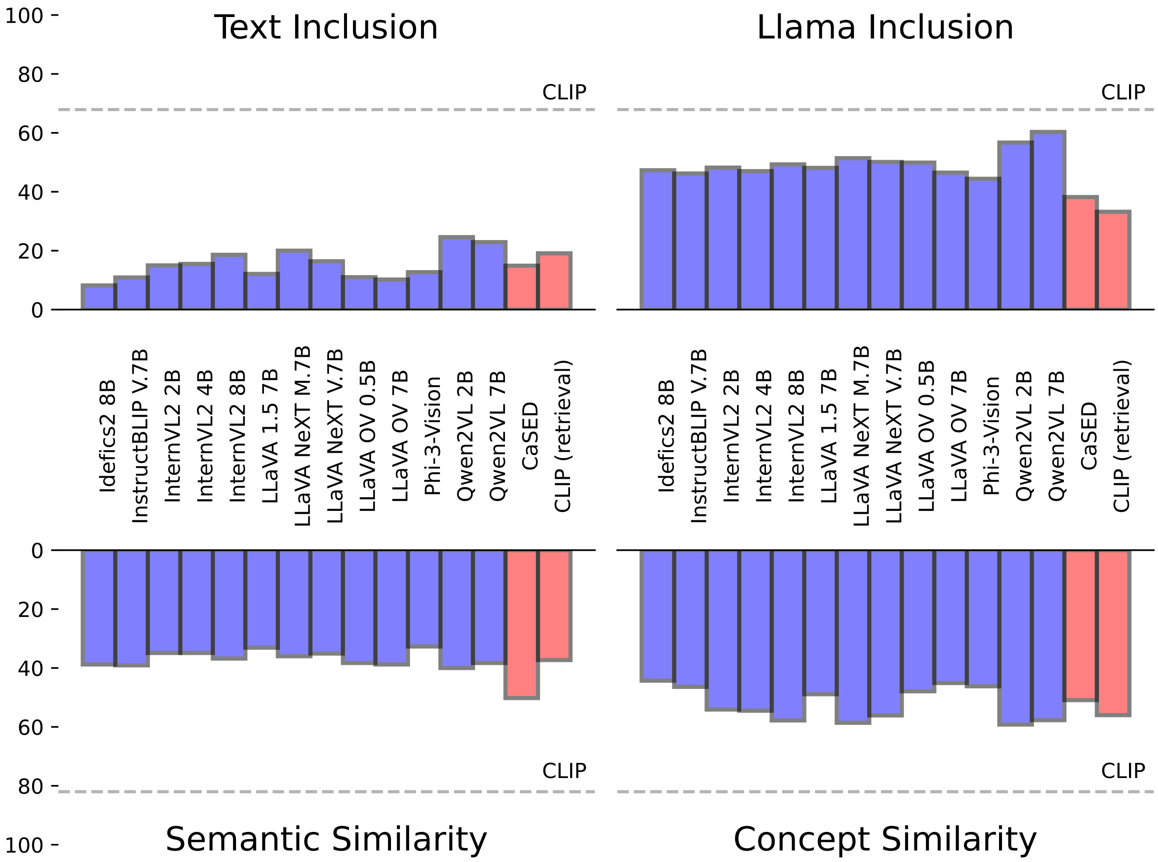


Figure 3.11: We extensively test 13 Large Multimodal Models (LMMs) for Open-World (OW) classification on 10 datasets using four evaluation metrics. We show that LMMs outperform contrastive-based approaches in OW (CaSED [48], and CLIP [205] with image-to-text retrieval) but still lag behind closed-world models with fixed categories (CLIP [205], dashed line).

cepts (even better than previous, contrastive-based alternatives [48]), demystifying the skepticism against LMMs on OW classification. However, LMMs also make notable errors, being far from closed-world baselines.

While the nature of the problem makes analyzing the severity of the mistakes hard, we find that different behaviors on different metrics can pinpoint sources of errors. In particular, mismatches in concept similarity and Llama inclusion uncover errors in the granularity of the predictions (*i.e.*, correct but generic) or very similar categories easy to confuse (*i.e.*, wrong but specific). We show how the former can be addressed via tailored prompting, while the latter via implementing reasoning strategies. Finally, we further analyze cases where the metrics identify a mistake and, using tagging models [101], check if predictions are incorrect only due to the single-label nature of the datasets.

Contributions. To summarize, our contributions are:

- We formalize a comprehensive evaluation protocol for the task of *Open-World classification* with LMMs, using 4 different metrics capturing both semantic and text alignment of predictions with the ground truth.

- We perform the first, large-scale assessment of LMMs on this task, using 13 models on 10 benchmarks, showing promising results yet multiple challenging cases.

- By combining the different metrics, we investigate the root of the models' mistakes, identifying various issues (*e.g.*, wrong granularity, fine-grained discrimination, labeling ambiguities) and showing how changes in the models (*e.g.*, prompts, reasoning) can reduce them.

- We use these results to draw conclusions on the source of errors that future research should account for when using these models, releasing our evaluation suite to encourage future research efforts on addressing them.

3.3.2 The proposed setting

In this section, we first formalize the setting of OW classification with LMMs, clarifying its goal and terminology *w.r.t.* related works (section 3.3.4). We then discuss how to evaluate performance in this setting, describing the different metrics and what they capture (section 3.3.2). Last, we provide details on the datasets and models considered in our analyses (section 3.3.2) before showing their results (section 3.3.2).

Preliminaries

Classification with LMMs. Let us define an LMM as a function $f_{\texttt{LMM}}$ generating a text output y in the space $\mathcal{T}$ given an image x in the space $\mathcal{X}$ and a text query $q \in \mathcal{T}$, *i.e.*, $f_{\texttt{LMM}} : \mathcal{X} \times \mathcal{T} \rightarrow \mathcal{T}$. To perform classification with LMMs, the query q contains a prompt of the type *"What type of object is in this image?"* and we expect the output y to be a semantic class $\mathcal{Y} \subset \mathcal{T}$. In the case of closed-world classification, we have a predefined list $\mathcal{C}$ of classes and we modify q by specifying the set $\mathcal{C}$ (*e.g.*, via a multi-choice question). In OW we let the LMM predict naturally on its original output space $\mathcal{T}$, without any constraint. As a consequence, the model can pick from the set $\mathcal{Y}$ of *all* possible semantic concepts, with $\mathcal{C} \subset \mathcal{Y}$ and $|\mathcal{C}| \ll |\mathcal{Y}|$.

Relationships with prior problem definitions. While we followed [296] and used *open-world* to define this setting, the term can be ambiguous. The traditional definition of OW recognition [14] refers to a different problem, where a model trained to recognize $\mathcal{C}$ classes should do so whether an instance belongs to an unknown one $u \notin \mathcal{C}$ and learn to recognize u. Other works refer to this task as *vocabulary-free* classification [48] due to the absence of a predefined vocabulary, *open-ended* recognition [278] due to the lack of constraints, or avoided any specific terminology in the context of multi-label recognition [280]. While these different definitions closely relate to each other, we follow [296], clarifying that OW here refers only to the lack of constraints in the output space of the LMMs.

Metrics

Evaluating open-world recognition with LMMs is challenging as, even if we have a ground truth, we have no guarantee that the model will output the same name when correct (*e.g.*, *sofa vs couch*), especially as the model may produce an undesired wordy output (*e.g.*, *the object in the image is a sofa*). These potential variations require specific evaluation criteria, accounting for different types of (mis)alignment between the prediction and the ground truth. Below, we describe the four metrics we consider for this task.

Text inclusion (TI). This metric, adopted in [296], refers to whether the ground truth is contained in the model's prediction. Specifically, let us define as y the ground truth and as $\hat{y}$ the model's prediction. Text inclusion score is defined as:

$$\mathrm{TI}(y, pr) = \begin{cases} 1 & \text{if}\, y \subseteq \hat{y}, \\ 0 & \text{otherwise} \end{cases} \tag{3.16}$$

where, in this context, $\subseteq$ refers to string inclusion. This metric assesses whether the predictions strictly adhere to the ground truth label but over-penalizes whether the two are semantically coherent (*e.g.*, the prediction *labrador* would be considered wrong for the label *labrador dog*).

Llama inclusion (LI). This metric evaluates whether the prediction aligns with the ground truth label based on the internal knowledge of a Large Language Model (LLM). Specifically, we employ Llama 3.2 3B [244] with the following prompt:

Llama inclusion instruction

> You are a model that determines whether an answer is a good reply to a question given also its target value.
>
> This is the question: What type of object is in this image?
> This is the answer: %s
> This is the target value: %s
>
> If the answer describes the target, reply positively. If the answer includes the target value or a synonym of it, reply positively. If the target is generic but it is related to the answer, reply positively. Reply only with “1” if yes, or “0” if no.

The score is 0 or 1, depending on the LLM’s answer. This is similar to methods that use LLM/LMM-as-a-judge [302, 27], but is specifically adapted to OW classification.

Semantic similarity (SS). Unlike previous metrics that assess alignment with the ground truth in a binary manner, SS captures the degree of semantic similarity on a continuous scale between 0 and 1. To achieve this, we employ a semantic similarity metric. Following [48], we define similarity as $\langle g_{\texttt{emb}}(\hat{y}), g_{\texttt{emb}}(y)\rangle$, where $g_{\texttt{emb}}$ is a text embedding function, and $\langle\cdot,\cdot\rangle$ denotes cosine similarity. As in [48], we use Sentence-BERT [209] for computing embeddings.

Concept similarity (CS). By considering the prediction as a whole, the semantic similarity previously defined ignores whether parts of the sentence (*e.g.*, *elephant*) are closer to the ground truth (*e.g.*, *animal*) than the sentence as a whole (*e.g.*, *a photo of an elephant in the room*). To address this, we consider CS as an additional metric, defining it as:

$$\max_{p\in\texttt{split}(\hat{y})} \langle g_{\texttt{emb}}(p), g_{\texttt{emb}}(y)\rangle \tag{3.17}$$

where `split` is a sentence splitting procedure that, in our case, is implemented via

	Prototypical				**Non-prototypical**			
Model	**TI**	**LI**	**SS**	**CS**	**TI**	**LI**	**SS**	**CS**
Idefics2 [132] 8B	30.8	52.7	54.5	63.1	3.7	27.9	35.4	41.3
InstructBLIP [58] Vicuna 7B	29.7	56.3	56.8	64.0	6.0	27.1	37.0	42.0
InternVL2 [39, 38] 2B	36.9	69.9	46.9	70.4	10.2	45.2	31.6	53.4
InternVL2 [39, 38] 4B	36.3	68.5	46.5	70.8	10.1	42.1	30.8	53.1
InternVL2 [39, 38] 8B	40.6	74.4	48.2	74.0	11.0	46.2	31.9	53.9
LLaVA-1.5 [162] 7B	34.6	63.1	45.3	65.8	8.6	44.3	33.0	49.5
LLaVA-NeXT [135] (Mistral 7B)	41.7	73.9	45.9	74.3	**11.3**	**46.8**	31.2	**54.4**
LLaVA-NeXT [135] (Vicuna 7B)	39.5	72.8	46.2	73.2	10.6	45.9	31.1	54.2
LLaVA-OV [136] (Qwen2 0.5B)	34.4	64.4	54.0	67.3	7.3	37.0	32.8	47.0
LLaVA-OV [136] (Qwen2 7B)	30.8	53.2	56.1	62.0	7.2	28.1	31.6	43.8
Phi-3-Vision [2]	34.1	60.1	47.7	65.1	6.0	28.7	26.0	39.5
Qwen2VL [253] 2B	44.9	77.8	52.2	74.7	7.8	34.3	27.7	42.7
Qwen2VL [253] 7B	**46.4**	**78.7**	51.9	**76.0**	10.3	42.6	30.8	49.8
Open-world baselines								
CaSED [48]	24.5	46.3	**58.9**	59.8	5.4	18.6	**41.8**	42.4
CLIP retrieval	28.6	42.9	40.2	60.6	7.5	24.6	28.1	43.4
Closed-world baselines								
CLIP [205]	76.4		91.5		56.0		73.6	
SigLIP [290]	81.8		90.5		61.7		76.1	

	Fine-grained				**Very Fine-grained**			
Model	**TI**	**LI**	**SS**	**CS**	**TI**	**LI**	**SS**	**CS**
Idefics2 [132] 8B	3.0	49.9	38.0	41.7	0.0	67.0	29.6	33.6
InstructBLIP [58] Vicuna 7B	10.4	48.8	35.6	47.2	0.0	61.0	30.0	34.3
InternVL2 [39, 38] 2B	14.9	47.0	31.6	50.7	0.7	32.9	33.1	43.9
InternVL2 [39, 38] 4B	16.2	44.4	32.0	52.0	1.7	36.8	33.8	44.2
InternVL2 [39, 38] 8B	22.3	46.7	34.8	56.7	2.3	32.5	36.0	49.4
LLaVA-1.5 [162] 7B	8.4	46.5	28.2	44.8	0.0	41.0	28.6	37.6
LLaVA-NeXT [135] (Mistral 7B)	26.8	43.7	35.3	60.1	1.4	47.2	34.2	46.9
LLaVA-NeXT [135] (Vicuna 7B)	16.9	44.5	32.2	53.2	1.3	42.2	34.5	46.1
LLaVA-OV [136] (Qwen2 0.5B)	6.0	42.7	38.5	43.3	0.6	65.6	30.5	37.1
LLaVA-OV [136] (Qwen2 7B)	6.4	40.4	39.0	43.8	0.0	76.7	31.9	32.4
Phi-3-Vision [2]	13.4	49.1	31.8	47.2	0.2	45.0	28.9	36.0
Qwen2VL [253] 2B	**35.7**	62.5	40.7	63.4	**12.9**	60.7	45.1	**62.3**
Qwen2VL [253] 7B	34.6	**64.0**	39.2	62.9	0.8	**63.0**	34.5	43.4
Open-world baselines								
CaSED [48]	27.4	46.6	**60.7**	61.7	0.7	47.1	38.5	38.5
CLIP retrieval	32.4	45.4	42.9	**65.4**	7.0	18.1	**39.7**	56.1
Closed-world baselines								
CLIP [205]	85.0		89.6		51.7		73.6	
SigLIP [290]	92.6		95.1		69.2		89.1	

Table 3.17: OW results averaged on the grouped datasets. TI stands for text inclusion, LI for Llama inclusion, SS for semantic similarity, and CS for concept similarity. Higher is better, **bold** indicates best.

Dataset	**Images**	**Classes**
Caltech101 [71] (C101)	2,465	100
DTD [46]	1,692	47
EuroSAT [93] (ESAT)	8,100	10
FGVCAircraft [177] (FGVC)	3,333	100
Flowers102 [189] (FLWR)	2,463	102
Food101 [19] (FOOD)	30,300	101
OxfordPets [195] (PETS)	3,669	37
Stanford Cars [122] (CARS)	8,041	196
SUN397 [259] (S397)	19,850	397
UCF101 [234] (U101)	3,783	101

Table 3.18: Summary details of the datasets used in our analyses.

spaCy [6].

Dataset and models

Datasets. Following previous works [48, 228, 306], we analyze four different challenges: coarse-grained (or prototypical), non-prototypical, fine-grained, and very fine-grained classification. For the **prototypical** classification, we include standard benchmarks such as Caltech101 [71] for objects and SUN397 [259] for places. The **non-prototypical** set comprises datasets that either lack nouns or involve non-standard domains. This includes DTD [46] (textures), UCF101 [234] (actions), and EuroSAT [93] (satellite images). The **fine-grained** set consists of datasets where classes belong to a shared superclass and/or are challenging to distinguish. These include Flowers102 [189] (flowers), Food101 [19] (food), and OxfordPets [195] (animals). Finally, the **very fine-grained** set comprises datasets where categories are not only within the same subclass but also highly difficult to differentiate. This includes StanfordCars [122], where labels specify car brands, models, and years of production, and FGVCAircraft [177], which categorizes aircraft models. We report a summary table of these datasets in table 3.18; for all these datasets, we used the same training and test splits used in previous works [48].

Models. We perform our evaluation considering 13 state-of-the-art LMMs of 8 types, including Idefics2 [132], InstructBLIP [58], InternVL2 [39, 38], LLaVA-1.5 [162], LLaVA-NeXT [136], LLaVa-OV [136], Phi-3-Vision [2], Qwen2VL [253]. We choose these models as they are publicly available and widely adopted by the community. These models encompass different design choices such as the vision encoder (*e.g.*, CLIP [205],

---

[6] We use the model available at `https://spacy.io/models/en#en_core_web_lg`

SigLIP [290], BLIP-2 [141]), language model (*e.g.*, Mistral [108], Vicuna [42], Qwen2 [253]), data types (*e.g.*, instruction following, multilingual, textbook-based), and pre-training strategies (*e.g.*, single *vs* multi-stage). Unless otherwise stated, we query the model with the same prompt of [296], *i.e.*, *"What type of object is in this image?"*, letting the models perform unconstrained generation. We report a summary of the models and their differences in table 3.19.

References. As a reference, we consider baselines based on contrastive Vision Language Models. Specifically, we report results using CLIP [205] and SigLIP [290] in the closed-world setting, where the models can access the list of target class names. Additionally, we include two baselines that adapt CLIP to the OW setting by formulating image classification as a retrieval task. The first retrieves the closest caption from a predefined database, while the second, CaSED [48], leverages retrieved captions to generate a list of candidate classes for the final prediction. For both baselines, we use the same retrieval database as in [48].

Are LMMs good at OW classification?

In this section, we analyze the performance of LMMs in an OW setting, with results summarized in table 3.17 and extensively presented for each dataset and metric, *i.e.*, text inclusion in table 3.20, Llama inclusion in table 3.21, semantic similarity in table 3.22, and concept similarity in table 3.23.

Prototypical classification. LMMs perform best on prototypical classes, with high scores in inclusions and similarities. They consistently outperform CaSED and CLIP retrieval on inclusion metrics and are generally comparable or superior on similarity scores.

Non-prototypical classification. Performance drops significantly, with the highest LI score at 46.8, nearly 15 points lower than closed-world CLIP. Predictions are also less semantically indicative of the target class, with an average CS of 49.3, much lower than the prototypical case (69.2).

Fine-grained classification. Greater variation is observed among different models, ranging from 41.7 to 63.4 in concept similarity. In this group, LMM predictions are slightly less accurate than those of CaSED and CLIP retrieval.

Very fine-grained classification. Many models achieve a TI of 0.0, except for Qwen2VL 2B, which scores 12.9 due to the exceptional performance of FGVAircraft (25.6 *vs* 4.6 for the second-best model). Most LMMs underperform CLIP retrieval in CS, suggesting an issue due to granularity.

Model	**Vision Enc**	**Language Enc**	**Training**	**Pre-training**
Idefics2 [132]	SOViT (SigLIP), 0.4B params; max 980x980.	Mistral 7B	Interleaved web docs, image-caption pairs (LAION-COCO), OCR data; fine-tuned on 50 curated datasets.	Joint dual encoder training with Perceiver pooling for vision-text alignment.
InstructBLIP [58]	ViT-g (BLIP-2), 1.1B params; 224x224.	Vicuna 7B	26 datasets transformed into instruction-tuning format: captioning, VQA, image generation.	Two-stage pre-training: Vision-language alignment via BLIP-2 and instruction-aware Query Transformer for task-specific feature extraction.
InternVL2 [39]	InternViT (custom), 0.3B params (or 6B for larger models); dynamic resolution, max 40 tiles of 448×448.	Qwen2 0.5B (for 1B and 2B versions), or InternLM2 8B (for 8B version).	Interleaved image-text, multilingual OCR, mathematical charts; strict quality control.	Progressive training: masked video modeling, cross-modal contrastive learning, and next-token prediction with spatiotemporal focus.
LLaVA-1.5 [162]	ViT-L (CLIP), 0.3B params; 336x336.	Vicuna 7B	158K multimodal instruction-following samples; pre-trained on filtered CC dataset (596K image-text pairs).	Frozen vision encoder during feature alignment stage; end-to-end fine-tuning.
LLaVA-NeXT [135]	ViT-L (CLIP), 0.3B params; 336x336, 672x672, 336x1344, and 1344x336.	Mistral 7B, or Vicuna 7B	Diverse tasks, including multi-image and video understanding.	Builds on LLaVA with extended ViT and additional multimodal datasets for improved generalization.
LLaVA-OV [136]	SOViT (SigLIP), 0.4B params; dynamic resolution (AnyRes-9), max 2304x2304.	Qwen2 0.5B, or Qwen2 7B	Single-image and video scenarios with task transfer capabilities; diverse visual benchmarks.	Pre-trained with balanced visual token representation across scenarios to enable task transfer.
Phi-3-Vision [2]	ViT-L (CLIP), 0.4B params; dynamic resolution, max 1344x1344.	Phi-3 Mini (3.8B params)	Synthetic data, filtered public docs, high-quality interleaved text-image data, math-/code examples.	Multi-stage training: custom vision encoder aligned with Phi-3 Mini language model using interleaved and fine-grained tasks.
Qwen2VL [253]	ViT (custom), 0.6B params; dynamic resolution (Naive Dynamic Resolution), no max.	Qwen2 1.5B, or Qwen2 7B	Multilingual datasets: MathVista, DocVQA, RealWorldQA; supports videos (20+ min) and multilingual text in images.	Pre-trained with dynamic resolution ViT for flexible input sizes and multilingual alignment strategies.

Table 3.19: Summary details of the models used in our analyses.

	Textual inclusion										
Model	C101	DTD	ESAT	FGVC	FLWR	FOOD	PETS	CARS	S397	U191	Avg.
Idefics2 [132] 8B	52.0	1.7	1.6	0.0	0.8	8.2	0.1	0.0	9.6	7.9	8.2
InstructBLIP [58] Vicuna 7B	47.8	3.0	5.5	0.0	6.0	24.3	0.8	0.0	11.6	9.6	10.9
InternVL2 [39, 38] 2B	52.8	10.8	7.4	1.4	14.1	23.3	7.2	0.0	21.1	12.4	15.0
InternVL2 [39, 38] 4B	49.6	11.8	6.0	3.4	12.8	28.2	7.8	0.0	23.0	12.7	15.5
InternVL2 [39, 38] 8B	55.0	12.5	6.0	4.6	19.1	33.9	13.8	**0.1**	26.3	**14.4**	18.6
LLaVA-1.5 [162] 7B	51.6	6.0	**11.7**	0.1	6.7	17.6	1.1	0.0	17.6	8.2	12.1
LLaVA-NeXT [135] (Mistral 7B)	58.0	13.6	7.4	2.8	17.6	35.5	27.1	0.0	25.4	13.0	20.0
LLaVA-NeXT [135] (Vicuna 7B)	54.9	12.2	7.2	2.5	11.9	29.6	9.4	0.0	24.0	12.5	16.4
LLaVa-OV [136] (Qwen2 0.5B)	53.4	9.2	4.2	1.2	2.9	12.6	2.5	**0.1**	15.5	8.7	11.0
LLaVa-OV [136] (Qwen2 7B)	55.5	12.6	4.9	0.0	14.2	5.0	0.1	0.0	6.2	4.0	10.2
Phi-3-Vision [2]	53.4	10.9	0.8	0.4	12.0	21.6	6.5	**0.1**	14.7	6.5	12.7
Qwen2VL [253] 2B	60.8	12.1	0.4	**25.6**	**42.9**	48.5	15.7	**0.1**	29.0	10.8	**24.6**
Qwen2VL [253] 7B	**63.2**	**15.7**	2.7	1.4	42.3	**49.3**	12.1	**0.1**	**29.5**	12.5	22.9
Open-world baselines											
CaSED [48]	35.5	5.1	3.0	1.4	28.1	19.4	**34.6**	0.0	13.5	8.1	14.9
CLIP retrieval	42.6	7.5	6.6	14.0	40.6	26.4	30.3	0.0	14.7	8.4	19.1
Closed-world baselines											
CLIP [205]	87.1	52.6	42.7	27.2	76.9	89.9	88.1	76.2	65.6	72.7	67.9
SigLIP [290]	93.6	60.8	42.1	46.0	88.2	94.1	95.4	92.3	69.9	82.1	76.5

Table 3.20: Text inclusion on the ten datasets. Higher is better, **bold** indicates best.

	Llama inclusion										
Model	C101	DTD	ESAT	FGVC	FLWR	FOOD	PETS	CARS	S397	U191	Avg.
Idefics2 [132] 8B	72.9	24.6	19.0	64.4	54.6	58.7	36.3	69.6	32.5	40.1	47.3
InstructBLIP [58] Vicuna 7B	76.8	26.2	19.1	59.9	57.4	47.6	41.3	62.0	35.8	36.0	46.2
InternVL2 [39, 38] 2B	74.9	48.5	35.0	35.8	49.3	44.3	47.4	30.0	64.9	52.1	48.2
InternVL2 [39, 38] 4B	74.4	45.7	30.1	40.5	37.5	45.9	49.7	33.1	62.5	50.4	47.0
InternVL2 [39, 38] 8B	77.2	50.5	28.6	29.7	36.0	53.7	50.4	35.3	71.5	**59.6**	49.3
LLaVA-1.5 [162] 7B	74.5	39.4	**45.0**	44.5	46.3	47.7	45.5	37.5	51.6	48.5	48.1
LLaVA-NeXT [135] (Mistral 7B)	77.8	54.0	28.0	43.4	33.4	63.2	34.6	50.9	69.9	58.3	51.4
LLaVA-NeXT [135] (Vicuna 7B)	77.3	52.2	26.4	43.1	29.2	60.6	43.6	41.2	68.2	59.1	50.1
LLaVa-OV [136] (Qwen2 0.5B)	76.5	46.5	28.7	61.2	55.1	28.1	44.9	70.0	52.2	35.8	49.9
LLaVa-OV [136] (Qwen2 7B)	81.3	45.6	11.8	**68.9**	48.9	22.0	50.2	**84.4**	25.0	27.0	46.5
Phi-3-Vision [2]	75.7	45.3	6.0	51.0	53.2	45.1	49.1	39.0	44.5	34.7	44.4
Qwen2VL [253] 2B	82.9	54.6	3.1	65.0	67.0	71.1	49.3	56.3	72.6	45.2	56.7
Qwen2VL [253] 7B	**84.3**	**60.8**	18.1	58.8	**71.0**	**75.0**	46.0	67.2	**73.0**	48.8	**60.3**
Open-world baselines											
CaSED [48]	57.7	16.7	7.3	30.7	46.0	35.1	**58.7**	63.5	34.9	31.7	38.2
CLIP retrieval	55.3	28.2	12.7	25.8	44.6	35.4	56.2	10.4	30.5	32.9	33.2
Closed-world baselines											
CLIP [205]	87.1	52.6	42.7	27.2	76.9	89.9	88.1	76.2	65.6	72.7	67.9
SigLIP [290]	93.6	60.8	42.1	46.0	88.2	94.1	95.4	92.3	69.9	82.1	76.5

Table 3.21: Llama inclusion on the ten datasets. Higher is better, **bold** indicates best. Note that the scores for CLIP closed-world equals the textual inclusion scores.

	Semantic similarity										
Model	C101	DTD	ESAT	FGVC	FLWR	FOOD	PETS	CARS	S397	U191	Avg.
Idefics2 [132] 8B	64.9	34.6	27.5	27.6	38.6	44.4	30.8	31.6	44.2	44.0	38.8
InstructBLIP [58] Vicuna 7B	**71.5**	32.8	30.0	21.4	38.9	41.6	26.4	38.5	42.1	48.3	39.1
InternVL2 [39, 38] 2B	50.5	25.6	26.0	23.4	31.2	39.6	23.9	42.9	43.3	43.1	34.9
InternVL2 [39, 38] 4B	49.2	26.1	24.7	23.6	30.2	41.1	24.6	44.1	43.8	41.8	34.9
InternVL2 [39, 38] 8B	50.1	26.7	24.4	25.5	32.8	44.2	27.3	46.6	46.3	44.6	36.8
LLaVA-1.5 [162] 7B	49.0	24.2	**34.2**	19.0	25.8	37.2	21.5	38.2	41.7	40.7	33.1
LLaVA-NeXT [135] (Mistral 7B)	48.2	27.7	23.9	23.6	30.2	45.3	30.3	44.8	43.6	42.1	36.0
LLaVA-NeXT [135] (Vicuna 7B)	49.2	27.9	23.1	23.4	29.3	43.0	24.4	45.7	43.3	42.3	35.1
LLaVA-OV [136] (Qwen2 0.5B)	64.7	28.8	21.6	21.0	41.4	42.7	31.4	40.0	43.2	47.9	38.3
LLaVA-OV [136] (Qwen2 7B)	68.7	32.2	19.4	29.4	37.5	41.7	37.8	34.4	43.4	43.2	38.8
Phi-3-Vision [2]	53.6	28.5	12.3	18.8	30.9	40.1	24.3	39.0	41.8	37.3	32.7
Qwen2VL [253] 2B	56.4	27.0	13.5	**32.8**	43.7	50.6	27.8	**57.4**	47.9	42.7	40.0
Qwen2VL [253] 7B	55.8	28.5	20.7	20.6	41.8	50.6	25.1	48.5	48.1	43.2	38.3
Open-world baselines											
CaSED [48]	65.3	**39.9**	32.2	30.0	**55.6**	**64.1**	**62.4**	47.1	**52.4**	**53.4**	**50.2**
CLIP retrieval	41.3	23.6	22.4	30.7	40.3	46.7	41.7	48.8	39.1	38.5	37.3
Closed-world baselines											
CLIP [205]	90.8	69.9	67.7	66.7	83.4	93.7	91.8	80.5	92.2	83.3	82.0
SigLIP [290]	97.8	75.6	63.1	80.0	92.0	96.4	96.8	98.1	83.1	89.6	87.3

Table 3.22: Semantic similarity on ten datasets. Higher is better, **bold** indicates best.

	Concept similarity										
Model	C101	DTD	ESAT	FGVC	FLWR	FOOD	PETS	CARS	S397	U191	Avg.
Idefics2 [132] 8B	76.3	38.5	30.9	29.7	41.5	48.4	35.3	37.5	49.9	54.6	44.3
InstructBLIP [58] Vicuna 7B	75.3	39.1	31.6	28.6	43.6	60.0	37.9	40.0	52.6	55.3	46.4
InternVL2 [39, 38] 2B	75.7	48.0	**52.9**	36.8	49.5	60.8	41.9	50.9	65.1	59.4	54.1
InternVL2 [39, 38] 4B	76.1	48.6	51.5	37.9	51.0	63.0	41.9	50.5	65.4	59.1	54.5
InternVL2 [39, 38] 8B	78.7	49.7	49.1	42.5	56.9	67.1	46.0	56.2	69.2	**62.9**	57.8
LLaVA-1.5 [162] 7B	72.1	41.3	51.6	29.0	41.6	56.8	35.9	46.2	59.4	55.5	48.9
LLaVA-NeXT [135] (Mistral 7B)	79.8	**51.0**	49.5	37.5	55.1	70.0	55.3	56.3	68.7	62.7	58.6
LLaVA-NeXT [135] (Vicuna 7B)	79.0	50.1	50.8	37.1	51.3	65.8	42.4	55.0	67.4	61.8	56.1
LLaVA-OV [136] (Qwen2 0.5B)	77.8	45.1	39.9	30.6	42.4	50.0	37.5	43.5	56.7	55.9	47.9
LLaVA-OV [136] (Qwen2 7B)	79.1	47.0	41.0	29.4	51.7	41.9	37.8	35.4	44.9	43.3	45.1
Phi-3-Vision [2]	74.1	44.0	25.3	29.1	43.0	58.3	40.3	42.9	56.1	49.1	46.2
Qwen2VL [253] 2B	79.4	47.3	24.2	**56.0**	67.9	75.7	46.7	**68.6**	70.0	56.6	**59.2**
Qwen2VL [253] 7B	**81.3**	50.4	39.8	30.8	**68.8**	**76.9**	43.1	56.0	**70.6**	59.1	57.7
Open-world baselines											
CaSED [48]	65.9	39.8	32.2	29.9	55.6	66.5	62.9	47.1	53.7	55.1	50.9
CLIP retrieval	63.9	38.1	37.8	50.7	62.3	67.8	**66.1**	61.5	57.3	54.4	56.0
Closed-world baselines											
CLIP [205]	90.8	69.9	67.7	66.7	83.4	93.7	91.8	80.5	92.2	83.3	82.0
SigLIP [290]	97.8	75.6	63.1	80.0	92.0	96.4	96.8	98.1	83.1	89.6	87.3

Table 3.23: Concept similarity on ten datasets. Higher is better, **bold** indicates best.

Overall trends. Across ten datasets, CLIP retrieval outperforms 9/13 models in TI, but LMMs consistently achieve higher Llama inclusion scores. CaSED ranks highest in semantic similarity due to its concise responses, while CLIP retrieval remains competitive. These results confirm insights from previous works, such as the influence of data exposure on coarse-grained categories [296, 163]. Additionally, stronger language models (*e.g.*, Mistral, Qwen) tend to yield better results than weaker counterparts (*e.g.*, Vicuna). LMMs generally outperform contrastive models in OW classification, leading in 11/16 metric/groups. However, top-performing models in one metric may struggle in others–for example, Qwen2VL 7B excels in LI on very fine-grained datasets, while InternVL2 8B and LLaVA-OV (Qwen2 7B) show different strengths in prototypical classification: *e.g.*, +21.2 LI of the first on the second but -7.9 SS.

While these results are promising, the gap with closed-world models is still large, *i.e.*, CLIP [205], SigLIP [290]. To better understand OW predictions, in the next section, we further explore what the proposed metrics metrics capture.

3.3.3 Experimental results

Interpreting model predictions through inclusion and similarity scores

To underscore the importance of jointly evaluating inclusion and similarity scores, we present qualitative results demonstrating how their combined analysis offers deeper insights into LMM failures. Figure 3.12 showcases qualitative results from various datasets, displaying ground truth class labels alongside model predictions. For instance, the case of *Caprese salad* illustrates this distinction: more descriptive predictions like *creamy sauce*, which LI considers incorrect, receive a relatively higher CS score than *food*, which is deemed correct by LI. This emphasizes that, despite *creamy sauce* being semantically closer to the ground truth, the label is wrong according to Llama due to its lack of alignment with the ground truth. Similar behavior is present in the other examples.

To reinforce our previous point, fig. 3.13 illustrates the relationship between LI and CS, highlighting the distinct contributions of these two metrics. Predictions in the top-right quadrant correspond to concepts that are semantically close to the ground truth and are also likely to be considered correct by LI (*e.g.*, *cellphone* with predictions such as *mobile phone* and *Nokia*). In contrast, the bottom-left quadrant represents the opposite case. For instance, in the same plot, *handheld device*—while somewhat related to *cellphone* and receiving a nonzero CS score–is still deemed incorrect by LI. Similarly, in the *Caesar salad* example, the prediction *food* appears in the bottom-right

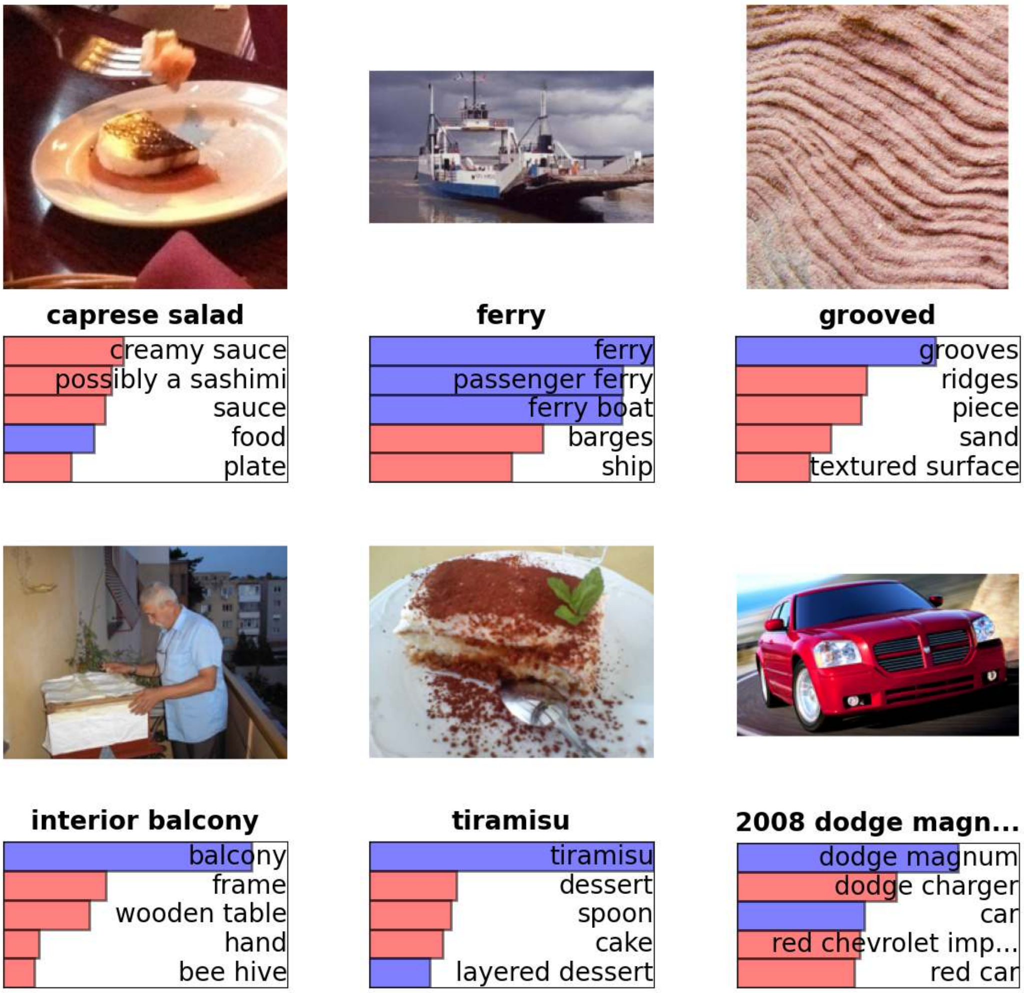


Figure 3.12: Per-image examples of model predictions. **Bold** indicates the ground truth class names. For visualization purposes, we show only the predictions with the highest/lowest concept similarity. Blue and red indicate positive and negative Llama inclusion values.

quadrant, as it is correct but overly generic. Meanwhile, *pasta salad*, being more specific yet incorrect, falls into the top-left quadrant.

Grouping model predictions

Following the intuition from above, we analyze the performance of LMMs defining four different groups of predictions: correct and specific, correct but generic (*e.g.*, *dog vs pug*); wrong but specific, predicting classes semantically similar to the target (*e.g.*, *pug vs pomeranian*); and wrong and generic, *i.e.*, where the prediction is semantically dissimilar from the target (*e.g.*, *sofa vs dalmatian*). To define these groups, we split the

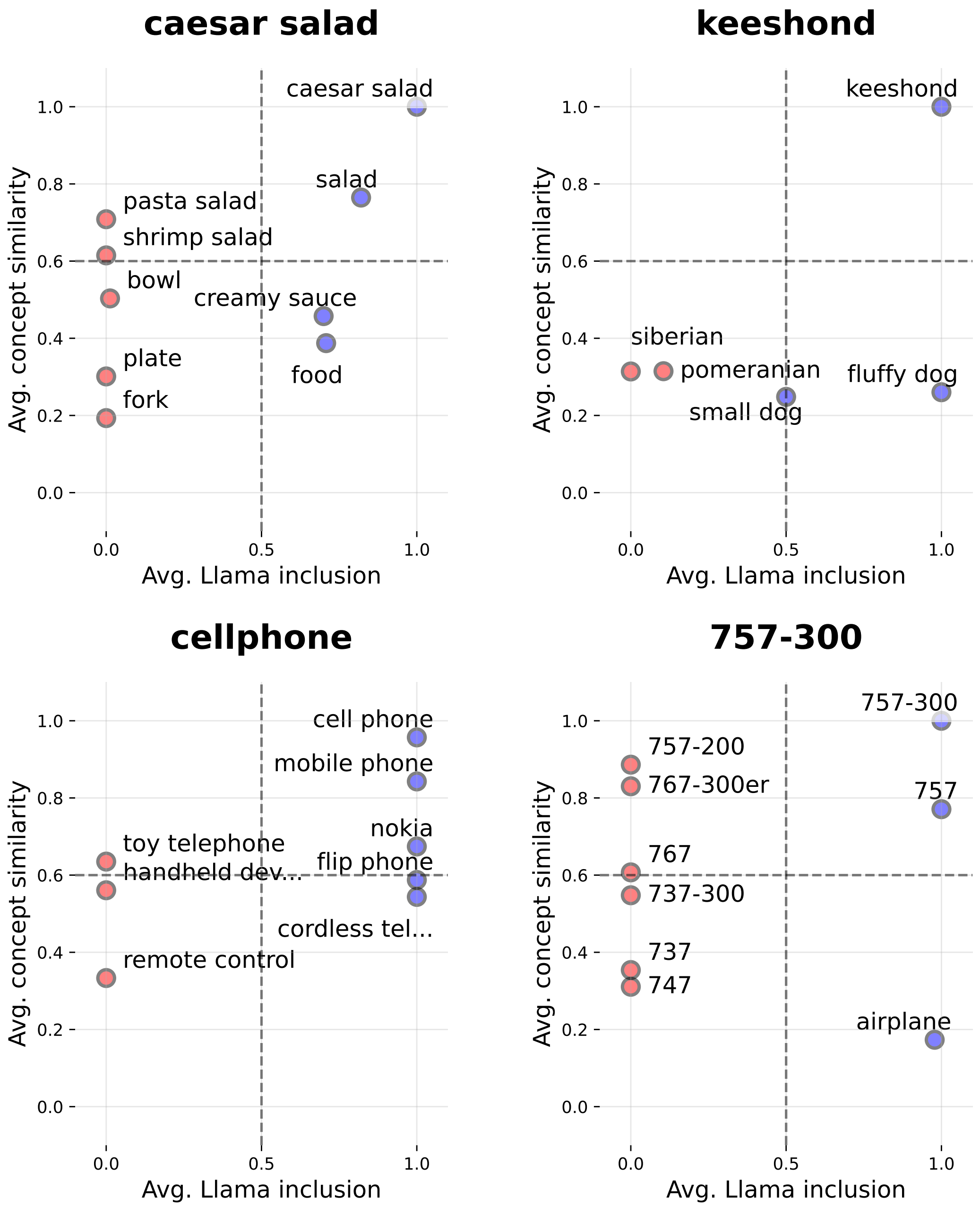
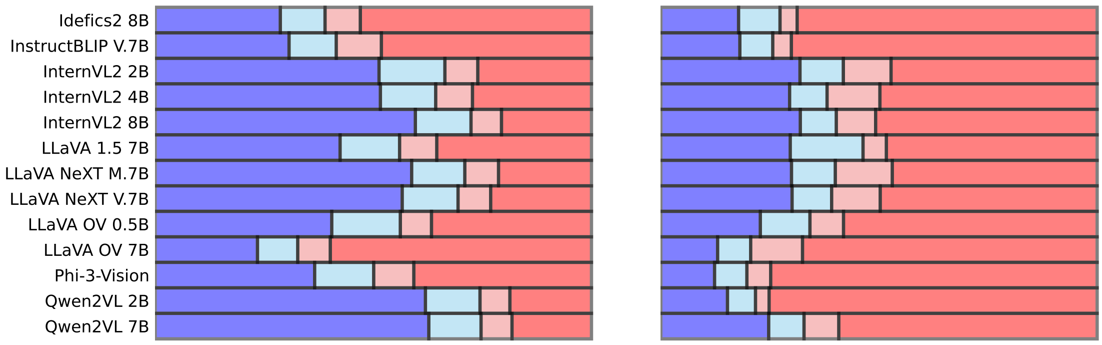
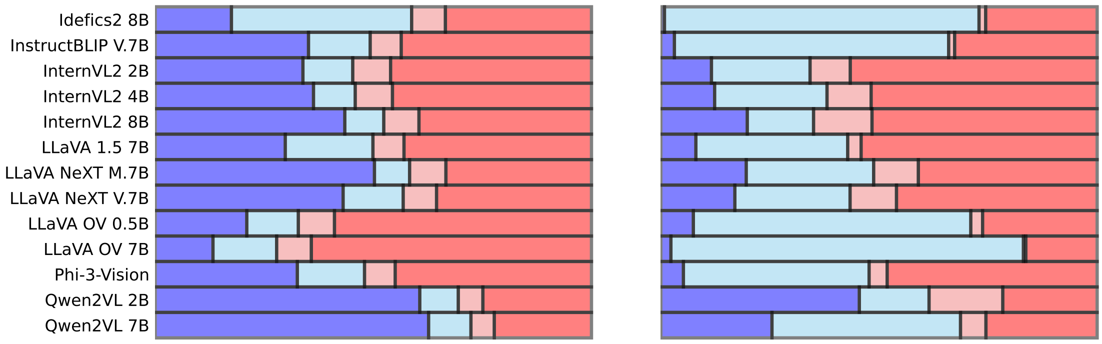


Figure 3.13: Per-class examples of model predictions. **Bold** indicates the ground truth class names. On the x-axis we report the average LI, and on the y-axis the average CS. For visualization purposes, we show the most frequent concepts predicted for each quadrant.



(a) Prototypical (b) Non-prototypical



(c) Fine-grained (d) Very fine-grained

Figure 3.14: Types of model predictions per dataset groups. We indicate with blue correct and specific and correct but generic predictions, and with red wrong but specific and wrong and generic mistakes.

model predictions into four sets by thresholding the LI and CS scores. We arbitrarily set the CS threshold at 0.6 to distinguish between generic and specific responses and the LI threshold at 0.5 to separate correct and wrong responses[7]. We visualize the ratios for the predictions in fig. 3.14. Intuitively, a good LMM should have an high amount of predictions as correct and specific. When not possible, however, having an equally high correct but generic ratio is still better than having errors of any form.

In terms of optimal predictions, we see that the best-performing models vary according to dataset groups. For prototypical classification, the models with the lowest error are InternVL 8B, Qwen2VL 2B, and Qwen2VL 7B. For non-prototypical tasks, instead, LLaVA 1.5 7B performs best, but InternVL 2B and InternVL 8B provide slightly more precise predictions. For fine-grained, trends are similar to the prototypical groups but with fewer correct and more generic responses. This is most evident for Idefics2 8B,

---

[7] Note that LI is either 0 or 1 on a per-sample basis, but it ranges between the two when considering aggregated results, *e.g.*, average per class.

Dataset	Agreement High (%)	 Medium (%)	 Low (%)
C101	71.4	15.8	12.8
S397	34.3	33.0	32.8
U101	33.8	26.8	39.5
FOOD	32.6	27.5	39.9
DTD	23.3	29.1	47.6
FLWR	13.9	25.5	60.6
ESAT	6.1	21.8	72.1
PETS	5.5	16.1	78.4
CARS	1.5	21.9	76.6
FGVC	0.1	4.0	96.0

Table 3.24: Agreement of LMMs correct predictions across datasets. Low indicates that less than 30% of the models predicted a sample correctly, while high indicates that more than 70% did.

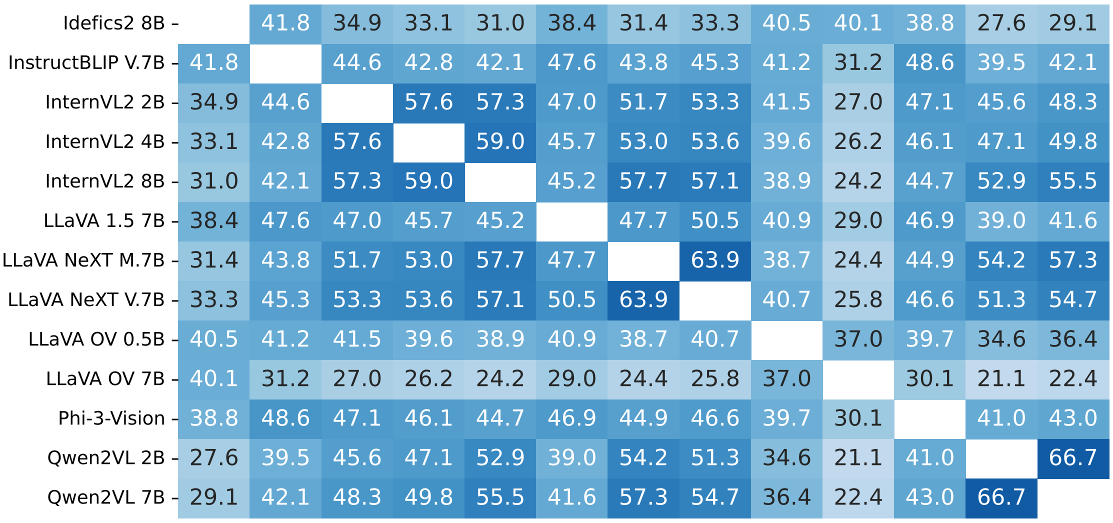
which works fairly well on fine-grained classification but provides responses lacking specificity. On very fine-grained, we perceive higher rates of wrong and generic, with more generic predictions across all models. Notably, Qwen2VL models perform better in the last two settings. On average, the models with the highest wrong predictions are LLaVA-OV 7B and InstructBLIP Vicuna 7B. The model that is, on average, more generic in its replies is Idefics2 8B.

Analyzing LMMs mistakes in OW

In the following, we further inspect the correct and wrong predictions of the different models. Specifically, each section will analyze one of the four cases: correct and specific (section 3.3.3), correct but generic (section 3.3.3), wrong but specific (section 3.3.3), and wrong and generic (section 3.3.3).

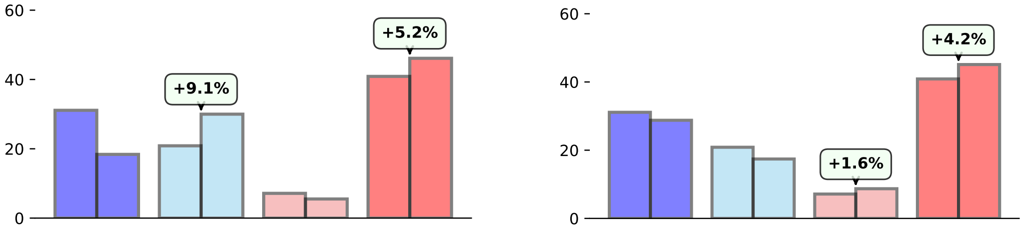
Correct and specific While this section describes successful cases, from section 3.3.2 we know that models perform differently. Thus, here we investigate whether LMMs share similar success cases.

Are correct predictions shared among models? To answer this question, we first evaluate the percentage of samples that receive correct predictions by multiple models across datasets. We report the results in table 3.24, splitting them according to low (less than 30% of models), medium (30%-70%), and high agreement (above 70%). The table shows that the models tend to agree on prototypical datasets (*e.g.*, 71.4% of high

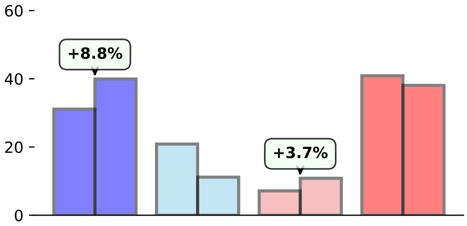


Idefics2 8B		41.8	34.9	33.1	31.0	38.4	31.4	33.3	40.5	40.1	38.8	27.6	29.1
InstructBLIP V.7B	41.8		44.6	42.8	42.1	47.6	43.8	45.3	41.2	31.2	48.6	39.5	42.1
InternVL2 2B	34.9	44.6		57.6	57.3	47.0	51.7	53.3	41.5	27.0	47.1	45.6	48.3
InternVL2 4B	33.1	42.8	57.6		59.0	45.7	53.0	53.6	39.6	26.2	46.1	47.1	49.8
InternVL2 8B	31.0	42.1	57.3	59.0		45.2	57.7	57.1	38.9	24.2	44.7	52.9	55.5
LLaVA 1.5 7B	38.4	47.6	47.0	45.7	45.2		47.7	50.5	40.9	29.0	46.9	39.0	41.6
LLaVA NeXT M.7B	31.4	43.8	51.7	53.0	57.7	47.7		63.9	38.7	24.4	44.9	54.2	57.3
LLaVA NeXT V.7B	33.3	45.3	53.3	53.6	57.1	50.5	63.9		40.7	25.8	46.6	51.3	54.7
LLaVA OV 0.5B	40.5	41.2	41.5	39.6	38.9	40.9	38.7	40.7		37.0	39.7	34.6	36.4
LLaVA OV 7B	40.1	31.2	27.0	26.2	24.2	29.0	24.4	25.8	37.0		30.1	21.1	22.4
Phi-3-Vision	38.8	48.6	47.1	46.1	44.7	46.9	44.9	46.6	39.7	30.1		41.0	43.0
Qwen2VL 2B	27.6	39.5	45.6	47.1	52.9	39.0	54.2	51.3	34.6	21.1	41.0		66.7
Qwen2VL 7B	29.1	42.1	48.3	49.8	55.5	41.6	57.3	54.7	36.4	22.4	43.0	66.7	

Figure 3.15: Percentage of correct and specific predictions shared between models. Higher values indicate higher agreement.



(a) Be generic (b) Be specific



(c) Domain-specific

Figure 3.16: Average gains per prediction type when asking models to be more generic/specific (a, b), or via dataset-specific prompts (c). Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes.

agreement on C101) but they do not for very fine-grained ones (*i.e.*, CARS and FGVC). Overall, we found that only 5.6% of the samples are correctly predicted by all models and there exist 6 labels out of almost 1200 that are never predicted correctly according to the LI score: *i.e.*, *birman*, *bishop of llandaff*, *egyptian mau*, *prince of wales feather*, *silverbush*, and *watercress*, all belonging to fine-grained datasets. These results confirm the ability of LMMs to capture generic concepts while struggling with very specific ones. In fig. 3.14, we observe that when the granularity constraint is relaxed, most models continue to predict the parent class with a remarkable level of accuracy, given the nature of the task.

Which models agree the most with each other? We additionally check the pairwise agreement on the model predictions on the correct and specific group, showing the results in fig. 3.15. Interestingly, models of the same family tend to share more predictions, *i.e.*, Qwen2VL 2B and Qwen2VL 7B share 66.7% correct and specific predictions, InternVL2 4B and InternVL2 8B 59.0%. This also happens with different language models (*e.g.*, LLaVA NeXT with Mistral and Vicuna share 63.9% of correct predictions), and differences might arise within lower performing families (*e.g.*, LLaVA-OV 0.5B and 7B agree only 37% of the time). While there is no clear pattern, the best-performing families (*e.g.*, LLaVA NeXT, InternVL2, Qwen2VL) tend to share more than half of the correct predictions (*e.g.*, InternVL2 8B and Qwen2VL 7B, 55.5%), suggesting that the main drive for the agreement is model capabilities rather than design choices.

Which model provides the best responses? To analyze which model provides the best responses, we compare their generations in pairs. Specifically, for each of the ten datasets, we randomly sample 10'000 pairs of generations, and instruct a Llama 3.2 model to identify the best response in the pair, similarly to what is done in the Chatbot Arena [43] but through automatic evaluation with LLM-as-a-judge [302]. We use the following prompt to instruct Llama 3.2 to judge the pairs of predictions and decide for a win:

	Average Elo ratings	
Rank	**Model**	**Rating**
1	Qwen2VL [253] 2B	1037
2	Qwen2VL [253] 7B	1037
3	Phi-3-Vision [2]	1029
4	LLaVA-NeXT [135] (Mistral 7B)	1018
5	LLaVA-NeXT [135] (Vicuna 7B)	1015
6	LLaVa-OV [136] (Qwen2 7B)	1014
7	LLaVa-OV [136] (Qwen2 0.5B)	1007
8	InternVL2 [39, 38] 8B	1004
9	InternVL2 [39, 38] 4B	994
10	InternVL2 [39, 38] 2B	991
11	LLaVA-1.5 [162] 7B	984
12	InstructBLIP [58] Vicuna 7B	943
13	Idefics2 [132] 8B	924

Table 3.25: Elo ratings on the ten datasets. Higher scores indicate comparatively better responses from the models.

Llama Elo ranking

You are a model that discriminates whether labels A or B better align with a target value.

This is label A: %s
This is label B: %s
This is the target value: %s

Does A align better with the target value? Does B align better with the target value? Reply only with "1" if A wins over B, or "0" if B wins over A.

We directly compare the quality of the outputs by evaluating the Elo score [64] of these model responses and report the average on the ten datasets in table 3.25. Results show that Qwen2VL models are the best at providing accurate predictions, similar to the trend in table 3.17.

Correct but generic As different classification scenarios may require different levels of granularity, we also check whether we can control the latter via prompting. We investigate three types of requests: *"Be generic."*, *"Be specific."*, and *domain*-specific prompts, focusing on the fine-grained and very fine-grained datasets, alongside DTD. In fig. 3.16, we report the average difference across datasets for each group of predictions and type of prompt, reporting in the appendix A the metric variations on datasets and models.

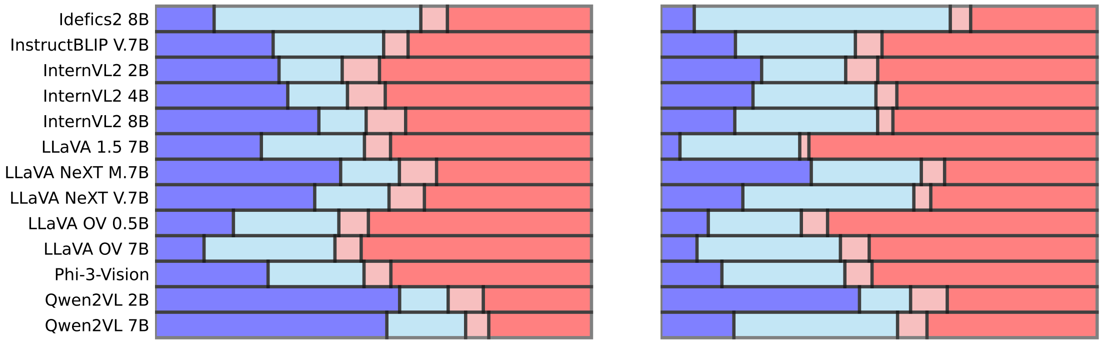
Be more generic. When queried for generic responses, we see a large shift from correct and specific predictions to correct but generic, and, to a smaller degree, the same happens for wrong ones. This highlights how models can provide good generic responses (+9.1%) but the large decrease in correct and specific ones means they become too generic.

Be more specific. In this case, all LMMs consistently get worse, equally increasing wrong but specific and wrong and generic predictions. While a decrease (especially in correct but generic) is expected, this hints that LMMs are stronger at providing more generic replies than more specific ones.

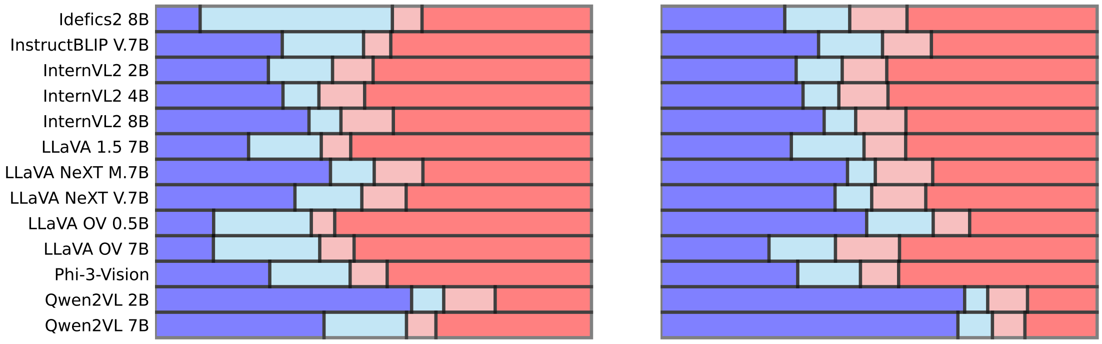
Domain-specific. When tackling specific fine-grained scenarios, it is possible to tailor a custom prompt, *e.g.*, when classifying flowers, we can directly ask *"What type of flower is in this image?"* instead of a generic object. Therefore, we explore whether informing the LMM on the target fine-grained scenario may fix the specificity issue. We update the prompt to use the terms "texture" (for DTD), "aircraft" (for FGVC), "flower" (for FLWR), "food" (for FOOD), "pet" (for PETS), or "car" (for CARS). Overall, domain-specific prompts positively influence the predictions, converting an average of 12.5% of generic responses into specific ones. Notably, LLaVA-OV 0.5B gets +29% on the correct and specific set, followed by Qwen2VL 7B with +15% (see appendix A). This shows how, while LMMs struggle to provide specific predictions off-the-shelf, injecting domain-specific OW performance.

Wrong but specific Here we analyze mistakes due to two objects being very similar (*e.g.*, *euphonium vs trombone*). As addressing this type of mistake requires reasoning on fine-level details of the images, we explore whether test-time reasoning can improve performance. Thus, we study the impact of introducing Chain-of-Thought (CoT) [255, 119] during inference.

Can CoT mitigate misclassification? We identify three simple techniques we can apply without modifying the architecture of the models: zero-shot CoT [119] appending the instruction *"Think step by step."* to the input query, *LlamaV-o1 prompt* using the multi-turn procedure of [241], and the *LLaVA-CoT prompt* [260] for reasoning in procedures. For this study, we focus on the InternVL2 and the Qwen2VL families, showing their average gains in fig. 3.18. Additional results are available in appendix A.

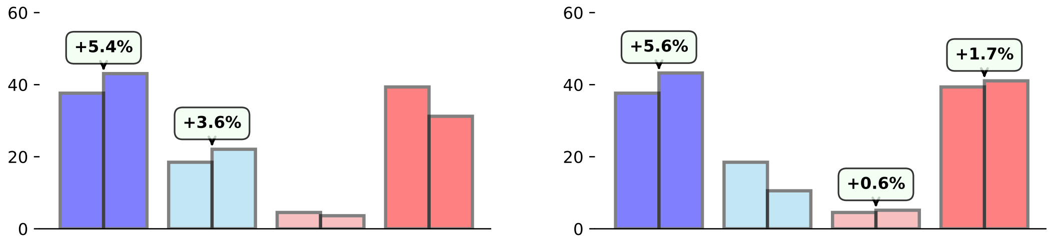


(a) Default prompt (b) Ask for generic

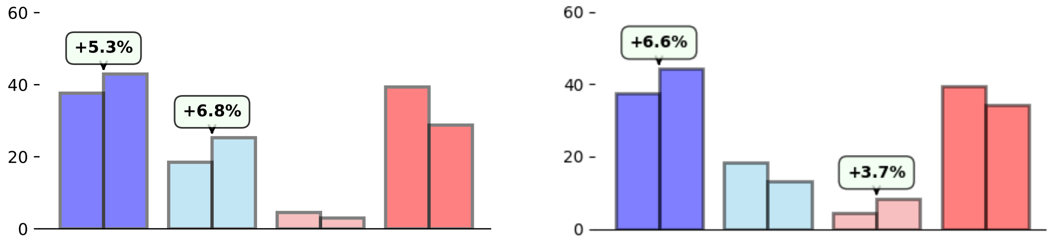


(c) Ask for specific (d) Dataset prompts

Figure 3.17: Types of model predictions when using the generic and specific prompts and the dataset-specific prompts. Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes.



(a) Zero-shot CoT (b) LlamaV-o1 prompt



(c) LLaVA-CoT prompt (d) Reasoning models

Figure 3.18: Average gains per prediction type when including chain-of-thought reasoning (a, b, c), or when with reasoning models (d). Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes.

Notably, test-time reasoning helps the models in making correct and specific responses. Performance-wise, Qwen2VL shows the highest gains, achieving up to +13% in correct and specific responses. While test-time reasoning enhances the OW of LMMs, we observe that the multi-turn prompt tends to steer the model either toward semantically correct predictions or completely divergent ones (+1.7 on wrong and generic). On the other hand, simply instructing LMMs to think with zero-shot CoT or providing a longer prompt (*LLaVA-CoT*), consistently increases their accuracy.

Do models tailored for reasoning excel in OW? As we saw positive gains from using test-time reasoning, below we further explore the capabilities of more advanced approaches. Specifically, both InternVL2 and Qwen2-VL have two improved versions tailored for reasoning: InternVL2.5 [36] and Qwen2.5VL [12]. In the following, we check whether these variants outperform their predecessors, less tailored to reasoning. We show the average relative gains in fig. 3.18 (d).

By directly replacing the base models with their reasoning counterparts, we get mixed results, *i.e.*, we see a large increase in correct prediction (+6.6% on average), but also in misclassification with semantically close concepts (+3.7%), the error we wanted to address. This shows that test-time reasoning might be more effective than off-the-self

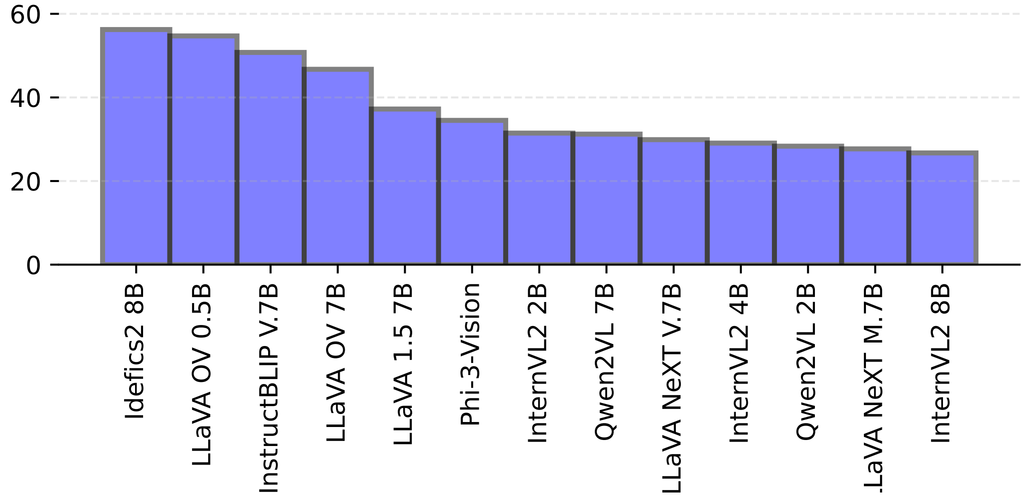


Figure 3.19: Percentage of model predictions considered wrong in the single-label setting but correct in multi-label.

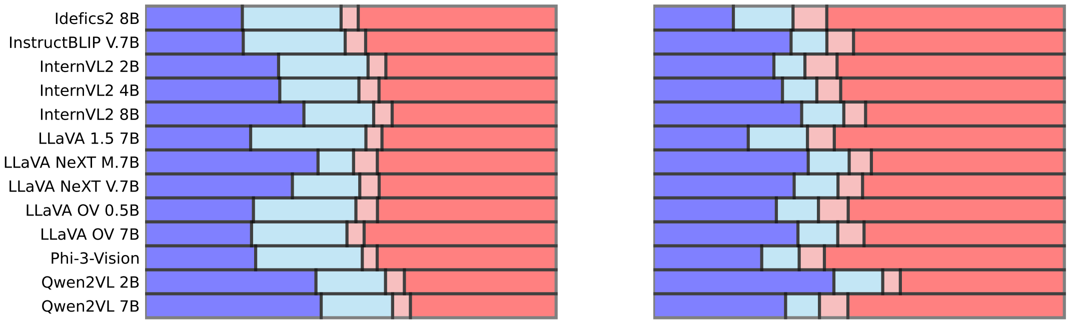
reasoning models at addressing such nuanced cases.

Wrong and generic In this category, predictions are not only wrong according to inclusion metrics but also based on semantic ones. While some of the mistakes are due to the lack of fine-grained understanding of the models (see section 3.3.3), we investigate to which extent LMMs are correct even within wrong predictions. Specifically, we explore cases where models simply focus on the wrong object in the image.

Do LMMs focus on the wrong object? To investigate this, we annotate images with multiple labels using RAM++ [101], a state-of-the-art model for tagging images with a list of concepts. Then, we compare LMM predictions to the list of tags, looking for cases with an extremely high CS (above 0.95 with any of the tags in the image). If this is the case, we assume the prediction is relevant for the image, despite its differences with the true label.

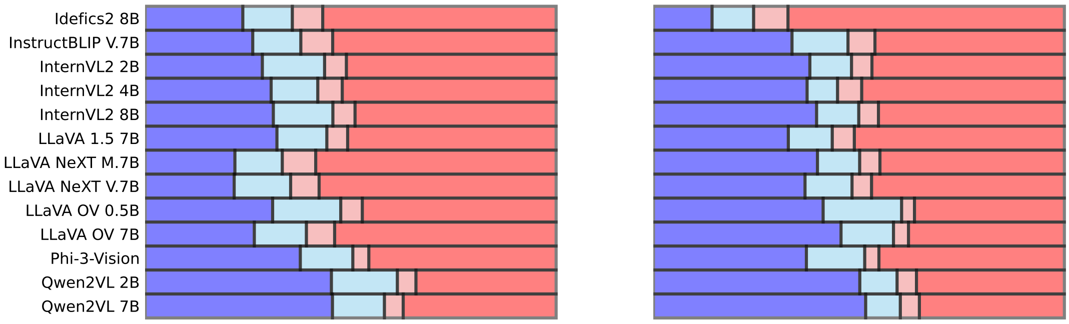
Figure 3.19 shows the percentage of wrong predictions that match a tag. As shown in the table, this percentage is high, ranging between 30% and 60% of the wrong predictions. Notably, this is high also for models with lower overall performance in table 3.17, such as Idefics2 and InstructBLIP.

Predicting more concepts. The previous experiment suggested LMMs often fail to predict the class names because they focus on the wrong part of the image. However, when prompted to provide multiple candidates, do LMMs get the correct prediction? To investigate this, we ask the model to (i) list the objects in the image; (ii) caption it, or (iii) describe its content. We report the relative gain per model in table A.7. The



(a) Default prompt

(b) List objects



(c) Caption

(d) Describe

Figure 3.20: Types of model predictions when using multi-label prompts. Blue indicates correct and specific and correct but generic predictions, red indicates wrong but specific and wrong and generic mistakes.

results show that providing outputs that focus on multiple labels on average improves the concept-based similarity, with the only exception of the caption case. Text inclusion improves consistently, showing that predictions become correct even according to this strict metric. Overall, these results highlight how LMM mistakes can be ascribed by mismatches between the label and the focus of the annotator, with the models often focusing on grounded image content even in case of mistakes.

3.3.4 Related work

Large Multimodal Models. While early large Vision Language Models aligned visual and text representation in a shared embedding space [205, 290], there has been an increasing effort in developing generative multimodal models [3, 141, 308, 229]. These models process an input image and text generating either a text [141, 162] or a multimodal [118, 301] output. While these models share common components (*e.g.*, visual and text encoders, text decoders) they differ in the specific strategies for modality alignment (*e.g.*, MLP projector [162], Q-former [141]), pretraining (*e.g.*, autoregressive [162], alignment [141]), fine-tuning (*e.g.*, supervised, instruction tuning) but also data source (*e.g.*, web data [132], textbook-style [2]) and structure (*e.g.*, captioning [58], interleaved image-text [132]). In this work, we do not aim to introduce a new LMM or novel methodologies for building LMMs. Instead, we focus on evaluating how these models perform in OW classification, testing 13 different models belonging to 8 different families [132, 39, 58, 162, 135, 136, 2, 11], covering multiple architectural, data, and design choices.

Classification with LMMs. Multiple works designed benchmarks to test the general capabilities [137, 164, 144], or shortcomings [97, 283, 150, 307] of LMMs. The most closely related works to ours are [280, 296, 163], investigating their classification performance. Yue *et al.* [280] developed an approach exploiting the next token prediction probability of an LMM, reporting results on multi-label recognition. Zhang *et al.* [296] tested multiple LMMs on closed-world and acrshortow settings, showing how data influences their performance and that generative LMMs underperform their contrastive counterpart. This latter finding is challenged by Liu *et al.* [163], who extended the analyses of [296] to multiple datasets and more recent models. However, [163] focused on closed-world classification, while [296] limited the analyses on OW to 4 datasets and a single metric (*i.e.*, text inclusion). In this work, we expand existing analyses in OW classification with LMMs, providing the largest study up-to-date in terms of datasets (10) and models (13). We also analyze the performance of LMMs according to four different metrics, capturing complementary aspects. Moreover, we use these metrics to analyze LMM mistakes in this scenario.

Analyzing model failures. There has been a growing interest in studying what type of mistakes models make. For instance, works on *failure modes* detection studied how to identify slices of data on which models underperform [67, 232, 275] and, through the use of LMMs, these slices can also be interpreted via natural language [116, 63, 54]. Other works examined the models' mistakes on specific datasets, to understand what prevents them from achieving perfect performance and to provide guidelines for future works. This has been the case for ImageNet [212], where previous studies discovered problems linked to spurious correlations [182, 231], fine-grained discrimination [249, 197], but also labeling itself [15]. Our work is similar to this latter trend, as we want to investigate what type of mistakes LMMs make when classifying images in the OW. We aim for our findings to serve as a foundation for future research focused on improving the performance of LMMs in this challenging task.

Chapter 4

Unsupervised evaluation

This chapter investigates the emerging paradigm of unsupervised evaluation, where the goal is to assess model performance without relying on ground-truth annotations. While traditional evaluation practices rely on labeled benchmarks and pre-defined metrics, the growing scale and generality of models—particularly Large Multimodal Models (LMMs)—calls for more flexible and autonomous forms of evaluation. In this context, we introduce APEx, a framework designed to automate the benchmarking process by leveraging Large Language Models (LLMs) to plan, execute, and interpret model evaluations. By removing the need for manual labels and fixed evaluation pipelines, APEx explores a new direction for assessing complex models in open-ended, dynamic settings.

4.1 Benchmarking of Large Multimodal Models

4.1.1 The problem

The more powerful machine learning models become, the greater the community's interest in testing their capabilities and limitations. Large Multimodal Models (LMMs)[140, 131] have been investigated by several studies with extensive analysis of their strengths and weaknesses [246, 282, 98]. Recent works [30, 300] typically adhere to a common workflow: (i) selecting models for testing, (ii) devising specialized benchmarks tailored to address specific research questions (which may entail data collection and annotation), and (iii) evaluating the models on these benchmarks, analyzing the results, and drawing conclusions.

Let us examine a simple example. Consider the question: *"Do LMMs understand colors?"* Answering this question requires testing one or multiple models on color recognition tasks. To perform this experimental evaluation, various considerations can arise.

How many colors should be included in the test? How complex should the color palette be? If a model shows proficiency with a specific set of colors, should we conclude the test or continue with further examination? In other words, performing such experimental validation requires the researchers to perform a great deal of manual work: designing the benchmark, testing the models, analyzing the results, and determining whether the depth of the analyses is sufficient to draw conclusions. This process is not only tedious but also complex, particularly in designing benchmarks, as it requires significant domain expertise. Additionally, data collection and curation can be notoriously expensive, and the protocol design and result analyses can be bound to subjectivity. These considerations lead us to investigate a framework capable of autonomously designing and executing a wide range of experiments aimed at evaluating the capabilities of existing LMMs.

We propose to address this question following the principles of programming visual tools via Large Language Models (LLMs) [244]. Our framework, named Automatic Programming of Experiments (APEx), redacts a scientific report to answer any user-specified inquiry by first instantiating a structured empty report and then iteratively improving it through a series of experiments. Specifically, an LLM processes the user request and: (i) generates a suitable benchmark via image retrieval, generation, or augmentation; (ii) deploys models to test; (iii) conducts the experiment; (iv) collects and analyzes the results. The findings are incorporated into the report, which is then fed back to the LLM to determine whether the information is sufficient to answer the initial inquiry. If not, APEx creates a new benchmark and runs more experiments. This iterative process continues until the LLM judges the report comprehensive enough to address the user's question. We evaluate APEx with a wide range of queries to demonstrate its flexibility in designing benchmarks and automatically conducting experiments, reaching conclusions that are valid and insightful. To demonstrate that APEx actually produces reliable reports, we show that it can successfully lead to the core findings of existing manually engineered benchmark studies [246]. Moreover, APEx can flexibly handle queries of various granularity, allowing users to thoroughly explore the strengths and weaknesses of existing LMMs. Finally, the modularity of APEx enables easy inclusion of new tools and capabilities in the future, widening its application scope.

To summarize, this work makes the following contributions:

- We introduce APEx, the first automated benchmark framework to test various capabilities of LMMs at user request;
- APEx automates benchmark design, experiments execution, and results analysis.

Its modular design makes it extensible to incorporate other tools and functionalities;

- We show that APEx outputs valid and comprehensive reports, being able to reproduce findings of previous studies.
- APEx facilitates the discovery of novel strengths and weaknesses of existing LMMs by addressing arbitrary queries at different granularity.

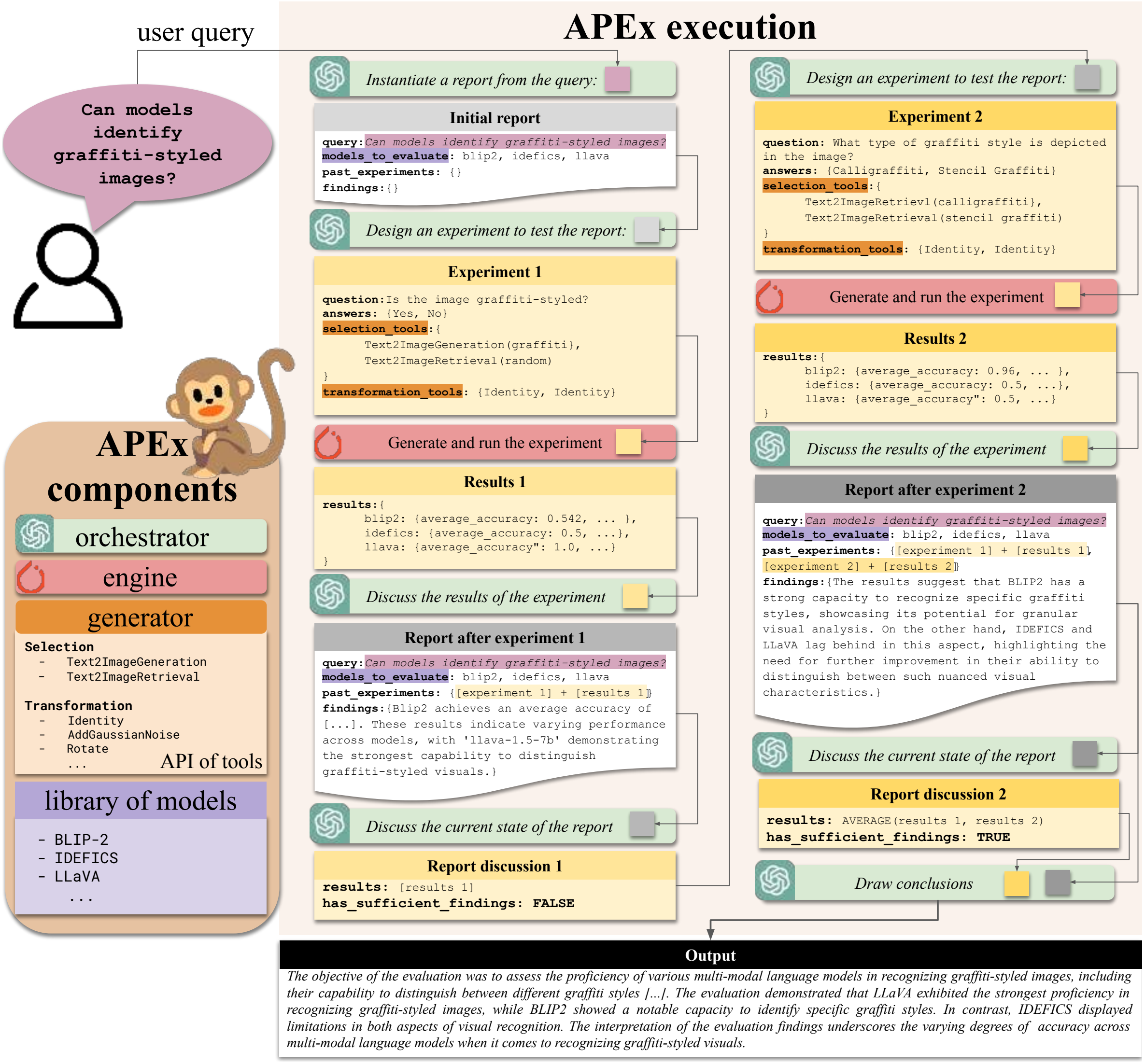
4.1.2 The proposed approach

In this section, we present our APEx framework that automatically conducts experiments on different LMMs to answer a given user inquiry. We refer to query as the text prompt in a question format that contains a hypothesis the user seeks to test. The goal of APEx is to provide an answer a in natural language that addresses the user query through experimental evidence.

APEx is a modular system containing multiple components supported by a set of tools, as shown in fig. 4.1. At its core, there is the orchestrator module, enabled by an LLM that reasons over the available tool-set to define experiments and analyze results. The generator module is equipped with an API (tools) that can perform image retrieval, generation, and transformation to prepare the data for executing benchmark experiments. The library module contains a list of pre-trained LMMs, and finally, the engine module conducts the experiment with the compute resource, possibly a GPU. Note that the modular structure of APEx is independent of the particular choices of each sub-module. Thus, its capabilities can be extended as more powerful LMMs, LLMs, generators, and tools become available. For example, given a user query (*e.g.*, *"Is BLIP-2 robust to left rotations?"*), APEx selects the appropriate tools by compiling an experiment configuration, runs the experiment, and analyzes the results to draw conclusions. The experiments are conducted progressively and the evidence is consolidated into a *report* that holds the status of the experimental campaign. The automatic benchmarking session terminates when the produced results are sufficiently conclusive to answer the initial query. In the following, we detail the components of APEx.

Initialization

Given a query, APEx is initialized providing specific definitions for the library of testable models, for the benchmark generator tools, and for the LLM used in the orchestrator.


user query
Can models identify graffiti-styled images?
APEx components
orchestrator
engine
generator
Selection
- Text2ImageGeneration
- Text2ImageRetrieval
Transformation
- Identity
- AddGaussianNoise
- Rotate
...
API of tools
library of models
- BLIP-2
- IDEFICS
- LLaVA
...
APEx execution
Instantiate a report from the query:
Initial report
Design an experiment to test the report:
Experiment 1
Generate and run the experiment
Results 1
Discuss the results of the experiment
Report after experiment 1
Discuss the current state of the report
Report discussion 1
Design an experiment to test the report:
Experiment 2
Generate and run the experiment
Results 2
Discuss the results of the experiment
Report after experiment 2
Discuss the current state of the report
Report discussion 2
Draw conclusions
Output

Initial report

```
query:Can models identify graffiti-styled images?
models_to_evaluate: blip2, idefics, llava
past_experiments: {}
findings:{}
```

Experiment 1

```
question:Is the image graffiti-styled?
answers: {Yes, No}
selection_tools:{
      Text2ImageGeneration(graffiti},
      Text2ImageRetrieval(random)
}
transformation_tools: {Identity, Identity}
```

Results 1

```
results:{
      blip2: {average_accuracy: 0.542, ... },
      idefics: {average_accuracy: 0.5, ...},
      llava: {average_accuracy": 1.0, ...}
}
```

Report after experiment 1

```
query:Can models identify graffiti-styled images?
models_to_evaluate: blip2, idefics, llava
past_experiments: {[experiment 1] + [results 1]}
findings:{Blip2 achieves an average accuracy of
[...]. These results indicate varying performance
across models, with 'llava-1.5-7b' demonstrating
the strongest capability to distinguish
graffiti-styled visuals.}
```

Report discussion 1

```
results: [results 1]
has_sufficient_findings: FALSE
```

Experiment 2

```
question: What type of graffiti style is depicted
in the image?
answers: {Calligraffiti, Stencil Graffiti}
selection_tools:{
      Text2ImageRetrievl(calligraffiti},
      Text2ImageRetrieval(stencil graffiti)
}
transformation_tools: {Identity, Identity}
```

Results 2

```
results:{
      blip2: {average_accuracy: 0.96, ... },
      idefics: {average_accuracy: 0.5, ...},
      llava: {average_accuracy": 0.5, ...}
}
```

Report after experiment 2

```
query:Can models identify graffiti-styled images?
models_to_evaluate: blip2, idefics, llava
past_experiments: {[experiment 1] + [results 1],
[experiment 2] + [results 2]}
findings:{The results suggest that BLIP2 has a
strong capacity to recognize specific graffiti
styles, showcasing its potential for granular
visual analysis. On the other hand, IDEFICS and
LLaVA lag behind in this aspect, highlighting the
need for further improvement in their ability to
distinguish between such nuanced visual
characteristics.}
```

Report discussion 2

```
results: AVERAGE(results 1, results 2)
has_sufficient_findings: TRUE
```

Output

The objective of the evaluation was to assess the proficiency of various multi-modal language models in recognizing graffiti-styled images, including their capability to distinguish between different graffiti styles [...]. The evaluation demonstrated that LLaVA exhibited the strongest proficiency in recognizing graffiti-styled images, while BLIP2 showed a notable capacity to identify specific graffiti styles. In contrast, IDEFICS displayed limitations in both aspects of visual recognition. The interpretation of the evaluation findings underscores the varying degrees of accuracy across multi-modal language models when it comes to recognizing graffiti-styled visuals.

Figure 4.1: Our automatic benchmarking tool has four components: an orchestrator for reasoning, an engine for function execution, a benchmark generator containing image selection and manipulation tools, and a library of LMMs. Given a user query, the orchestrator instantiates a report containing the query and the LMMs to be tested. Then orchestrator receives the report and specifies a first experiment to be executed. The relative benchmark is generated and executed by the engine, with the results collected, discussed by the orchestrator and added to the report. The orchestrator repeats the experimentation loop until it is deemed sufficient to answer the query. In that case, the orchestrator summarizes the report, returning its findings.

When receiving an input query query from the user, the first step of APEx is to instantiate an empty report as a JSON formatted document. The report will contain a snapshot of the request, the performed experiments, and their results. In practice, it serves as a read-write buffer for the orchestrator, driving the decisions regarding what to test and when to halt. We initialize the report by feeding the query to the orchestrator, *i.e.*, $\text{report} = \text{orchestrator}\left([\text{prompt}_S, \text{prompt}_I, \text{query}]\right)$, with $[\cdot]$ being the concatenation operation, prompt_I being the specific initialization prompt, and prompt_S the system prompt containing the task specification, the models' library, and the benchmark generator tools. Both library and the generator are specified with their documentation in the form of docstrings. Note that we do not use in-context learning [20] (*i.e.*, providing few program demonstrations as in [87, 238]), but directly use the docstrings as input. This removes the need for providing good quality demonstrations, being scalable *w.r.t.* the number of tools [201].

The initialized report contains only the specific query and the set of LMMs to test (*i.e.*, *"models_to_evaluate"* in fig. 4.1). These models will be the ones used in the following iterative experimentation. We may target all models for generic queries (*e.g.*, *"Can models...?"*) or specific models for more detailed ones (*e.g.*, BLIP-2 if the query is *"Can BLIP-2...?"*).

Iterative experimentation

The core of APEx is the experimentation loop. In this loop, the orchestrator takes as input the query, the history of experiments and results (*i.e.*, the report), and formulates new experiments if needed. In particular, it considers the question that should be answered with the current experimental results and then generates a suitable benchmark, testing the selected models on it. Specifically, we define the experiment by feeding the LLM with the report and the user query, *i.e.*,

$$\text{experiment} = \text{orchestrator}\left([\text{prompt}_S, \text{prompt}_E, \text{report}]\right),$$

with prompt_E being the specific prompt. The experiment contains two parts: the goal definition and the benchmark dataset generation. In the following, we describe each part in detail as well as the experiment execution.

Goal definition. This part defines the goal of the experiment. It is composed of two elements: a question we want the LMM to answer (*e.g.*, *"Is the image blue?"*, not to be confused with the query) and the set of possible answer choices (*e.g.*, "{*yes*, *no*}"). Note that all experiments are structured in a VQA format, where the LMM receives the question and the input image x. We measure the performance using answer ranking [158] for evaluating LMMs with multiple-choice questions (*i.e.*, computing the likelihood that the model generates a specific answer among the list). We allow the LMMs to abstain from answering by adding *Unknown* as an option for the answer.

Benchmark dataset generation. Given an experiment, APEx can generate a dataset $\mathcal{D}$ for testing. Specifically, $\mathcal{D}$ should be formed with triplets

$$\mathcal{D} = \{(x_i, \text{question}, \text{choice}_i)\}_{i=1}^{N},$$

where N is the size of $\mathcal{D}$, choice_i is the ground-truth answer for the given pair x_i, question. Within experiment we have the specific tools_E needed to generate the dataset.

Since the ground-truth label is a function of the tools used, we associate a set of selection and transformation tools to each answer in choice. This is a consequence of the structured JSON output, allowing for easy matching between the elements of each field. Note that the current set of tools_E comprises both image collection and generation instructions (*e.g.*, retrieval from existing datasets [212], Stable Diffusion [211]) and the transformations to be applied (*e.g.*, rotation, flip). Note that the transformations are represented by Python code that the model can directly execute by outputting an interpretable function call, *e.g.*, *src.tools.transform.OverlayColor([255, 0, 0])*. When the engine produces the dataset, we evaluate these calls to augment or generate the data.

Experiment execution. After the benchmark generation, all the components needed to perform the experiments are available. Specifically, the engine tests each of the LMMs models on $\mathcal{D}$, evaluating their performance. The output of this engine execution is a new set of results_E.

Reporting and conclusion

After each experimental loop, we obtain a new result set of results_E. APEx expands the results and asks the orchestrator to discuss them, obtaining the findings $\text{findings}_E = \text{orchestrator}\left([\text{prompt}_F, \text{experiment}, \text{results}]\right)$, where prompt_F is the specific prompt for extracting the findings. The findings, in natural language, help the orchestrator in performing reasoning and draw conclusions beyond the quantitative metrics in results_E.

Both the results and the findings are appended at the end of the current report, updating its status *i.e.*, $\text{report} \leftarrow [\text{report}, \text{results}_E, \text{findings}_E]$. The updated report contains all the evidence collected so far to answer the user query. orchestrator then judges whether the updated report is sufficient to answer the user query, filling a boolean variable sufficiency = orchestrator ([prompt$_B$, report]), with prompt$_B$ being the prompt for obtaining the boolean value. In case the orchestrator deems the report *not* sufficient to answer the query, the report is fed back to the orchestrator to continue the experimental loop, enriching the report with the results of a new experiment.

Once the orchestrator deems the report sufficient to answer the query, APEx will use the orchestrator to analyze the report and draw the final conclusions of the experiment, *i.e.*, conclusions = orchestrator ([prompt$_C$, report]), with prompt$_C$ the specific prompt. Note that conclusions can be customized based on the particular user needs, *e.g.*, they can be succinct, reporting only the answer to the query, or expanded with a summary of the experiments, their results, and possible directions. Once conclusions are drawn, APEx provides them as output to the user.

To demonstrate the final artifacts of the execution of APEx, we provide two examples of qualitative results. In particular in fig. 4.2 we show the intermediate reports and the output generated with our approach.

Implementation details

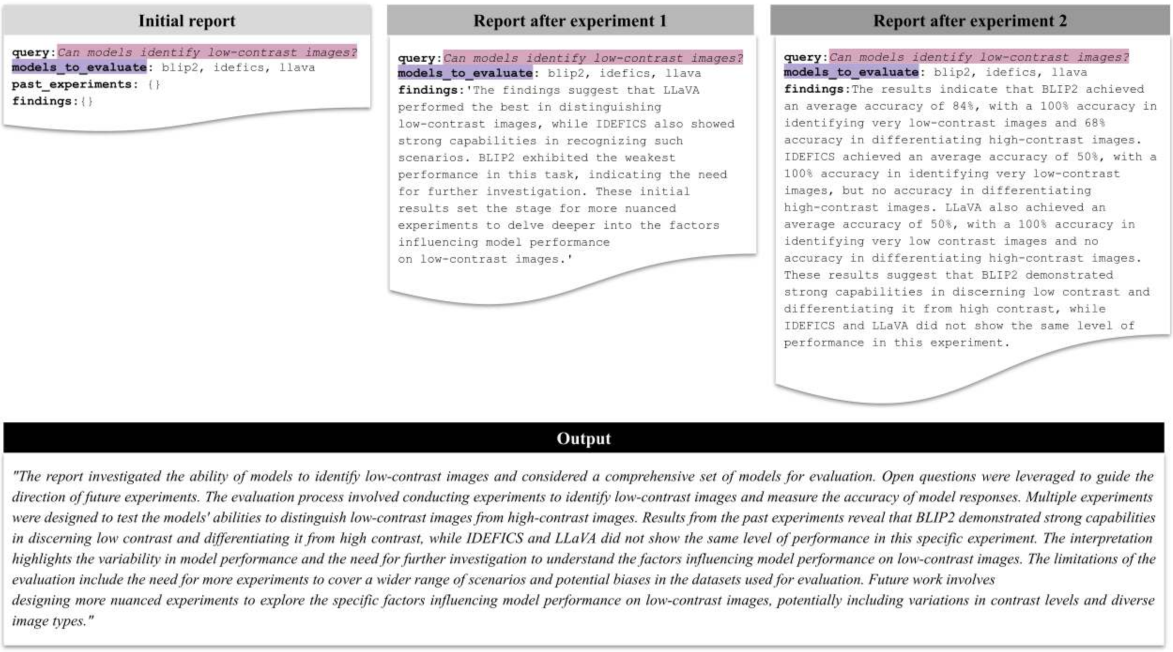
Orchestrator. The core component of APEx is the orchestrator, implemented as an LLM. We used gpt-3.5-turbo [190] for its efficiency and widespread use. To instruct the model to produce structured outputs, we use `function calling`[1], meaning the LLM outputs adhere to the structured JSON format rather than being unrestricted text, ensuring easy mapping of the model outputs to functions/tools available in APEx. In cases where the orchestrator's output is invalid, *e.g.*, it lacks JSON fields, or it requests non-available tools, APEx informs the orchestrator of the exception message (within the prompt), using this information as prior to *self-heal* the previous output. In the case of complex healing (*e.g.*, multiple sequential failures), a new experiment is designed from scratch. To avoid infinite loops, we set the maximum number of self-healing retries to 3 before regenerating the experiment from scratch. Moreover, in all of our evaluations, we limit the maximum number of experiments the orchestrator can design to 5.

Benchmark generation. As stated in section 4.1.2 the data benchmark generation step in APEx comprises both image generation tools and tools for retrieval. For generation, we use Stable Diffusion XL Turbo [199] for its efficiency. For retrieval, we use the ImageNet dataset [212] due to the large number of classes. In both cases, in APEx, it is possible to define an image type (*e.g.*, photo, oil painting) and a class name (*e.g.*, Siamese cat, sofa). The prompts corresponding to the two available tools are *Text-*

---

[1] `https://platform.openai.com/docs/guides/function-calling`

Example 1: *“Can models identify low-contrast images?”*



Initial report

```
query:Can models identify low-contrast images?
models_to_evaluate: blip2, idefics, llava
past_experiments: {}
findings:{}
```

Report after experiment 1

```
query:Can models identify low-contrast images?
models_to_evaluate: blip2, idefics, llava
findings:'The findings suggest that LLaVA
performed the best in distinguishing
low-contrast images, while IDEFICS also showed
strong capabilities in recognizing such
scenarios. BLIP2 exhibited the weakest
performance in this task, indicating the need
for further investigation. These initial
results set the stage for more nuanced
experiments to delve deeper into the factors
influencing model performance
on low-contrast images.'
```

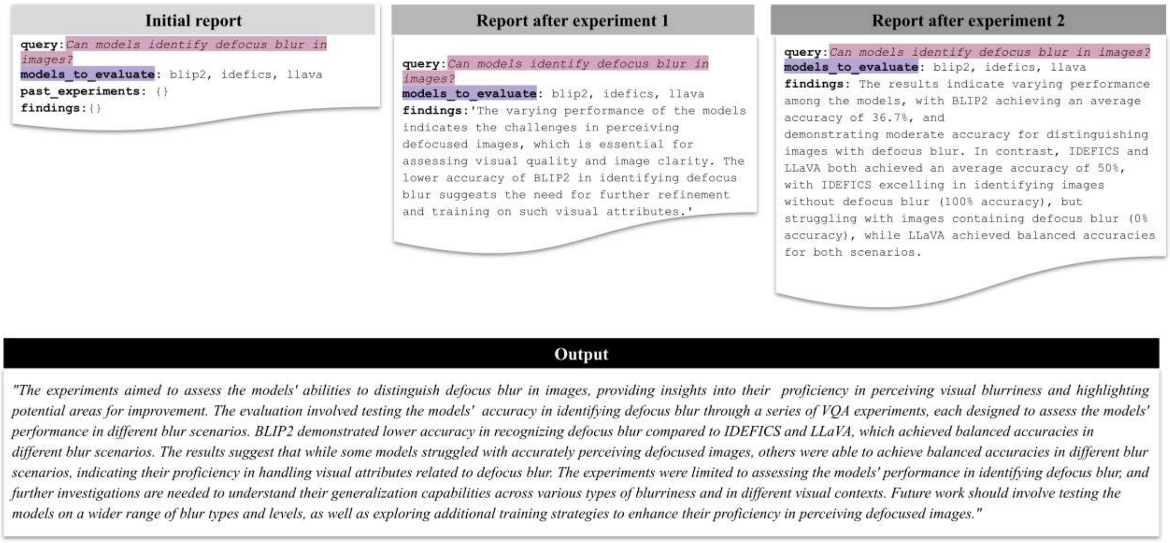
Report after experiment 2

```
query:Can models identify low-contrast images?
models_to_evaluate: blip2, idefics, llava
findings:The results indicate that BLIP2 achieved
an average accuracy of 84%, with a 100% accuracy in
identifying very low-contrast images and 68%
accuracy in differentiating high-contrast images.
IDEFICS achieved an average accuracy of 50%, with a
100% accuracy in identifying very low-contrast
images, but no accuracy in differentiating
high-contrast images. LLaVA also achieved an
average accuracy of 50%, with a 100% accuracy in
identifying very low contrast images and no
accuracy in differentiating high-contrast images.
These results suggest that BLIP2 demonstrated
strong capabilities in discerning low contrast and
differentiating it from high contrast, while
IDEFICS and LLaVA did not show the same level of
performance in this experiment.
```

Output

"The report investigated the ability of models to identify low-contrast images and considered a comprehensive set of models for evaluation. Open questions were leveraged to guide the direction of future experiments. The evaluation process involved conducting experiments to identify low-contrast images and measure the accuracy of model responses. Multiple experiments were designed to test the models' abilities to distinguish low-contrast images from high-contrast images. Results from the past experiments reveal that BLIP2 demonstrated strong capabilities in discerning low contrast and differentiating it from high contrast, while IDEFICS and LLaVA did not show the same level of performance in this specific experiment. The interpretation highlights the variability in model performance and the need for further investigation to understand the factors influencing model performance on low-contrast images. The limitations of the evaluation include the need for more experiments to cover a wider range of scenarios and potential biases in the datasets used for evaluation. Future work involves designing more nuanced experiments to explore the specific factors influencing model performance on low-contrast images, potentially including variations in contrast levels and diverse image types."

Example 2: *“Can models identify defocus blur in images?”*



Initial report

```
query:Can models identify defocus blur in
images?
models_to_evaluate: blip2, idefics, llava
past_experiments: {}
findings:{}
```

Report after experiment 1

```
query:Can models identify defocus blur in
images?
models_to_evaluate: blip2, idefics, llava
findings:'The varying performance of the models
indicates the challenges in perceiving
defocused images, which is essential for
assessing visual quality and image clarity. The
lower accuracy of BLIP2 in identifying defocus
blur suggests the need for further refinement
and training on such visual attributes.'
```

Report after experiment 2

```
query:Can models identify defocus blur in images?
models_to_evaluate: blip2, idefics, llava
findings: The results indicate varying performance
among the models, with BLIP2 achieving an average
accuracy of 36.7%, and
demonstrating moderate accuracy for distinguishing
images with defocus blur. In contrast, IDEFICS and
LLaVA both achieved an average accuracy of 50%,
with IDEFICS excelling in identifying images
without defocus blur (100% accuracy), but
struggling with images containing defocus blur (0%
accuracy), while LLaVA achieved balanced accuracies
for both scenarios.
```

Output

"The experiments aimed to assess the models' abilities to distinguish defocus blur in images, providing insights into their proficiency in perceiving visual blurriness and highlighting potential areas for improvement. The evaluation involved testing the models' accuracy in identifying defocus blur through a series of VQA experiments, each designed to assess the models' performance in different blur scenarios. BLIP2 demonstrated lower accuracy in recognizing defocus blur compared to IDEFICS and LLaVA, which achieved balanced accuracies in different blur scenarios. The results suggest that while some models struggled with accurately perceiving defocused images, others were able to achieve balanced accuracies in different blur scenarios, indicating their proficiency in handling visual attributes related to defocus blur. The experiments were limited to assessing the models' performance in identifying defocus blur, and further investigations are needed to understand their generalization capabilities across various types of blurriness and in different visual contexts. Future work should involve testing the models on a wider range of blur types and levels, as well as exploring additional training strategies to enhance their proficiency in perceiving defocused images."

Figure 4.2: Examples of intermediate reports and final output generated with APEx in the case of two different queries. APEx progressively expands the reports as new experiments are conducted and new results are collected.

ToImageGeneration and *TextToImageRetrieval*, passed to APEx in form of docstrings (see, *e.g.*, fig. 4.1, Experiment box). To perform experiments with broader categories, APEx uses the metaclasses from ImageNet-X [102], clustering the 1000 classes into 17 groups (*e.g.*, plants, fruits). When retrieval is impossible, *e.g.*, the requested class is not available in the dataset, APEx automatically resorts to generation. We also use the special keyword *random* to denote sampling a semantic class in the dataset. This is used whenever the experiment does not require any specific class, and all of them can be considered for the benchmark. Regarding the transformation tools, *i.e.*, tools to manipulate the input samples for visual variations, we follow the design as in [246]. These transformations can be roughly split into geometric, pixel, semantic, and style transforms. Additional details are provided in the Appendix.

LMMs evaluation. While in principle APEx supports a large variety of LMMs, in our implementation we demonstrate its capabilities using three widely used auto-regressively trained VLMs (referred as LMMs in [246]): BLIP-2 [140], IDEFICS [131], and LLaVA [139]. All the models are downloaded from HuggingFace[2] and are extended to support answer ranking. We focus on small models, using `blip2-opt-2.7b`, `idefics-9b-instruct`, and `llava-1.5-7b`. We quantize all the models to 8 bits and use a single NVIDIA RTX A4000 for the experiments. About the quantitative metrics in results$_E$, we evaluate the methods with average accuracy and class-wise accuracy. In cases when the models abstain from answering, we compute the abstention rate. The quantitative metrics are reported in a key-value format organized by evaluated models (see fig. 4.1, Results box).

List of tools. The list of transforms implemented in APEx are `Add Gaussian Noise`, `Add JPEG Compression`, `Apply CutMix`, `Apply MixUp`, `Change Brightness`, `Change Contrast`, `Crop Random Shuffle And Recompose`, `Defocus Blur Image`, `Edit Image Style`, `Edit Image Weather`, `Flip Image`, `Identity`, `Overlay Color`, `Paste Generated Object At Random Position`, `Paste Geometric Shape At Random Position`, `Paste Text At Random Position`, `Rotate Image`, and `Zoom At Random Position`. We report the full docstrings of all the tools in appendix C. We devise the transformation tools to produce data types defined in [246]. All data types are covered, but not all are explicitly defined with a tool (*e.g.*, we do not design specific tools for style transfer to cartoons, but provide a generic tool for style editing).

---

[2] `https://huggingface.co`

4.1.3 Experimental results

Our tool generates free-form text output, summarizing findings related to the input query. An illustrative example of qualitative results obtained using APEx is depicted in fig. 4.1. In addition, we quantitatively evaluate our framework as described below. As an important feature is the possibility to reproduce conclusions obtained by manually designed benchmark, we first demonstrate that APEx can reproduce the core conclusions of [246] regarding LMMs on data type identification, *i.e.*, on recognizing image alterations (*e.g.*, *Can models identify vertical flip/Gaussian noise/cartoon-styled images?*). Second, we showcase the flexibility of APEx in handling queries of different nature and with different granularity. In particular, we present the analyses of APEx for identifying groups of transformations, for classifying semantic groups (*e.g.*, dog breeds, plants, fruits) and recognition with the presence/absence of data type groups (*e.g.*, *"Can models recognize domestic animals with style-type transformations?"*).

Finding reproducibility analysis: data type identification

We compare the conclusions obtained by APEx to the ones obtained by the manually-designed benchmark presented in [246]. Specifically, the authors created and curated two datasets (natural and synthetic) of images featuring a single animal, spanning 27 data-type transformations covering 4 broad categories: geometric (*e.g.*, left-rotation), pixel (*e.g.*, applying Gaussian noise), style (*e.g.*, creating a cartoon-version), and semantic (*e.g.*, replacing a single animal with multiple animals). They considered both contrastively-trained VLMs, *e.g.*, CLIP [204]), and auto-regressively trained VLMs, *e.g.*, LLaVA (referred to as LMMs), and asks them to identify the transformation applied on the image. For fairness with contrastive ones, they evaluate auto-regressive VLMs using answer ranking [59, 158]. Their key finding is that VLMs excel in recognizing semantic content but fail to acquire an understanding of visual data-types.

To be comparable with [246], we prompt APEx with each user query per data type and let APEx deploy the models, define the experiments, generate the data, test models, and analyze results. The prompt template for formulating the query follows the template: *"Can models identify {data_type} in images?"*. For some special cases, we rephrased the query to reduce ambiguities (*e.g.*, we replace *"Can models identify multi-different in images?"* with *"Can models identify if multiple objects of different classes are in the image?"*. We report all the used prompts in appendix C.

The evaluation protocol of [246] covers all the data types together, while we test each data type independently. As our performance metrics (average accuracy) have

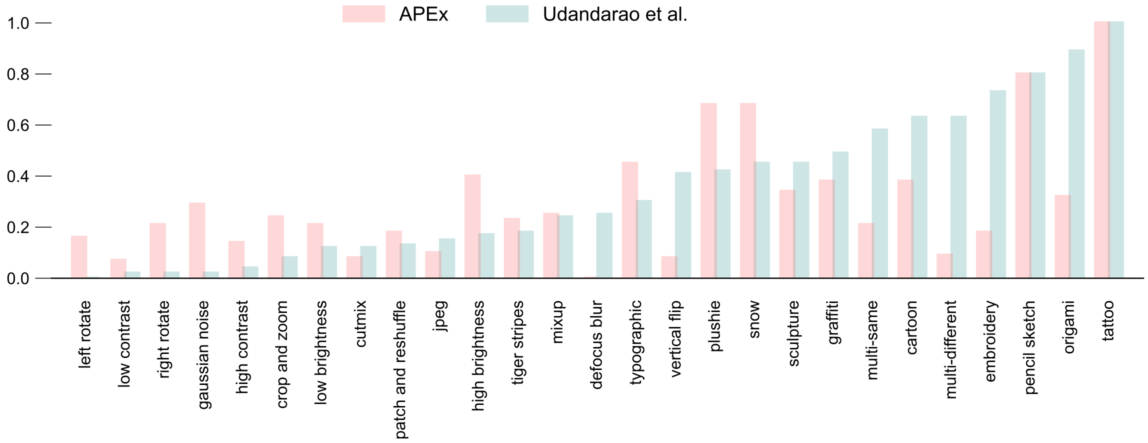


Figure 4.3: Summary of the normalized results (vertical axis) achieved by the models across the 27 data type identification tasks, in comparison to those obtained in [246]. The values equal the min-max normalized performance across the set of experiments designed by APEx.

higher numerical values, we report the normalized metrics in fig. 4.3, where the data types at the horizontal axis are listed in an ascending order based on the results in [246].

Despite their differences in numerical values per data type, the analysis of APEx leads to a similar trend drawn from the manually designed and curated benchmark, especially when we organize the analyses by data type groups, *i.e.*, geometric, pixel, style, and semantic [246] (as shown in table 4.1). For example, both APEx and [246] identify that LMMs perform better when recognizing tattoos and embroidery rather than lower-level transformations (*e.g.*, low/high contrast, jpeg). More importantly, we reach the same conclusion that LMMs are generally better in style/semantic than pixel/geometric understanding. We report the complete list of experiments APEx generated for each of the 27 data types, regarding the generated questions, the answers, and the tools, in appendix B.

Flexibility analyses

Coarse *query* handling. We further evaluate APEx in handling query of a coarse granularity. Instead of focusing on specific data type, we form the query using the group of transformations as in [246]: geometric, pixel, semantic and style. We form general queries: *"Can models identify {transformation_group} transformations?"*

This study introduces multiple challenges. First, APEx has to reason on a higher level, identifying which subset of tools is relevant to answer the query, *e.g.*, for the geometric family, it should focus on geometric transformations. Second, it must test

	APEx		[246]
Rank	**Avg. types**	**Groups**	
1	Style	Style	Style
2	Semantic	Semantic	Semantic
3	Pixel	Pixel	Pixel
4	Geometric	Geometric	Geometric

Table 4.1: Ranking of data type group recognition performance from the best (top) to worst (bottom). APEx achieves the same ranking of [246] both when testing each data type independently and aggregating metrics by their group (Avg. types), and when directly querying for the group understanding (Groups).

models on multiple axes, devising experiments that regard not only the presence/absence of a data type but also the exact identification against alternatives of the same group, in a hierarchical fashion. Similarly to the findings in section 4.1.3, results in table 4.1 confirm the findings of [246] and those in section 4.1.3 where, despite different numbers in absolute terms, the trend is preserved. The table shows that APEx, just like in [246], geometric transformations (0.5 accuracy for APEx, 0.1 informedness for [246]) are harder to recognize *w.r.t.* style manipulations (0.78 accuracy for APEx, 0.46 informedness for [246]). Table B.2 in appendix B, reports the quantitative analyses obtained via APEx and the type of experiments conducted. Interestingly, APEx tends to experiment with multiple data types in a category (*e.g.*, compression, noise, gray-scale for pixel-level ones). When the objective is less clear (*e.g.*, style-type), APEx explores various comparisons with different levels of granularity, *e.g.*, from varying both semantic and style (dog oil paintings *vs* bird pastel) to fixing the former and varying the latter (baroque structure *vs* art deco structure).

Data classes recognition. To further assess the potential of APEx, we challenge it on fine-grained recognition experiments, requiring precise semantic modeling. Specifically, we ask APEx to assess LMMs' performance on nine semantic groups of classes, *i.e.*, dog breeds, bird types, vehicles, furniture, building types, fruits, household objects, domestic animals, and plants.[3] An example query is *"Can models recognize dog breeds?"*.

Figure 4.4 shows the average class-wise accuracy of the different models of each group. The experiments show that BLIP-2 generally outperforms the other models, in particular on vehicles, buildings, and household objects. For animals, the performance varies on the species, with LLaVA being the best on birds and dogs, while BLIP-2 outperforming it on the coarser domestic category. We can also observe interesting

---

[3] We select these groups from the 17 of ImageNet-X [102], removing too generic ones (*e.g.*, other, commodity) and merging similar concepts (*e.g.*, vehicle and wheeled vehicle).

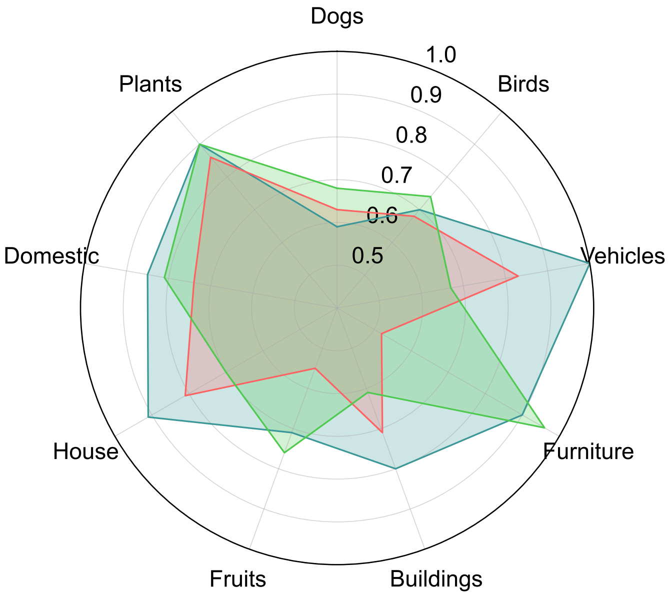


Figure 4.4: Data classes recognition accuracy averaged over experiments of APEx for BLIP-2, IDEFICS, and LLaVA on the nine recognition tasks.

phenomenon. For instance, for birds, no model can distinguish blue jay from cardinal, and goldfinch from canary. The only model that can distinguish cathedrals from skyscrapers is BLIP-2, achieving 0.88 accuracy. On the other hand, it cannot separate churches from museums, achieving 0.56, with the others achieving perfect scores. When asking about fruit colors (*i.e.*, showing apples and bananas and asking whether they are red or yellow), LLaVA achieves random chance performance, while BLIP-2 reaches perfect score. In addition, we report the automatically generated quantitative results, in table B.3 in appendix B. The table shows that APEx designs interesting types of experiments, comparing classes with extremely close similarity (*e.g.*, husky *vs* malamute), sometimes going from coarse comparisons to finer ones (*e.g.*, for domestic animals, first cat *vs* dog, then Persian *vs* Siamese), and even asking for attributes (in buildings, red *vs* blue).

Since APEx can process arbitrary queries, even those combining multiple requests, we perform a final experiment by testing LMMs, combining semantics with data types. Specifically, we consider the 9 semantic groups and the 4 data-types groups of the previous experiment and combine them, *i.e.*, forming queries of the type: *"Can models recognize [class] under [data type]?"*. We report the results in table 4.2. Overall, we see a similar trend of table 4.1, with performance improving as we move from geometric to

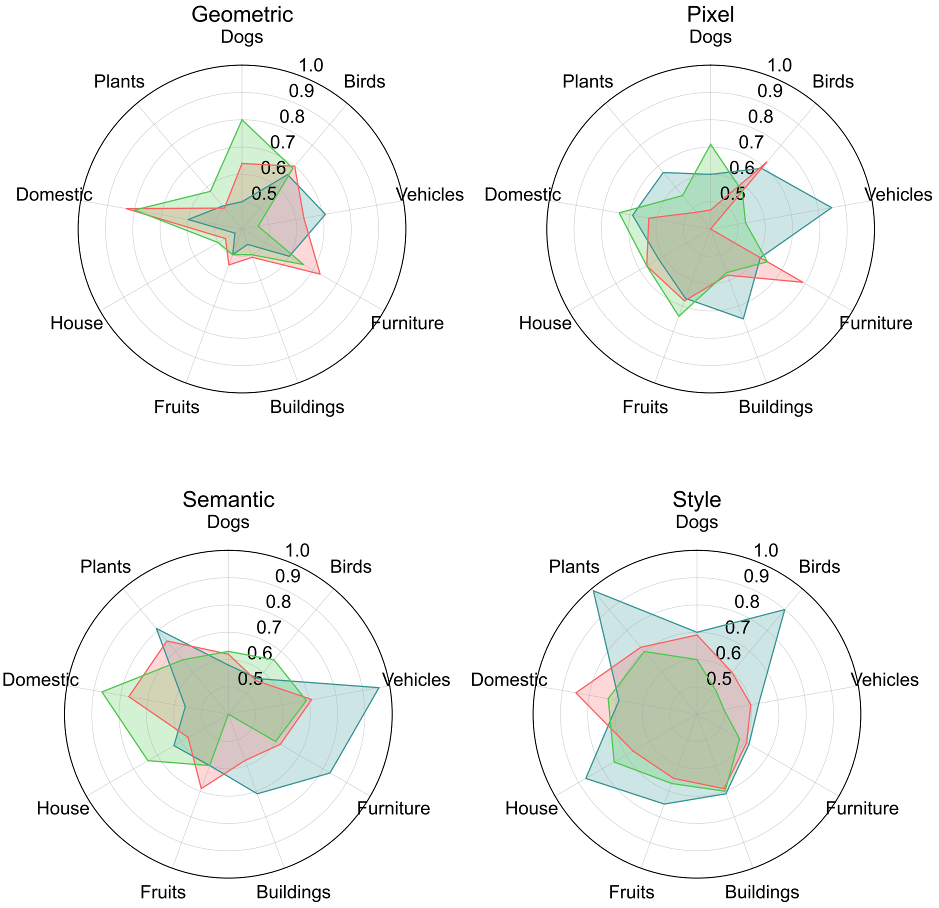


Table 4.2: Summary of the average accuracy achieved by BLIP-2, IDEFICS, and LLaVA across the nine recognition tasks when adding different data type transformations. Accuracy is averaged across the set of experiments designed by APEx.

style transformations. Also, similarly to fig. 4.4, BLIP-2 tends to achieve better results on objects (*e.g.*, vehicles) while LLaVA is better with animals (*e.g.*, dogs). Interestingly, we see different trends based on the data types, with the best performing model often changing across transformations, *e.g.*, fruits have IDEFICS as best for geometric- and semantic-type data, while LLaVA for pixel-type and BLIP-2 for style-type. These patterns are easier to spot with APEx, thanks to its possibility to test various configurations without manual effort.

4.1.4 Related work

Our work is related to previous research on modular machine learning and studies that investigate the capabilities of LMMs. Below, we review the most relevant works.

Visual programming. Composing atomic operations to realize complex functions is a popular approach in Visual Question Answering (VQA) [5]. In this context, modular neural networks [4, 236] decompose questions (*e.g.*, *"How many dogs are in the image?"*) into sub-tasks (*e.g.*, detecting, counting) that can solve the whole query when their execution is chained (*e.g.*, counting after detection). This paradigm has recently evolved with the enhanced reasoning capabilities of LLMs [190]. Specifically, in Natural Language Processing, emerged frameworks augment LLMs with the ability to use external tools [216, 194, 201]. This is usually achieved by specifying the set of tools within the prompt (*e.g.*, as APIs) and providing examples of use cases, performing in-context learning [20]. For vision, VisProg [87] and ViperGPT [238] proposed to use the same paradigm to solve various computer vision tasks. Specifically, they leverage the LLMs to generate programs executed by a suite of pre-trained vision models. Note that the LLM acts as the parser and sub-task decomposer in [4]. Subsequent works focused on two main directions. The first is to improve the visual programming pipeline by, *e.g.*, expanding the set of available tools [223], adding verification steps [172], or self-training [114]. The second direction focuses on applying them for specific tasks, *e.g.*, animal behavior understanding [274], open-vocabulary 3D grounding [279], and text-to-image generation [44]. Concurrently, [221] used a set of tools to automatically interpret neurons of deep networks.

APEx builds on previous works tackling visual tasks via LLM-based program generation [87, 238]. However, in contrast to previous studies, we propose a paradigm shift introducing a novel framework for generating programs for rigorously testing LMMs, rather than solving specific tasks.

Analyzing vision language models. Several previous studies investigated the capabilities and shortcomings of Vision Language Models (VLMs). For instance, [243, 175] studied VLMs on fine-grained differences in textual inputs, showing issues for capturing changes in relationships, objects, and attributes. In [282] the authors further analyzed the impact of words and patch orders, showing that VLMs act as bag-of-words if fine-grained relationships are not considered when forming training batches. Other works studied VLMs biases concepts seen in the training set. Examples are [240] that shows biases in how VLMs associate attributes to objects, [80] that studies shape *vs* texture bias, and [247, 68, 178] that show how frequency of concepts impact zero-shot generalization. Other works focused on testing VLMs for specific tasks such as time/location reasoning [292], egocentric vision [40], information retrieval [34], spatial reasoning [111], or general multimodal understanding [28, 225]. Similarly, studies explored the robustness of VLMs to distribution-shifts [202], corruptions [30], adversarial attacks [300], and missing modalities [174]. It is important to note that most of these works required designing specific benchmarks and it is not uncommon that follow-up works identify and fix benchmark issues of previous work [98].

Differently from these works, we do not focus on a single analysis but provide a general tool for automatically creating benchmarks and inspecting what VLMs (or LMMs), can do. As our software is publicly available, researchers and AI professionals can not only use it for LMMs testing but also contribute to APEx, adding modules for expanding its capabilities.

Chapter 5

Conclusion

This thesis investigated how supervision can be reduced or removed across three core stages of the deep learning pipeline: training, inference, and evaluation. Motivated by the limitations of traditional supervised approaches, *i.e.*, their dependence on large volumes of labeled data, we proposed solutions that explore unsupervised alternatives across various tasks.

In the context of training, we focused on unsupervised fine-tuning, particularly under the Source-Free Unsupervised Domain Adaptation setting. We proposed a self-supervised clustering-based method for Facial Expression Recognition that integrates task knowledge through a robust pseudo-labeling strategy, and a vision-language–guided distillation approach for adapting for video-based Action Recognition. These methods demonstrate that unsupervised fine-tuning can rival traditional supervised or source-dependent adaptation methods, enabling model transfer to new domains with minimal supervision.

In the inference stage, we examined how modern Vision Language Models and Large Multimodal Model can be adapted to perform classification and semantic segmentation tasks without relying on a predefined set of categories. We formalized the Vocabulary-free Image Classification task and proposed a solution based on external vision-language databases, enabling training-free classification in an open-world setting. We extended this framework to Vocabulary-free Semantic Segmentation by enabling pixel-level labeling without predefined vocabularies. Additionally, we evaluated Large Multimodal Models in Open-World classification tasks driven by natural language queries, introducing metrics and analyses to understand their performance under unsupervised conditions.

Lastly, in the evaluation stage, we introduced APEx, a framework for unsupervised benchmarking of Large Multimodal Models. APEx automates the design, execution, and reporting of evaluation experiments based on natural language prompts, eliminating the need for ground-truth annotations. By leveraging the capabilities of Large Language Models and modular tools, APEx redefines evaluation as a dynamic and flexible process, opening new directions for scalable model assessment.

Across all three stages, this thesis highlights how reducing supervision can lead to more adaptive, efficient, and scalable learning systems. The proposed methods and settings challenge traditional assumptions about the necessity of labeled data and help lay the groundwork for future research in label-efficient and autonomous artificial intelligence.

5.1 Future directions

While this thesis advances the understanding and practical implementation of unsupervised methods across training, inference, and evaluation, several open challenges and directions remain for future research.

For unsupervised training, one key challenge is improving the robustness and generality of fine-tuning methods using pseudo-labeling and self-supervised adaptation, particularly in highly dynamic or low-resource domains. Future work could explore tighter integration between vision-language priors and self-supervised signals, leveraging the complementary strengths. Additionally, there is room to investigate adaptive fine-tuning strategies that generalize across multiple target domains or scenarios without supervision, as current approaches often remain tailored to specific applications. Extending these methods to operate in more open, domain-agnostic settings would improve their robustness and applicability. Furthermore, adapting these strategies to work incrementally or in real-time, *e.g.*, through continual learning or test-time adaptation, could enable models to adjust dynamically to novel conditions without explicit retraining.

For unsupervised inference, a promising direction involves addressing the limitations of retrieval-based approaches in Vocabulary-free Image Classification and Vocabulary-free Semantic Segmentation, *i.e.*, those tackled by CaSED and its extensions. These methods are sensitive to the quality and content of the external database, as well as to issues of semantic granularity and label inconsistency. Future work could explore dynamically adapting the retrieval space to the input instance, incorporating domain-specific information or user intent, and developing more robust mechanisms for maintaining consistency across predictions.

Beyond retrieval-based methods, future research on unsupervised inference with Large Multimodal Models presents another rich area of exploration. These models offer flexible, natural language–driven interaction, but their performance in Open-World classification settings remains limited by ambiguity in prompt interpretation and lack of control over semantic specificity. Future directions could focus on designing better prompting strategies, integrating memory or self-consistency modules, and developing evaluation-aware inference protocols that can adapt responses based on context and task intent.

In unsupervised evaluation, the increasing use of Large Language Models for experiment design opens new possibilities and challenges. While APEx shows the feasibility of automatic benchmarking, it is still limited by the reasoning capabilities of current language models. Future work may focus on improving the fidelity and reliability of automated evaluation by combining LLMs with structured knowledge bases or verification modules. A notable limitation of APEx lies in the lack of control over the data generation and task formulation processes, which can lead to inconsistencies, ambiguous tasks, or low-quality experiment designs. Incorporating interactive elements, *e.g.*, human-in-the-loop validation or feedback, could improve the relevance and trustworthiness of the evaluation. Moreover, expanding the tools and extending benchmarking beyond vision-language tasks could push toward more comprehensive and interpretable evaluations.

Reducing supervision across the entire learning pipeline, *i.e.*, spanning training, inference, and evaluation, presents a compelling path toward more scalable and adaptable machine learning systems. This thesis has shown that it is possible to relax the reliance on labeled data while preserving performance and flexibility across a wide range of tasks and settings. As we move forward, the challenge will be to deepen these approaches and broaden their applicability, while ensuring that models remain reliable, interpretable, and aligned with human goals. Ultimately, striking this balance will be key to enabling more autonomous and robust artificial intelligence.

Bibliography

[1] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2010.

[2] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv*, 2024.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *NeurIPS*, 2022.

[4] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *CVPR*, 2016.

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, 2015.

[6] Mouath Aouayeb, Wassim Hamidouche, Catherine Soladie, Kidiyo Kpalma, and Renaud Seguier. Learning vision transformer with squeeze and excitation for facial expression recognition. *arXiv*, 2021.

[7] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[8] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv*, 2019.

[9] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *NeurIPS*, 2020.

[10] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[11] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv*, 2023.

[12] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv*, 2025.

[13] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 2021.

[14] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, 2015.

[15] Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *arXiv*, 2020.

[16] Andreas Blattmann, Robin Rombach, Kaan Oktay, Jonas Müller, and Björn Ommer. Retrieval-augmented diffusion models. *NeurIPS*, 2022.

[17] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *ICML*, 2022.

[18] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *ECCV*, 2014.

[19] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *ECCV*, 2014.

[20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020.

[21] Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *NeurIPS*, 2019.

[22] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[23] Manuel G Calvo and Lauri Nummenmaa. Perceptual and affective mechanisms in facial expression recognition: An integrative review. *Cognition and Emotion*, 2016.

[24] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

[25] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.

[26] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021.

[27] Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. In *ICML*, 2024.

[28] Liang Chen, Yichi Zhang, Shuhuai Ren, Haozhe Zhao, Zefan Cai, Yuchi Wang, Tianyu Liu, and Baobao Chang. Towards end-to-end embodied decision making with multi-modal large language model: Explorations with gpt4-vision and beyond. In *NeurIPS Workshop*, 2023.

[29] Min-Hung Chen, Zsolt Kira, Ghassan AlRegib, Jaekwon Yoo, Ruxin Chen, and Jian Zheng. Temporal attentive alignment for large-scale video domain adaptation. In *ICCV*, 2019.

[30] Shuo Chen, Jindong Gu, Zhen Han, Yunpu Ma, Philip Torr, and Volker Tresp. Benchmarking robustness of adaptation methods on pre-trained vision-language models. *NeurIPS*, 2024.

[31] Tianshui Chen, Tao Pu, Hefeng Wu, Yuan Xie, Lingbo Liu, and Liang Lin. Cross-domain facial expression recognition: A unified evaluation benchmark and adversarial graph learning. *TPAMI*, 2021.

[32] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[33] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv*, 2022.

[34] Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? In *EMNLP*, 2023.

[35] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020.

[36] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv*, 2024.

[37] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv*, 2024.

[38] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 2024.

[39] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, 2024.

[40] Sijie Cheng, Zhicheng Guo, Jingwen Wu, Kechen Fang, Peng Li, Huaping Liu, and Yang Liu. Can vision-language models think from a first-person perspective? *arXiv*, 2023.

[41] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev.

Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023.

[42] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna.lmsys.org (accessed 14 April 2023)*, 2023.

[43] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *ICML*, 2024.

[44] Jaemin Cho, Abhay Zala, and Mohit Bansal. Visual programming for step-by-step text-to-image generation and evaluation. *NeurIPS*, 2024.

[45] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

[46] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

[47] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *TPAMI*, 2002.

[48] Alessandro Conti, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. Vocabulary-free image classification. *NeurIPS*, 2023.

[49] Alessandro Conti, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. Vocabulary-free image classification. *NeurIPS*, 2024.

[50] Alessandro Conti, Enrico Fini, Paolo Rota, Yiming Wang, Massimiliano Mancini, and Elisa Ricci. Automatic benchmarking of large multimodal models via iterative experiment programming. *arXiv preprint*, 2024.

[51] Alessandro Conti, Massimiliano Mancini, Enrico Fini, Yiming Wang, Paolo Rota, and Elisa Ricci. On large multimodal models as open-world image classifiers. *arXiv preprint*, 2025.

[52] Alessandro Conti, Paolo Rota, Yiming Wang, and Elisa Ricci. Cluster-level pseudo-labelling for source-free cross-domain facial expression recognition. *BMVC*, 2022.

[53] Common Crawl. Common crawl, 2023. Accessed May 17, 2023. `https://commoncrawl.org/`.

[54] Gabriela Csurka, Tyler L Hayes, Diane Larlus, and Riccardo Volpi. What could go wrong? discovering and describing failure modes in computer vision. *arXiv*, 2024.

[55] Yufeng Cui, Lichen Zhao, Feng Liang, Yangguang Li, and Jing Shao. Democratizing contrastive language-image pre-training: A clip benchmark of data, model, and supervision. *arXiv*, 2022.

[56] Victor G Turrisi da Costa, Giacomo Zara, Paolo Rota, Thiago Oliveira-Santos, Nicu Sebe, Vittorio Murino, and Elisa Ricci. Dual-head contrastive domain adaptation for video action recognition. In *WACV*, 2022.

[57] Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *JMLR*, 2022.

[58] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, and Junqi Zhao. Instructblip: Towards general-purpose vision-language models with instruction tuning. *NeurIPS*, 2023.

[59] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *NeurIPS*, 2024.

[60] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[61] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *CVPR*, 2021.

[62] Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. Redcaps: Web-curated image-text data created by the people, for the people. *arXiv*, 2021.

[63] Lisa Dunlap, Yuhui Zhang, Xiaohan Wang, Ruiqi Zhong, Trevor Darrell, Jacob Steinhardt, Joseph E Gonzalez, and Serena Yeung-Levy. Describing differences in image sets with natural language. In *CVPR*, 2024.

[64] Arpad E Elo. The proposed uscf rating system, its development, theory, and applications. *Chess life*, 1967.

[65] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996.

[66] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[67] Sabri Eyuboglu, Maya Varma, Khaled Kamal Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Re. Domino: Discovering systematic errors with cross-modal embeddings. In *ICLR*, 2022.

[68] Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (clip). In *ICML*, 2022.

[69] Amir Hossein Farzaneh and Xiaojun Qi. Facial expression recognition in the wild via deep attentive center loss. In *WACV*, 2021.

[70] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshop*, 2004.

[71] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshop*, 2004.

[72] Christoph Feichtenhofer, Axel Pinz, and Richard P Wildes. Spatiotemporal multiplier networks for video action recognition. In *CVPR*, 2017.

[73] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016.

[74] Enrico Fini, Pietro Astolfi, Adriana Romero-Soriano, Jakob Verbeek, and Michal Drozdzal. Improved baselines for vision-language pre-training, 2023.

[75] Riccardo Franceschini, Enrico Fini, Cigdem Beyan, Alessandro Conti, Federica Arrigoni, and Elisa Ricci. Multimodal emotion recognition with modality-pairwise unsupervised contrastive loss. In *ICPR*, 2022.

[76] FreeBSD. Web2 dictionary (revision 326913), 2023. Accessed May 17, 2023. `https://svnweb.freebsd.org/base/head/share/dict/web2?view=markup&pathrev=326913`.

[77] Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. In *ECCV*, 2024.

[78] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation, 2015.

[79] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv*, 2021.

[80] Paul Gavrikov, Jovita Lukasik, Steffen Jung, Robert Geirhos, Bianca Lamm, Muhammad Jehanzeb Mirza, Margret Keuper, and Janis Keuper. Are vision language models texture or shape biased and can we steer them? *arXiv*, 2024.

[81] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *TPAMI*, 2020.

[82] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *ECCV*, 2022.

[83] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016.

[84] Lluis Gomez, Yash Patel, Marçal Rusinol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *CVPR*, 2017.

[85] Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. In *NeurIPS*, 2013.

[86] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv*, 2024.

[87] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *CVPR*, 2023.

[88] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *ICML*, 2020.

[89] Kai Han, Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. Automatically discovering and learning new visual categories with ranking statistics. In *ICLR*, 2019.

[90] Behzad Hasani, Pooran Singh Negi, and Mohammad Mahoor. Breg-next: Facial affect computing using adaptive residual networks with bounded gradient. *Transactions on Affective Computing*, 2020.

[91] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[92] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

[93] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

[94] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv*, 2015.

[95] Yunzhong Hou and Liang Zheng. Visualizing adapted knowledge in domain transfer. In *CVPR*, 2021.

[96] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, 2019.

[97] Cheng-Yu Hsieh, Jieyu Zhang, Zixian Ma, Aniruddha Kembhavi, and Ranjay Krishna. Sugarcrepe: Fixing hackable benchmarks for vision-language compositionality. *NeurIPS*, 2023.

[98] Cheng-Yu Hsieh, Jieyu Zhang, Zixian Ma, Aniruddha Kembhavi, and Ranjay Krishna. Sugarcrepe: Fixing hackable benchmarks for vision-language compositionality. *NeurIPS*, 2024.

[99] Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling up vision-language pre-training for image captioning. In *CVPR*, 2022.

[100] Ziniu Hu, Ahmet Iscen, Chen Sun, Zirui Wang, Kai-Wei Chang, Yizhou Sun, Cordelia Schmid, David A Ross, and Alireza Fathi. Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In *CVPR*, 2023.

[101] Xinyu Huang, Yi-Jie Huang, Youcai Zhang, Weiwei Tian, Rui Feng, Yuejie Zhang, Yanchun Xie, Yaqian Li, and Lei Zhang. Open-set image tagging with multi-grained text supervision. *arXiv*, 2023.

[102] Badr Youbi Idrissi, Diane Bouchacourt, Randall Balestriero, Ivan Evtimov, Caner Hazirbas, Nicolas Ballas, Pascal Vincent, Michal Drozdzal, David Lopez-Paz, and Mark Ibrahim. Imagenet-x: Understanding model mistakes with factor of variation annotations. *arXiv*, 2022.

[103] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021.

[104] Xu Ji, Joao F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, 2019.

[105] Yanli Ji, Yuhan Hu, Yang Yang, Fumin Shen, and Heng Tao Shen. Cross-domain facial expression recognition via an intra-category common feature and inter-category distinction feature fusion network. *Neurocomputing*, 2019.

[106] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.

[107] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.

[108] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[109] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *Transactions on Big Data*, 2019.

[110] Armand Joulin, Laurens Van Der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *ECCV*, 2016.

[111] Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's" up" with vision-language models? investigating their struggle with spatial reasoning. In *EMNLP*, 2023.

[112] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[113] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017.

[114] Zaid Khan, Vijay Kumar BG, Samuel Schulter, Yun Fu, and Manmohan Chandraker. Self-training large language models for improved visual program synthesis with visual reinforcement. In *CVPR*, 2024.

[115] Youngeun Kim, Donghyeon Cho, Kyeongtak Han, Priyadarshini Panda, and Sungeun Hong. Domain adaptation without source data, 2020.

[116] Younghyun Kim, Sangwoo Mo, Minkyu Kim, Kyungmin Lee, Jaeho Lee, and Jinwoo Shin. Discovering and mitigating visual biases through keyword explanation. In *CVPR*, 2024.

[117] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *ICCV*, 2023.

[118] Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding language models to images for multimodal inputs and outputs. In *ICML*. PMLR, 2023.

[119] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *NeurIPS*, 2022.

[120] Yu Kong and Yun Fu. Human action recognition and prediction: A survey. *IJCV*, 2022.

[121] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshop*, 2013.

[122] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshop*, 2013.

[123] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.

[124] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *ICCV*, 2011.

[125] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *CVPR*, 2020.

[126] Jogendra Nath Kundu, Naveen Venkat, Ambareesh Revanur, R Venkatesh Babu, et al. Towards inheritable models for open-set domain adaptation. In *CVPR*, 2020.

[127] Weicheng Kuo, Yin Cui, Xiuye Gu, AJ Piergiovanni, and Anelia Angelova. F-vlm: Open-vocabulary object detection upon frozen vision and language models. *ICLR*, 2023.

[128] Vinod K Kurmi, Venkatesh K Subramanian, and Vinay P Namboodiri. Domain impression: A source data free domain adaptation method. In *WACV*, 2021.

[129] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020.

[130] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

[131] Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander Rush, Douwe Kiela, et al. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *NeurIPS*, 2023.

[132] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? *NeurIPS*, 2025.

[133] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *CVPR*, 2019.

[134] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *NeurIPS*, 2020.

[135] Bo Li, Kaichen Zhang, Hao Zhang, Dong Guo, Renrui Zhang, Feng Li, Yuanhan Zhang, Ziwei Liu, and Chunyuan Li. Llava-next: Stronger llms supercharge multimodal capabilities in the wild. *https://llava-vl.github.io/blog/2024-05-10-llava-next-stronger-llms*, 2024.

[136] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv*, 2024.

[137] Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. Seed-bench: Benchmarking multimodal large language models. In *CVPR*, 2024.

[138] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *CVPR*, 2023.

[139] Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. Llava-med: Training a large language-and-vision assistant for biomedicine in one day. *NeurIPS*, 2024.

[140] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv*, 2023.

[141] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[142] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[143] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *NeurIPS*, 2021.

[144] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, 2024.

[145] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *CVPR*, 2020.

[146] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *CVPR*, 2020.

[147] Shan Li and Weihong Deng. Deep emotion transfer network for cross-database facial expression recognition. In *ICPR*, 2018.

[148] Shan Li and Weihong Deng. A deeper look at facial expression dataset bias. *Transactions on Affective Computing*, 2020.

[149] Shan Li, Weihong Deng, and JunPing Du. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In *CVPR*, 2017.

[150] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *EMNLP*, 2023.

[151] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *CVPR*, 2023.

[152] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *CVPR*, 2023.

[153] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*, 2020.

[154] Jian Liang, Dapeng Hu, Jiashi Feng, and Ran He. Dine: Domain adaptation from single and multiple black-box predictors. In *CVPR*, 2022.

[155] Jian Liang, Dapeng Hu, Yunbo Wang, Ran He, and Jiashi Feng. Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer. *TPAMI*, 2021.

[156] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *NeurIPS*, 2022.

[157] Benedetta Liberatori, Alessandro Conti, Paolo Rota, Yiming Wang, and Elisa Ricci. Test-time zero-shot temporal action localization. In *CVPR*, 2024.

[158] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *ACL*, 2022.

[159] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[160] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv*, 2023.

[161] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023.

[162] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023.

[163] Huan Liu, Lingyu Xiao, Jiangjiang Liu, Xiaofan Li, Ze Feng, Sen Yang, and Jingdong Wang. Revisiting mllms: An in-depth analysis of image classification abilities. *arXiv*, 2024.

[164] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, 2024.

[165] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

[166] Alexander Long, Wei Yin, Thalaiyasingam Ajanthan, Vu Nguyen, Pulak Purkait, Ravi Garg, Alan Blair, Chunhua Shen, and Anton van den Hengel. Retrieval augmented classification for long-tail visual recognition. In *CVPR*, 2022.

[167] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[168] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks, 2015.

[169] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *NeurIPS*, 2018.

[170] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[171] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[172] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. *NeurIPS*, 2024.

[173] Chuofan Ma, Yi Jiang, Xin Wen, Zehuan Yuan, and Xiaojuan Qi. Codet: Co-occurrence guided region-word alignment for open-vocabulary object detection. *NeurIPS*, 2023.

[174] Mengmeng Ma, Jian Ren, Long Zhao, Davide Testuggine, and Xi Peng. Are multimodal transformers robust to missing modality? In *CVPR*, 2022.

[175] Zixian Ma, Jerry Hong, Mustafa Omer Gul, Mona Gandhi, Irena Gao, and Ranjay Krishna. Crepe: Can vision-language foundation models reason compositionally? In *CVPR*, 2023.

[176] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv*, 2013.

[177] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv*, 2013.

[178] Prasanna Mayilvahanan, Thaddäus Wiedemer, Evgenia Rusak, Matthias Bethge, and Wieland Brendel. Does clip's generalization performance mainly stem from high train-test similarity? In *NeurIPS Workshop*, 2023.

[179] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018.

[180] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 1995.

[181] Shervin Minaee, Mehdi Minaei, and Amirali Abdolrashidi. Deep-emotion: Facial expression recognition using attentional convolutional network. *Sensors*, 2021.

[182] Mazda Moayeri, Sahil Singla, and Soheil Feizi. Hard imagenet: Segmentations for objects with strong spurious cues. *NeurIPS*, 2022.

[183] Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *Transactions on Affective Computing*, 2017.

[184] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfreund, Carl Vondrick, and Aude Oliva. Moments in time dataset: one million videos for event understanding. *TPAMI*, 2019.

[185] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[186] Jonathan Munro and Dima Damen. Multi-modal domain adaptation for fine-grained action recognition. In *CVPR*, 2020.

[187] Roberto Navigli and Simone Paolo Ponzetto. Babelnet: Building a very large multilingual semantic network. In *ACL*, 2010.

[188] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

[189] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

[190] OpenAI. Introducing chatgpt. *https://openai.com/blog/chatgpt*, 2022.

[191] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. *NeurIPS*, 2011.

[192] Boxiao Pan, Zhangjie Cao, Ehsan Adeli, and Juan Carlos Niebles. Adversarial cross-domain action recognition with co-attention. In *AAAI CAI*, 2020.

[193] Preksha Pareek and Ankit Thakkar. A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artificial Intelligence Review*, 2021.

[194] Aaron Parisi, Yao Zhao, and Noah Fiedel. Talm: Tool augmented language models. *arXiv*, 2022.

[195] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, 2012.

[196] Roberto Pecoraro, Valerio Basile, Viviana Bono, and Sara Gallo. Local multi-head channel self-attention for facial expression recognition. *arXiv*, 2021.

[197] Momchil Peychev, Mark Müller, Marc Fischer, and Martin Vechev. Automated classification of model errors on imagenet. *NeurIPS*, 2023.

[198] Hieu Pham, Zihang Dai, Golnaz Ghiasi, Kenji Kawaguchi, Hanxiao Liu, Adams Wei Yu, Jiahui Yu, Yi-Ting Chen, Minh-Thang Luong, Yonghui Wu, et al. Combined scaling for zero-shot transfer learning. *arXiv*, 2021.

[199] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv*, 2023.

[200] Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *ECCV*, 2020.

[201] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *ICLR*, 2024.

[202] Jielin Qiu, Yi Zhu, Xingjian Shi, Florian Wenzel, Zhiqiang Tang, Ding Zhao, Bo Li, and Mu Li. Benchmarking robustness of multimodal image-text models under distribution shift. *Journal of Data-centric Machine Learning Research*, 2023.

[203] Zhen Qiu, Yifan Zhang, Hongbin Lin, Shuaicheng Niu, Yanxia Liu, Qing Du, and Mingkui Tan. Source-free domain adaptation via avatar prototype generation and adaptation, 2021.

[204] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[205] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[206] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[207] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *ICML*, 2023.

[208] Rita Ramos, Desmond Elliott, and Bruno Martins. Retrieval-augmented image captioning. In *EACL*, 2023.

[209] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP-IJCNLP*, 2019.

[210] Pitchaporn Rewatbowornwong, Nattanat Chatthee, Ekapol Chuangsuwanich, and Supasorn Suwajanakorn. Zero-guidance segmentation using zero segment labels. *ICCV*, 2023.

[211] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[212] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[213] Aadarsh Sahoo, Rutav Shah, Rameswar Panda, Kate Saenko, and Abir Das. Contrast and mix: Temporal contrastive video domain adaptation with background mixing. *NeurIPS*, 2021.

[214] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 2021.

[215] Andrey V Savchenko. Facial expression and attributes recognition based on multitask learning of lightweight neural networks. In *SISY*, 2021.

[216] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *NeurIPS*, 2024.

[217] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

[218] Christoph Schuhmann, Robert Kaczmarczyk, Aran Komatsuzaki, Aarush Katta, Richard Vencu, Romain Beaumont, Jenia Jitsev, Theo Coombes, and Clayton Mullis. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. In *NeurIPS Workshop*, 2021.

[219] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv*, 2021.

[220] Javier Selva, Anders S Johansen, Sergio Escalera, Kamal Nasrollahi, Thomas B Moeslund, and Albert Clapés. Video transformers: A survey. *arXiv*, 2022.

[221] Tamar Rott Shaham, Sarah Schwettmann, Franklin Wang, Achyuta Rajaram, Evan Hernandez, Jacob Andreas, and Antonio Torralba. A multimodal automated interpretability agent. *arXiv*, 2024.

[222] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018.

[223] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *NeurIPS*, 2024.

[224] Jiawei Shi, Songhao Zhu, and Zhiwei Liang. Learning to amend facial expression representation via de-albino and affinity. *arXiv*, 2021.

[225] Zhelun Shi, Zhipin Wang, Hongxing Fan, Zhenfei Yin, Lu Sheng, Yu Qiao, and Jing Shao. Chef: A comprehensive evaluation framework for standardized assessment of multimodal large language models. *arXiv*, 2023.

[226] Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. *ICCV*, 2023.

[227] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *arXiv*, 2022.

[228] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *NeurIPS*, 2022.

[229] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. In *CVPR*, 2022.

[230] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. In *CVPR*, 2022.

[231] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? In *ICLR*, 2022.

[232] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. Understanding failures of deep networks via robust feature extraction. In *CVPR*, 2021.

[233] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv*, 2012.

[234] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv*, 2012.

[235] Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning. In *SIGIR*, 2021.

[236] Sanjay Subramanian, Medhini Narasimhan, Kushal Khangaonkar, Kevin Yang, Arsha Nagrani, Cordelia Schmid, Andy Zeng, Trevor Darrell, and Dan Klein. Modular visual question answering via code generation. In *ACL*, 2023.

[237] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Artificial Intelligence*, 2016.

[238] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *ICCV*, 2023.

[239] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv*, 2018.

[240] Yingtian Tang, Yutaro Yamada, Yoyo Zhang, and Ilker Yildirim. When are lemons purple? the concept association bias of vision-language models. In *EMNLP*, 2023.

[241] Omkar Thawakar, Dinura Dissanayake, Ketan More, Ritesh Thawkar, Ahmed Heakl, Noor Ahsan, Yuhao Li, Mohammed Zumri, Jean Lahoud, Rao Muhammad Anwer, et al. Llamav-o1: Rethinking step-by-step visual reasoning in llms. *arXiv*, 2025.

[242] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 2016.

[243] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *CVPR*, 2022.

[244] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv*, 2023.

[245] Hugo Touvron, Alexandre Sablayrolles, Matthijs Douze, Matthieu Cord, and Hervé Jégou. Grafit: Learning fine-grained image representations with coarse labels. In *ICCV*, 2021.

[246] Vishaal Udandarao, Max F Burg, Samuel Albanie, and Matthias Bethge. Visual data-type understanding does not emerge from scaling vision-language models. In *ICLR*, 2024.

[247] Vishaal Udandarao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip HS Torr, Adel Bibi, Samuel Albanie, and Matthias Bethge. No" zero-shot" without exponential data: Pretraining concept frequency determines multimodal model performance. *arXiv*, 2024.

[248] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *ECCV*, 2020.

[249] Vijay Vasudevan, Benjamin Caine, Raphael Gontijo Lopes, Sara Fridovich-Keil, and Rebecca Roelofs. When does dough become a bagel? analyzing the remaining mistakes on imagenet. *NeurIPS*, 2022.

[250] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

[251] Kai Wang, Xiaojiang Peng, Jianfei Yang, Debin Meng, and Yu Qiao. Region attention networks for pose and occlusion robust facial expression recognition. *TIP*, 2020.

[252] Mengmeng Wang, Jiazheng Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *arXiv*, 2021.

[253] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv*, 2024.

[254] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *ICLR*, 2022.

[255] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 2022.

[256] Zhengyao Wen, Wenzhong Lin, Tao Wang, and Ge Xu. Distract your attention: multi-head cross attention network for facial expression recognition. *arXiv*, 2021.

[257] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, 2019.

[258] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[259] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[260] Guowei Xu, Peng Jin, Li Hao, Yibing Song, Lichao Sun, and Li Yuan. Llava-o1: Let vision language models reason step-by-step. *arXiv*, 2024.

[261] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. Videoclip: Contrastive pre-training for zero-shot video-text understanding. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

[262] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *CVPR*, 2023.

[263] Mengde Xu, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. Side adapter network for open-vocabulary semantic segmentation. In *CVPR*, 2023.

[264] Mengde Xu, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. Side adapter network for open-vocabulary semantic segmentation. In *CVPR*, 2023.

[265] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*, 2019.

[266] Yuecong Xu, Haozhi Cao, Kezhi Mao, Zhenghua Chen, Lihua Xie, and Jianfei Yang. Aligning correlation information for domain adaptation in action recognition. *Neural Networks and Learning Systems*, 2022.

[267] Yuecong Xu, Jianfei Yang, Haozhi Cao, Kezhi Mao, Jianxiong Yin, and Simon See. Arid: A new dataset for recognizing action in the dark. In *International Workshop on Deep Learning for Human Activity Recognition*, 2021.

[268] Yuecong Xu, Jianfei Yang, Haozhi Cao, Keyu Wu, Min Wu, and Zhenghua Chen. Source-free video domain adaptation by learning temporal consistency for action recognition. In *ECCV*, 2022.

[269] Yuecong Xu, Jianfei Yang, Haozhi Cao, Keyu Wu, Min Wu, Rui Zhao, and Zhenghua Chen. Multi-source video domain adaptation with temporal attentive moment alignment. *arXiv*, 2021.

[270] Yuecong Xu, Jianfei Yang, Haozhi Cao, Min Wu, Xiaoli Li, Lihua Xie, and Zhenghua Chen. Leveraging endo- and exo-temporal regularization for black-box video domain adaptation, 2022.

[271] Jinyu Yang, Weizhi An, Sheng Wang, Xinliang Zhu, Chaochao Yan, and Junzhou Huang. Label-driven reconstruction for domain adaptation in semantic segmentation. In *ECCV*, 2020.

[272] Shiqi Yang, Joost van de Weijer, Luis Herranz, Shangling Jui, et al. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. *NeurIPS*, 2021.

[273] Shiqi Yang, Yaxing Wang, Joost Weijer, and Luis Herranz. Unsupervised domain adaptation without source data by casting a bait, 2020.

[274] Shaokai Ye, Jessy Lauer, Mu Zhou, Alexander Mathis, and Mackenzie Mathis. Amadeusgpt: a natural language interface for interactive animal behavioral analysis. *NeurIPS*, 2024.

[275] Sriram Yenamandra, Pratik Ramesh, Viraj Prabhu, and Judy Hoffman. Facts: First amplify correlations and then slice to discover bias. In *ICCV*, 2023.

[276] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv*, 2022.

[277] Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Convolutions die hard: Open-vocabulary segmentation with single frozen convolutional clip. *NeurIPS*, 2023.

[278] Qihang Yu, Xiaohui Shen, and Liang-Chieh Chen. Towards open-ended visual recognition with large language models. In *ECCV*, 2024.

[279] Zhihao Yuan, Jinke Ren, Chun-Mei Feng, Hengshuang Zhao, Shuguang Cui, and Zhen Li. Visual programming for zero-shot open-vocabulary 3d visual grounding. In *CVPR*, 2024.

[280] Kaiyu Yue, Bor-Chun Chen, Jonas Geiping, Hengduo Li, Tom Goldstein, and Ser-Nam Lim. Object recognition as next token prediction. In *CVPR*, 2024.

[281] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *CVPR*, 2024.

[282] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *ICLR*, 2022.

[283] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *ICLR*, 2023.

[284] Stefanos Zafeiriou, Dimitrios Kollias, Mihalis A Nicolaou, Athanasios Papaioannou, Guoying Zhao, and Irene Kotsia. Aff-wild: valence and arousal'in-the-wild'challenge. In *CVPR*, 2017.

[285] Giacomo Zara, Alessandro Conti, Subhankar Roy, Stéphane Lathuilière, Paolo Rota, and Elisa Ricci. The unreasonable effectiveness of large language-vision models for source-free video domain adaptation. In *ICCV*, 2023.

[286] Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-vocabulary object detection using captions. In *CVPR*, 2021.

[287] Marcus Vinicius Zavarez, Rodrigo F Berriel, and Thiago Oliveira-Santos. Cross-database facial expression recognition based on fine-tuned deep convolutional network. In *SIBGRAPI*, 2017.

[288] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In *CVPR*, June 2019.

[289] Yan Zeng, Xinsong Zhang, and Hang Li. Multi-grained vision language pre-training: Aligning texts with visual concepts. In *ICML*, 2022.

[290] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

[291] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. *ICCV*, 2023.

[292] Gengyuan Zhang, Yurui Zhang, Kerui Zhang, and Volker Tresp. Can vision-language models be a good guesser? exploring vlms for times and location reasoning. In *WACV*, 2024.

[293] Haojian Zhang, Yabin Zhang, Kui Jia, and Lei Zhang. Unsupervised domain adaptation of black-box source models. *arXiv*, 2021.

[294] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *CVPR*, 2018.

[295] Yuhang Zhang, Chengrui Wang, and Weihong Deng. Relative uncertainty learning for facial expression recognition. *NeurIPS*, 2021.

[296] Yuhui Zhang, Alyssa Unell, Xiaohan Wang, Dhruba Ghosh, Yuchang Su, Ludwig Schmidt, and Serena Yeung-Levy. Why are visually-grounded language models bad at image classification? *NeurIPS*, 2024.

[297] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial expression recognition to interpersonal relation prediction. *IJCV*, 2018.

[298] Hang Zhao, Xavier Puig, Bolei Zhou, Sanja Fidler, and Antonio Torralba. Open vocabulary scene parsing. In *ICCV*, 2017.

[299] Xu Zhao, Wenchao Ding, Yongqi An, Yinglong Du, Tao Yu, Min Li, Ming Tang, and Jinqiao Wang. Fast segment anything, 2023.

[300] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Chongxuan Li, Ngai-Man Man Cheung, and Min Lin. On evaluating adversarial robustness of large vision-language models. *NeurIPS*, 2024.

[301] Kaizhi Zheng, Xuehai He, and Xin Eric Wang. Minigpt-5: Interleaved vision-and-language generation via generative vokens. *arXiv*, 2023.

[302] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *NeurIPS*, 2023.

[303] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos, 2018.

[304] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[305] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *IJCV*, 2019.

[306] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 2022.

[307] Yiyang Zhou, Chenhang Cui, Jaehong Yoon, Linjun Zhang, Zhun Deng, Chelsea Finn, Mohit Bansal, and Huaxiu Yao. Analyzing and mitigating object hallucination in large vision-language models. *ICLR*, 2024.

[308] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *ICLR*, 2024.

[309] Ronghang Zhu, Gaoli Sang, and Qijun Zhao. Discriminative feature adaptation for cross-domain facial expression recognition. In *ICB*, 2016.

[310] Yi Zhu, Xinyu Li, Chunhui Liu, Mohammadreza Zolfaghari, Yuanjun Xiong, Chongruo Wu, Zhi Zhang, Joseph Tighe, R Manmatha, and Mu Li. A comprehensive study of deep video action recognition. *arXiv*, 2020.

Appendix A

Extended results of classification with Large Multimodal Models

Below, we report the extended results of the analyses we conducted for open-world image classification. In table A.1 we show the variation in correct and wrong predictions for each model when using more generic/specific prompts and domain-specific information. We additionally report the variation in text inclusion, Llama inclusion, and concept similarity for each model and dataset in table A.2 and table A.3. For the chain-of-thought experiments, we provide the variations on the correct and wrong predictions in table A.4, and the per-dataset and model variations in table A.5. We also provide the variations for the list, caption, and describe experiments in table A.6 and in table A.7. Finally, we report the complete results table for the reasoning models tested on the ten classification datasets in table A.8.

Model	Correct		Wrong	
	Specific	Generic	Specific	Generic
Be generic				
Idefics2 [132] 8B	-5.9	+11.3	-1.4	-4.0
InstructBLIP [58] Vicuna 7B	-9.9	+2.2	+0.5	+7.3
InternVL2 [39, 38] 2B	-5.3	+4.8	-1.2	+1.7
InternVL2 [39, 38] 4B	-9.3	+14.5	-3.9	-1.4
InternVL2 [39, 38] 8B	-20.6	+21.9	-5.5	+4.2
LLaVA-1.5 [162] 7B	-20.0	+3.8	-3.9	+20.0
LLaVA-NeXT [135] (Mistral 7B)	-8.0	+11.8	-3.2	-0.5
LLaVA-NeXT [135] (Vicuna 7B)	-17.8	+22.2	-4.2	-0.2
LLaVA-OV [136] (Qwen2 0.5B)	-7.1	-2.8	-0.8	+10.7
LLaVA-OV [136] (Qwen2 7B)	-2.9	+2.9	+0.6	-0.5
Phi-3-Vision [2]	-11.9	+6.2	+0.1	+5.6
Qwen2VL [253] 2B	-10.5	+0.6	+0.3	+9.6
Qwen2VL [253] 7B	-36.4	+19.5	+1.4	+15.5
Be specific				
Idefics2 [132] 8B	-3.2	-3.3	+0.7	+5.8
InstructBLIP [58] Vicuna 7B	+2.1	-6.7	+0.6	+4.0
InternVL2 [39, 38] 2B	-2.4	+0.2	+0.7	+1.5
InternVL2 [39, 38] 4B	-1.1	-5.4	+1.8	+4.7
InternVL2 [39, 38] 8B	-2.3	-3.5	+3.0	+2.8
LLaVA-1.5 [162] 7B	-2.9	-7.0	+0.8	+9.1
LLaVA-NeXT [135] (Mistral 7B)	-2.3	-3.4	+2.6	+3.1
LLaVA-NeXT [135] (Vicuna 7B)	-4.5	-1.7	+2.0	+4.2
LLaVA-OV [136] (Qwen2 0.5B)	-4.5	-1.8	-1.4	+7.7
LLaVA-OV [136] (Qwen2 7B)	+2.1	-5.6	+1.9	+1.6
Phi-3-Vision [2]	+0.4	-3.6	+2.3	+0.9
Qwen2VL [253] 2B	+2.7	-3.8	+3.7	-2.7
Qwen2VL [253] 7B	-14.4	+0.8	+1.4	+12.1
Dataset-specific				
Idefics2 [132] 8B	+14.9	-32.5	+7.0	+10.6
InstructBLIP [58] Vicuna 7B	+9.1	-10.7	+5.6	-4.0
InternVL2 [39, 38] 2B	+2.6	-3.9	+1.7	-0.4
InternVL2 [39, 38] 4B	+2.2	-5.5	+2.6	+0.7
InternVL2 [39, 38] 8B	-0.1	-3.6	+2.5	+1.2
LLaVA-1.5 [162] 7B	+5.6	-7.0	+3.6	-2.2
LLaVA-NeXT [135] (Mistral 7B)	+0.3	-7.1	+4.6	+2.2
LLaVA-NeXT [135] (Vicuna 7B)	+3.4	-8.6	+4.3	+1.0
LLaVA-OV [136] (Qwen2 0.5B)	+29.2	-8.9	+1.6	-21.9
LLaVA-OV [136] (Qwen2 7B)	+13.6	-14.8	+8.7	-7.5
Phi-3-Vision [2]	+5.5	-7.6	+2.6	-0.5
Qwen2VL [253] 2B	+13.6	-5.8	+1.1	-8.8
Qwen2VL [253] 7B	+15.0	-10.1	+2.1	-7.0

Table A.1: Gains on the types of model prediction when instructing the models to be more generic/specific, and when using dataset-specific prompts techniques on six datasets, *i.e.*, DTD, FGVCAircraft, Flowers102, Food101, OxfordPets, StanfordCars.

Model	DTD			FGVCAircraft			Flowers102		
	TI	LI	CS	TI	LI	CS	TI	LI	CS
Be generic									
Idefics2 [132] 8B	-7.0	-9.7	-8.5	-2.4	+27.2	-2.4	-25.7	+8.9	-13.3
InstructBLIP [58] Vicuna 7B	-10.6	-20.7	-9.3	-0.7	-9.1	-4.4	-29.6	+2.5	-19.0
InternVL2 [39, 38] 2B	-3.0	+3.8	-3.9	-2.5	+43.5	-15.8	-6.1	+31.8	-11.4
InternVL2 [39, 38] 4B	-6.3	-4.7	-3.9	-7.2	+63.2	-20.1	-13.7	+35.8	-15.2
InternVL2 [39, 38] 8B	-12.4	-16.1	-10.8	-7.3	+44.3	-21.3	-20.1	+37.7	-17.9
LLaVA-1.5 [162] 7B	-4.4	-21.7	-7.0	-0.2	+23.1	-0.3	-10.5	+27.0	-2.2
LLaVA-NeXT [135] (Mistral 7B)	-4.5	-6.0	-3.5	-5.9	+27.6	-17.7	-13.4	+25.5	-13.1
LLaVA-NeXT [135] (Vicuna 7B)	-3.1	-9.7	-5.6	-5.3	+59.6	-17.3	-12.8	+37.2	-12.3
LLaVa-OV [136] (Qwen2 0.5B)	-10.1	-20.4	-6.9	-8.7	+5.3	-18.2	-15.9	+18.2	-16.3
LLaVa-OV [136] (Qwen2 7B)	-39.1	-32.8	-26.1	-2.4	+4.8	-7.7	-29.8	+4.9	-21.4
Phi-3-Vision [2]	-5.7	-0.1	+0.5	-1.8	+8.1	-5.0	-28.0	-2.1	-13.5
Qwen2VL [253] 2B	-8.0	-13.4	-8.9	-35.5	-1.4	-36.6	-16.3	+1.7	-17.8
Qwen2VL [253] 7B	-11.1	-17.6	-10.5	-39.6	-9.9	-43.6	-55.7	-20.8	-40.0
Be specific									
Idefics2 [132] 8B	-1.0	-4.2	-1.7	+0.0	-3.5	-0.2	+0.2	-2.3	+0.0
InstructBLIP [58] Vicuna 7B	+1.9	+13.9	+0.1	+0.0	-12.4	-1.8	+5.5	+5.6	+1.8
InternVL2 [39, 38] 2B	-0.5	-4.5	-0.7	-0.8	+5.9	-3.6	-2.2	+8.1	-1.8
InternVL2 [39, 38] 4B	-0.8	-0.1	-1.1	+1.2	-11.1	+2.7	+0.9	-3.5	-1.3
InternVL2 [39, 38] 8B	+1.5	+0.0	+0.2	+0.1	-6.2	+1.5	-0.5	-4.5	+0.2
LLaVA-1.5 [162] 7B	+0.5	+0.3	-0.7	+0.0	-4.1	-0.6	+0.6	-4.1	-0.6
LLaVA-NeXT [135] (Mistral 7B)	+0.7	-2.0	+0.2	+1.1	-12.6	+2.5	+0.2	-3.0	-0.2
LLaVA-NeXT [135] (Vicuna 7B)	-0.8	-3.1	-1.0	-0.6	-5.8	-3.5	-3.0	+16.9	-4.0
LLaVa-OV [136] (Qwen2 0.5B)	-1.3	-10.0	-3.6	-1.2	-6.3	-1.4	+4.2	+7.2	+3.0
LLaVa-OV [136] (Qwen2 7B)	+1.1	+5.5	+1.5	+0.0	-3.9	+0.0	+2.6	+0.7	+1.4
Phi-3-Vision [2]	+0.7	+3.6	+0.4	+0.3	+6.2	+0.3	+5.2	+0.3	+4.1
Qwen2VL [253] 2B	+4.0	+4.1	+3.1	+6.4	-5.7	+9.1	+4.1	-3.2	+4.6
Qwen2VL [253] 7B	-2.8	-8.1	-4.1	-1.0	+0.8	-1.0	-21.9	-15.0	-16.3

Model	Food101			OxfordPets			StanfordCars		
	TI	LI	CS	TI	LI	CS	TI	LI	CS
Be generic									
Idefics2 [132] 8B	-19.8	+30.5	-20.7	-0.5	-15.5	-4.2	+0.0	+8.2	-19.1
InstructBLIP [58] Vicuna 7B	-15.7	-13.9	-13.3	-13.6	-20.5	-8.4	+0.0	+10.2	-24.5
InternVL2 [39, 38] 2B	-4.5	-11.5	-6.2	-14.8	-8.2	-9.1	+0.0	+13.9	-8.2
InternVL2 [39, 38] 4B	-13.3	-11.6	-10.4	-10.0	-3.6	-7.1	-0.0	+51.4	-14.1
InternVL2 [39, 38] 8B	-21.3	-18.0	-17.9	-15.0	-3.0	-9.6	-0.0	+65.6	-18.3
LLaVA-1.5 [162] 7B	-23.0	-41.7	-27.2	-5.0	-20.7	-4.1	+0.0	+45.3	-19.9
LLaVA-NeXT [135] (Mistral 7B)	-8.5	-0.4	-5.2	-19.6	-12.3	-14.2	-0.0	+43.2	-21.7
LLaVA-NeXT [135] (Vicuna 7B)	-19.0	-10.2	-16.2	-23.9	-0.7	-19.2	+0.0	+50.1	-24.5
LLaVa-OV [136] (Qwen2 0.5B)	-30.2	-51.5	-26.5	-25.6	-29.4	-20.3	-0.1	+3.4	-27.6
LLaVa-OV [136] (Qwen2 7B)	-11.2	-2.3	-15.0	-0.0	-0.7	-0.3	+0.0	+4.8	-24.4
Phi-3-Vision [2]	-14.7	-17.8	-11.8	-16.9	-30.2	-9.9	+0.0	+39.4	-15.9
Qwen2VL [253] 2B	-18.4	-22.5	-13.5	-41.6	-22.8	-26.9	-0.0	-18.4	-6.3
Qwen2VL [253] 7B	-41.5	-31.4	-30.2	-15.4	-11.1	-10.0	+0.0	-4.3	-34.4
Be specific									
Idefics2 [132] 8B	-2.2	-5.6	-2.3	-0.1	-5.4	-1.5	+0.0	+5.4	-1.8
InstructBLIP [58] Vicuna 7B	-0.2	+2.9	-0.3	-0.1	-4.7	-0.9	+0.0	-22.3	+7.1
InternVL2 [39, 38] 2B	+0.1	-3.4	-0.5	-2.4	-1.8	-1.2	-0.0	+1.2	-2.6
InternVL2 [39, 38] 4B	+0.1	-3.3	-1.3	+7.6	-4.9	+3.6	-0.0	-15.1	+2.1
InternVL2 [39, 38] 8B	-0.2	-3.2	-0.5	+2.0	-7.1	+1.4	+0.0	-13.4	+1.6
LLaVA-1.5 [162] 7B	-0.3	-8.1	-2.2	+0.3	+0.9	-0.8	+0.0	-11.7	-0.4
LLaVA-NeXT [135] (Mistral 7B)	-0.5	-2.6	-0.4	-0.9	-1.8	-0.5	+0.0	-18.1	+0.3
LLaVA-NeXT [135] (Vicuna 7B)	-1.5	-5.8	-1.6	-3.4	-2.7	-3.0	-0.0	-14.5	-1.6
LLaVa-OV [136] (Qwen2 0.5B)	-3.6	-6.0	-4.5	+0.5	-1.2	+0.8	-0.1	-2.4	-7.2
LLaVa-OV [136] (Qwen2 7B)	+2.6	-0.8	+3.2	+0.3	-4.8	+0.3	+0.0	-5.5	+2.9
Phi-3-Vision [2]	+3.4	-0.9	+1.0	+2.4	-1.6	+1.4	-0.1	-2.5	+1.6
Qwen2VL [253] 2B	+2.9	+1.3	+1.9	+13.4	+5.5	+7.8	+0.1	-3.4	+4.5
Qwen2VL [253] 7B	-11.2	-17.8	-8.7	-9.1	-3.9	-4.7	+0.0	-0.3	-5.5

Table A.2: Relative performance variation with the generic/specific prompts on six datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity.

Model	DTD			FGVCAircraft			Flowers102		
	TI	LI	CS	TI	LI	CS	TI	LI	CS
Idefics2 [132] 8B	+6.0	+3.2	+6.8	+2.4	-37.3	+2.2	+24.9	-10.2	+12.9
InstructBLIP [58] Vicuna 7B	+9.0	+11.2	+6.1	+0.7	-0.3	+4.1	+26.6	-11.1	+17.9
InternVL2 [39, 38] 2B	+2.3	-5.9	+0.6	+1.5	-20.9	+7.6	+2.2	-24.6	+7.1
InternVL2 [39, 38] 4B	+3.8	+0.9	+1.3	+3.8	-25.3	+10.2	+5.8	-12.0	+5.6
InternVL2 [39, 38] 8B	+6.5	+1.5	+1.5	+2.6	-13.9	+8.4	+1.9	-9.4	+2.4
LLaVA-1.5 [162] 7B	+0.4	-0.6	+0.7	+0.1	-8.4	+0.7	+3.8	-18.7	+1.6
LLaVA-NeXT [135] (Mistral 7B)	+3.7	+1.1	+1.4	+3.4	-29.4	+10.0	+2.4	-5.5	+3.4
LLaVA-NeXT [135] (Vicuna 7B)	+0.4	-3.7	+0.8	+2.9	-27.0	+10.2	+1.0	-11.1	+2.0
LLaVA-OV [136] (Qwen2 0.5B)	+8.2	+8.7	+3.9	+7.5	-18.5	+16.9	+16.8	-17.1	+16.7
LLaVA-OV [136] (Qwen2 7B)	+36.4	+22.8	+23.9	+2.4	-5.4	+7.7	+16.0	-7.1	+10.7
Phi-3-Vision [2]	+6.0	-4.5	-1.8	+1.4	-8.4	+5.0	+19.7	-6.5	+12.2
Qwen2VL [253] 2B	+7.5	+7.2	+5.5	+21.8	+0.7	+19.4	+10.2	-1.3	+11.1
Qwen2VL [253] 7B	+7.9	+3.1	+4.4	+38.2	+16.1	+42.2	+13.6	-1.8	+11.8

Model	Food101			OxfordPets			StanfordCars		
	TI	LI	CS	TI	LI	CS	TI	LI	CS
Idefics2 [132] 8B	+15.3	-19.6	+15.0	+0.4	+9.9	+3.2	+0.0	-2.1	+16.8
InstructBLIP [58] Vicuna 7B	+6.4	+1.5	+5.7	+12.8	+16.6	+7.9	+0.0	-7.4	+19.9
InternVL2 [39, 38] 2B	+0.8	+5.4	+1.8	+9.8	+5.9	+6.5	-0.0	-8.4	+3.1
InternVL2 [39, 38] 4B	+3.0	+4.1	+2.2	+4.7	+3.6	+3.8	+0.0	-18.5	+4.9
InternVL2 [39, 38] 8B	+2.0	+0.5	+1.5	+2.0	+3.8	+2.0	-0.1	-14.9	+0.8
LLaVA-1.5 [162] 7B	+7.6	+3.6	+5.5	+3.9	+15.4	+3.7	+0.0	-3.7	+7.1
LLaVA-NeXT [135] (Mistral 7B)	+3.5	-2.2	+2.0	-0.3	+8.8	+0.5	+0.0	-23.9	+4.5
LLaVA-NeXT [135] (Vicuna 7B)	+5.0	-2.2	+3.1	+14.7	+3.9	+12.4	-0.0	-7.6	+5.4
LLaVA-OV [136] (Qwen2 0.5B)	+25.6	+38.7	+20.6	+25.4	+10.9	+19.4	+0.1	-4.0	+19.6
LLaVA-OV [136] (Qwen2 7B)	+10.1	+3.5	+14.8	-0.0	-4.3	+0.5	+0.0	-8.7	+22.3
Phi-3-Vision [2]	+5.4	+2.1	+3.8	+10.4	+5.0	+6.4	-0.1	-0.5	+9.0
Qwen2VL [253] 2B	+9.8	+6.9	+6.5	+38.7	+22.3	+24.5	+0.0	+17.8	+5.5
Qwen2VL [253] 7B	+7.7	+3.5	+4.9	+3.3	+9.5	+4.3	-0.1	+13.7	+17.2

Table A.3: Relative performance variation with dataset-specific prompts. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity.

	Correct		Wrong	
Model	**Specific**	**Generic**	**Specific**	**Generic**
Zero-shot chain-of-thought				
InternVL2 [39, 38] 2B	+3.5	-5.5	+1.4	+0.6
InternVL2 [39, 38] 4B	+4.8	+9.7	-2.0	-12.5
InternVL2 [39, 38] 8B	+3.9	+2.1	-1.1	-4.9
Qwen2VL [253] 2B	+8.0	+3.1	-0.4	-10.8
Qwen2VL [253] 7B	+6.9	+8.6	-2.5	-13.1
LlamaV-o1 prompt				
InternVL2 [39, 38] 2B	+6.7	-8.8	+0.4	+1.7
InternVL2 [39, 38] 4B	+0.5	-9.8	+0.4	+9.0
InternVL2 [39, 38] 8B	+3.7	-6.2	+0.6	+1.9
Qwen2VL [253] 2B	+12.7	-8.6	+1.0	-5.1
Qwen2VL [253] 7B	+4.4	-6.3	+0.8	+1.0
LLaVA-CoT prompt				
InternVL2 [39, 38] 2B	7.3	-1.2	-0.9	-5.2
InternVL2 [39, 38] 4B	5.2	10.0	-2.3	-12.8
InternVL2 [39, 38] 8B	1.5	22.5	-2.8	-21.1
Qwen2VL [253] 2B	6.3	-4.2	0.3	-2.4
Qwen2VL [253] 7B	6.3	7.0	-2.0	-11.3
Reasoning models				
InternVL2.5 [36] 2B	-2.5	-7.6	+4.2	+6.0
InternVL2.5 [36] 4B	+4.7	-2.4	+4.4	-6.6
InternVL2.5 [36] 8B	+0.7	-0.3	+4.1	-4.5
Qwen2.5VL [12] 3B	+10.8	-6.5	+2.6	-6.9
Qwen2.5VL [12] 7B	+19.1	-9.4	+3.5	-13.2

Table A.4: Gains on the types of model prediction when instructing the models to reason with chain-of-thought, and when using reasoning models on five datasets, *i.e.*, Caltech101, DTD, Flowers102, OxfordPets, UCF101.

	Caltech101			DTD			Flowers102			OxfordPets			UCF101		
Model	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**
Zero-shot chain-of-thought															
InternVL2 [39, 38] 2B	+0.3	-0.5	+4.0	+2.3	+18.5	+5.3	-3.5	-7.8	+1.3	+4.9	-14.9	+5.3	+6.3	+5.3	+6.3
InternVL2 [39, 38] 4B	-5.1	+2.2	-0.7	+1.4	+29.3	+3.1	+0.9	+16.0	+2.7	+3.0	+6.9	+4.9	+4.6	+23.9	+4.6
InternVL2 [39, 38] 8B	-2.3	+2.2	+0.5	+4.0	+20.7	+3.5	-1.7	+13.8	+0.1	+2.3	-9.7	+4.2	+4.3	+14.9	+3.2
Qwen2VL [253] 2B	+1.8	+3.6	+5.5	+6.5	+25.4	+8.2	-1.9	+5.4	+2.0	+6.8	+3.6	+6.7	+5.7	+23.2	+7.9
Qwen2VL [253] 7B	-2.9	+5.0	+1.5	+6.5	+21.5	+5.5	-4.9	+13.6	-1.5	+7.1	+17.0	+7.7	+2.3	+23.9	+5.4
LlamaV-o1 multi-round prompt															
InternVL2 [39, 38] 2B	+0.4	+2.4	+4.1	+3.0	+7.8	+4.1	-2.6	-7.0	+1.9	+9.9	-18.8	+8.3	+5.5	+11.4	+5.0
InternVL2 [39, 38] 4B	-2.7	-4.3	-3.3	-0.5	+4.1	-0.7	+0.2	-9.1	-0.5	+3.4	-31.1	-5.2	+4.1	+3.9	+2.6
InternVL2 [39, 38] 8B	-1.5	-2.0	+1.3	+3.2	+8.9	+3.6	-2.5	+3.0	-2.2	+8.2	-15.5	+7.6	+5.4	+3.1	+2.8
Qwen2VL [253] 2B	+0.8	+0.3	+4.8	+6.3	+8.1	+6.7	-5.8	-7.7	-4.1	+27.6	+0.1	+19.5	+7.9	+19.1	+9.7
Qwen2VL [253] 7B	-1.0	-3.0	-1.5	+3.3	+2.4	+0.2	-6.8	-17.3	-10.3	+22.9	-3.4	+16.4	+3.3	+10.9	+4.2
LLaVA-COT prompt															
InternVL2 [39, 38] 2B	-0.6	+6.4	+4.1	+1.6	+19.8	+3.9	-3.8	+11.2	+1.2	+7.9	-8.6	+6.7	+5.8	+11.9	+5.2
InternVL2 [39, 38] 4B	+0.1	+5.0	+2.0	+0.8	+23.7	+2.9	-4.9	+30.8	-3.0	+6.7	+7.7	+7.2	+3.6	+16.4	+3.7
InternVL2 [39, 38] 8B	-1.7	+5.8	+1.3	+0.4	+25.3	+2.8	-8.9	+44.7	-6.8	-2.9	+28.9	+1.3	+3.2	+18.8	+0.8
Qwen2VL [253] 2B	+1.6	+0.4	+5.2	+4.4	+10.3	+6.4	-8.9	-7.9	-5.4	+5.6	-8.0	+6.1	+9.1	+18.5	+10.6
Qwen2VL [253] 7B	+0.3	+3.0	+3.4	+0.3	+12.5	+4.5	-10.2	+8.8	-7.2	+8.5	+16.2	+9.6	+6.6	+22.6	+7.1

Table A.5: Relative performance variation with chain-of-thought prompts on five datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity.

	Correct		**Wrong**	
Model	**Specific**	**Generic**	**Specific**	**Generic**
List				
IDEFICS2 [132] 8B	-4.2	-9.5	+4.0	+9.7
INSTRUCTBLIP [58] Vicuna 7B	+9.6	-16.1	+1.5	+5.0
INTERNVL2 [39, 38] 2B	-3.1	-14.2	+3.5	+13.9
INTERNVL2 [39, 38] 4B	-1.4	-10.9	+0.9	+11.3
INTERNVL2 [39, 38] 8B	-2.6	-6.7	+0.8	+8.5
LLAVA-1.5 [162] 7B	-2.6	-13.7	+2.6	+13.6
LLAVA-NEXT [135] (Mistral 7B)	-4.4	+1.3	-0.4	+3.5
LLAVA-NEXT [135] (Vicuna 7B)	-1.6	-5.7	+1.3	+6.0
LLAVA-OV [136] (Qwen2 0.5B)	+3.6	-14.7	+2.0	+9.1
LLAVA-OV [136] (Qwen2 7B)	+9.3	-13.5	+2.2	+2.0
PHI-3-VISION [2]	-0.5	-16.8	+2.5	+14.9
QWEN2VL [253] 2B	+2.4	-5.0	-0.3	+3.0
QWEN2VL [253] 7B	-10.7	-9.1	+2.6	+17.2
Caption				
IDEFICS2 [132] 8B	+0.1	-12.0	+3.2	+8.6
INSTRUCTBLIP [58] Vicuna 7B	+2.3	-13.1	+2.8	+8.0
INTERNVL2 [39, 38] 2B	-4.0	-6.6	+1.0	+9.6
INTERNVL2 [39, 38] 4B	-2.1	-7.9	+1.1	+8.9
INTERNVL2 [39, 38] 8B	-7.4	-2.5	+0.9	+9.0
LLAVA-1.5 [162] 7B	+6.4	-15.9	+1.2	+8.4
LLAVA-NEXT [135] (Mistral 7B)	-20.2	+2.9	+2.4	+15.0
LLAVA-NEXT [135] (Vicuna 7B)	-14.2	-2.7	+2.2	+14.6
LLAVA-OV [136] (Qwen2 0.5B)	+4.7	-8.4	+0.0	+3.7
LLAVA-OV [136] (Qwen2 7B)	+0.7	-10.6	+2.7	+7.2
PHI-3-VISION [2]	+10.9	-13.2	+0.3	+2.0
QWEN2VL [253] 2B	+3.8	-0.9	-0.1	-2.8
QWEN2VL [253] 7B	+2.7	-4.7	+0.2	+1.8
Describe				
IDEFICS2 [132] 8B	-9.4	-13.9	4.2	19.1
INSTRUCTBLIP [58] Vicuna 7B	9.9	-11.2	1.6	-0.3
INTERNVL2 [39, 38] 2B	5.6	-11.6	0.6	5.4
INTERNVL2 [39, 38] 4B	4.6	-11.9	1.0	6.2
INTERNVL2 [39, 38] 8B	1.1	-6.7	0.3	5.3
LLAVA-1.5 [162] 7B	7.2	-17.4	1.5	8.7
LLAVA-NEXT [135] (Mistral 7B)	-2.1	1.6	-0.9	1.4
LLAVA-NEXT [135] (Vicuna 7B)	1.1	-5.0	0.0	3.9
LLAVA-OV [136] (Qwen2 0.5B)	15.0	-5.8	-2.1	-7.0
LLAVA-OV [136] (Qwen2 7B)	19.8	-10.5	-0.5	-8.8
PHI-3-VISION [2]	10.3	-11.8	-0.2	1.6
QWEN2VL [253] 2B	8.7	-7.9	0.1	-0.9
QWEN2VL [253] 7B	8.8	-8.9	0.3	-0.2

Table A.6: Gains on the types of model prediction when instructing the models with multi-label prompts on five datasets, *i.e.*, Caltech101, DTD, Flowers102, OxfordPets, UCF101.

	Caltech101			DTD			Flowers102		
Model	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**	**TI**	**LI**	**CS**
List									
Idefics2 [132] 8B	-1.5	-12.8	-2.8	+3.6	+3.4	+2.0	+3.6	-19.3	-0.3
InstructBLIP [58] Vicuna 7B	+12.5	-3.5	+4.5	+6.8	+15.8	+6.0	+13.7	-24.2	+5.9
InternVL2 [39, 38] 2B	-0.8	-6.1	+1.2	-1.5	-2.8	-1.7	-4.2	-23.5	-2.4
InternVL2 [39, 38] 4B	+1.8	-4.3	+0.7	-1.8	+3.6	-1.6	-3.7	-12.7	-5.7
InternVL2 [39, 38] 8B	-1.7	-2.2	-0.2	-0.8	+0.1	-1.2	-4.8	-2.3	-5.2
LLaVA-1.5 [162] 7B	-0.7	-6.3	-2.7	+0.7	+2.6	+1.4	+0.8	-21.5	-0.9
LLaVA-NeXT [135] (Mistral 7B)	-2.5	-2.1	-0.4	-1.1	-1.2	-0.8	-5.1	-2.1	-4.1
LLaVA-NeXT [135] (Vicuna 7B)	-1.5	-2.9	-0.6	-0.7	-1.4	-1.0	-2.7	-3.7	-2.8
LLaVA-OV [136] (Qwen2 0.5B)	+6.3	-8.1	+0.5	+1.8	+5.7	+4.4	+8.1	-23.8	+4.0
LLaVA-OV [136] (Qwen2 7B)	+6.5	+1.1	+3.6	+3.0	+14.2	+6.2	+1.6	-7.2	-0.6
Phi-3-Vision [2]	-1.1	-11.9	+1.0	-1.9	+1.6	-1.8	+3.3	-17.1	+1.4
Qwen2VL [253] 2B	+0.4	-0.1	+2.6	+5.9	+9.7	+6.3	-10.6	-12.2	-8.0
Qwen2VL [253] 7B	-1.8	-10.4	-1.0	+0.4	+2.7	+1.4	-24.2	-44.0	-21.4
Caption									
Idefics2 [132] 8B	-2.8	-5.4	-4.9	+5.3	+13.5	+3.2	+6.2	-33.5	-0.7
InstructBLIP [58] Vicuna 7B	+9.4	-8.6	+0.0	+4.7	+14.6	+3.4	+6.6	-34.5	-0.8
InternVL2 [39, 38] 2B	-6.3	-4.5	-9.9	-1.1	-5.1	-6.3	-2.2	-20.3	-7.3
InternVL2 [39, 38] 4B	-0.8	-6.5	-9.2	-2.2	-2.9	-5.3	-1.9	-13.6	-7.9
InternVL2 [39, 38] 8B	-4.2	-4.3	-9.5	+1.0	+0.6	-4.8	-5.4	-4.2	-13.5
LLaVA-1.5 [162] 7B	+0.9	+1.6	+5.4	+3.0	+4.8	+7.1	-1.0	-27.5	+3.7
LLaVA-NeXT [135] (Mistral 7B)	-9.4	-15.5	-17.7	-5.5	-20.9	-9.0	-8.1	-10.2	-15.2
LLaVA-NeXT [135] (Vicuna 7B)	-8.4	-13.5	-16.9	-4.9	-20.3	-9.7	-2.2	-5.9	-11.3
LLaVA-OV [136] (Qwen2 0.5B)	+2.5	-0.1	-8.9	+1.2	+2.7	+0.2	+12.6	-23.0	-0.3
LLaVA-OV [136] (Qwen2 7B)	+0.7	-6.8	-8.1	-1.7	+1.9	-3.1	-6.2	-28.4	-11.2
Phi-3-Vision [2]	+2.5	+0.4	+3.5	+2.4	+10.0	+5.8	+12.4	-4.6	+8.7
Qwen2VL [253] 2B	+0.3	+0.3	-5.2	+2.9	+7.1	+0.5	-6.6	-8.9	-14.8
Qwen2VL [253] 7B	-0.6	-1.1	+0.9	+1.7	+1.7	+1.2	-8.5	-15.8	-13.5
Describe									
Idefics2 [132] 8B	-10.0	-21.1	-5.1	-0.8	-4.9	-2.2	+3.3	-33.0	-0.6
InstructBLIP [58] Vicuna 7B	+11.5	-4.6	-1.9	+6.9	+18.7	+5.2	+15.2	-25.5	+3.5
InternVL2 [39, 38] 2B	-1.4	+1.4	+3.4	+0.9	-0.5	+1.0	-2.3	-18.0	+1.1
InternVL2 [39, 38] 4B	+2.0	-0.9	+2.2	+34.6	+5.4	-20.8	-0.4	-10.6	+0.1
InternVL2 [39, 38] 8B	-1.7	-0.2	+1.0	+0.3	+2.1	+0.4	-5.1	+0.0	-5.4
LLaVA-1.5 [162] 7B	+1.9	+0.6	+6.6	+3.8	+6.6	+7.5	+0.5	-28.0	+4.7
LLaVA-NeXT [135] (Mistral 7B)	-1.9	+0.8	+0.8	+0.3	-1.0	+0.8	-4.9	-3.8	-4.1
LLaVA-NeXT [135] (Vicuna 7B)	-1.7	+0.5	+0.4	-0.5	-0.7	+0.4	-2.7	-5.1	-3.7
LLaVA-OV [136] (Qwen2 0.5B)	+6.7	+7.6	+4.3	+6.1	+21.7	+8.0	+12.1	-6.9	+7.6
LLaVA-OV [136] (Qwen2 7B)	+8.1	+6.5	+5.8	+5.1	+22.4	+7.0	+7.4	-5.3	+3.0
Phi-3-Vision [2]	-0.1	-0.1	+2.0	+2.8	+11.5	+5.6	+11.7	-2.7	+8.5
Qwen2VL [253] 2B	+1.8	+1.8	+5.0	+5.4	+8.1	+6.7	-6.5	-13.3	-3.8
Qwen2VL [253] 7B	-1.0	-1.9	+2.9	+2.1	+2.4	+4.1	-2.6	-15.6	-4.1

Model	**OxfordPets** **TI**	 **LI**	 **CS**	**UCF101** **TI**	 **LI**	 **CS**
List						
Idefics2 [132] 8B	+1.8	-17.6	+3.5	+2.7	-6.6	-1.2
InstructBLIP [58] Vicuna 7B	+2.6	-29.1	+0.9	+6.7	+21.1	+9.3
InternVL2 [39, 38] 2B	+4.8	-32.8	+2.2	-3.1	-11.2	-1.6
InternVL2 [39, 38] 4B	+3.4	-35.5	+3.1	+2.2	-0.2	+0.6
InternVL2 [39, 38] 8B	+2.3	-26.5	+2.4	+1.0	-3.9	-3.2
LLaVA-1.5 [162] 7B	+0.9	-23.0	+1.4	-1.5	-10.2	+0.0
LLaVA-NeXT [135] (Mistral 7B)	-9.5	-7.6	-4.3	-0.6	-0.6	-1.8
LLaVA-NeXT [135] (Vicuna 7B)	+3.6	-19.0	+6.5	-1.3	-3.7	-1.5
LLaVA-OV [136] (Qwen2 0.5B)	+2.1	-39.0	+3.0	+5.5	+19.5	+4.0
LLaVA-OV [136] (Qwen2 7B)	+2.1	-46.9	+0.9	+13.6	+29.1	+19.9
Phi-3-Vision [2]	+1.5	-40.7	+1.9	+1.2	+3.0	+3.9
Qwen2VL [253] 2B	-3.4	-23.1	-5.0	+6.3	+18.6	+7.5
Qwen2VL [253] 7B	-5.6	-39.2	-1.0	+0.4	+0.8	-0.7
Caption						
Idefics2 [132] 8B	+8.3	-18.9	+6.4	+3.4	+1.3	+0.2
InstructBLIP [58] Vicuna 7B	+3.5	-24.6	+1.6	+5.2	+11.1	+4.5
InternVL2 [39, 38] 2B	+10.3	-13.3	+3.0	+0.2	-7.0	-2.4
InternVL2 [39, 38] 4B	+13.8	-18.3	+4.3	-0.1	-3.6	-19.7
InternVL2 [39, 38] 8B	+2.9	-20.3	-1.5	-1.4	-10.0	-4.6
LLaVA-1.5 [162] 7B	+5.6	-19.6	+7.9	+3.2	+9.2	+5.3
LLaVA-NeXT [135] (Mistral 7B)	-16.2	-10.6	-17.8	-4.8	-28.0	-9.7
LLaVA-NeXT [135] (Vicuna 7B)	+0.5	-17.0	-3.8	-4.8	-24.3	-9.5
LLaVA-OV [136] (Qwen2 0.5B)	+10.2	-18.2	+4.7	+7.1	+22.6	+5.2
LLaVA-OV [136] (Qwen2 7B)	+5.6	-35.4	+2.0	+9.5	+21.3	+16.9
Phi-3-Vision [2]	+17.1	-14.2	+12.8	+3.8	+13.5	+4.9
Qwen2VL [253] 2B	+20.0	+0.2	+8.5	+3.6	+15.4	+5.2
Qwen2VL [253] 7B	+13.2	-7.2	+10.0	+48.4	-32.7	+3.8
Describe						
Idefics2 [132] 8B	+2.1	-22.0	+4.0	-2.4	-19.9	-6.8
InstructBLIP [58] Vicuna 7B	+9.6	-12.3	+6.8	+5.0	+24.1	+8.3
InternVL2 [39, 38] 2B	+14.9	-18.3	+11.4	+9.7	-23.0	-6.1
InternVL2 [39, 38] 4B	+16.3	-20.4	+12.4	+1.7	+1.6	+0.8
InternVL2 [39, 38] 8B	+10.7	-18.7	+9.0	+0.9	-1.6	-2.1
LLaVA-1.5 [162] 7B	+5.6	-22.9	+7.9	+2.7	+10.0	+5.7
LLaVA-NeXT [135] (Mistral 7B)	-6.6	-3.1	-1.9	+1.1	+3.6	-1.1
LLaVA-NeXT [135] (Vicuna 7B)	+7.6	-13.1	+9.5	-1.1	+1.6	-1.8
LLaVA-OV [136] (Qwen2 0.5B)	+11.7	-7.6	+10.7	+5.4	+1.5	-7.7
LLaVA-OV [136] (Qwen2 7B)	+20.3	-18.0	+14.3	+16.6	+43.1	+21.6
Phi-3-Vision [2]	+14.6	-14.0	+11.2	+4.4	+15.0	+5.8
Qwen2VL [253] 2B	+12.8	-12.1	+10.0	+17.7	-8.0	+0.1
Qwen2VL [253] 7B	+24.6	-2.2	+18.6	+4.4	+14.2	+4.5

Table A.7: Relative performance variation with multi-label prompts on five datasets. TI stands for text inclusion, LI for Llama inclusion, and CS for concept similarity.

	Datasets										
Model	C101	DTD	ESAT	FGVC	FLWR	FOOD	PETS	CARS	S397	U101	Avg.
Text inclusion											
InternVL2.5 [37] 2B	55.8	12.6	**12.6**	1.5	10.9	17.0	8.7	0.0	16.3	13.7	14.9
InternVL2.5 [37] 4B	55.6	10.9	12.1	0.9	12.2	24.9	14.6	0.0	23.7	14.9	17.0
InternVL2.5 [37] 8B	56.4	12.1	8.4	3.0	16.8	29.7	7.2	**0.1**	24.7	13.8	17.2
Qwen2.5VL [12] 3B	62.1	13.9	1.6	18.8	49.7	44.2	38.9	0.0	30.7	18.0	27.8
Qwen2.5VL [12] 7B	**65.6**	**16.7**	4.4	**32.7**	**56.1**	**54.9**	**65.1**	0.0	**33.6**	**21.5**	**35.1**
Closed-world baselines											
CLIP [205]	87.1	52.6	42.7	27.2	76.9	89.9	88.1	76.2	65.6	72.7	67.9
SigLIP [290]	93.6	60.8	42.1	46.0	88.2	94.1	95.4	92.3	69.9	82.1	76.5
Llama inclusion											
InternVL2.5 [37] 2B	76.8	49.2	**47.2**	55.4	42.4	34.2	39.2	49.3	49.3	51.1	49.4
InternVL2.5 [37] 4B	77.1	48.7	42.6	61.4	43.3	52.0	49.4	49.8	63.1	53.6	54.1
InternVL2.5 [37] 8B	78.4	48.9	45.5	59.1	51.2	53.2	48.2	60.6	62.7	52.7	56.1
Qwen2.5VL [12] 3B	81.4	58.1	6.3	58.9	71.5	68.7	51.4	58.9	78.9	58.8	59.3
Qwen2.5VL [12] 7B	**84.5**	**59.8**	12.6	**69.6**	**75.2**	**76.4**	**71.0**	**71.2**	**81.1**	**67.0**	**66.8**
Closed-world baselines											
CLIP [205]	87.1	52.6	42.7	27.2	76.9	89.9	88.1	76.2	65.6	72.7	67.9
SigLIP [290]	93.6	60.8	42.1	46.0	88.2	94.1	95.4	92.3	69.9	82.1	76.5
Semantic similarity											
InternVL2.5 [37] 2B	49.5	25.2	31.4	21.4	26.7	33.5	22.4	41.8	39.8	41.5	33.3
InternVL2.5 [37] 4B	51.7	26.7	**31.7**	20.9	29.4	41.6	27.4	41.9	46.5	43.5	36.1
InternVL2.5 [37] 8B	**53.2**	27.1	29.5	21.4	32.1	42.2	24.2	42.9	**47.0**	43.2	36.3
Qwen2.5VL [12] 3B	51.8	27.4	12.3	28.9	45.4	48.0	31.4	50.9	**47.0**	43.2	38.6
Qwen2.5VL [12] 7B	48.8	**28.2**	18.9	**36.5**	**47.4**	**52.4**	**41.1**	**55.0**	**47.0**	**44.2**	**42.0**
Closed-world baselines											
CLIP [205]	90.8	69.9	67.7	66.7	83.4	93.7	91.8	80.5	92.2	83.3	82.0
SigLIP [290]	97.8	75.6	63.1	80.0	92.0	96.4	96.8	98.1	83.1	89.6	87.3
Concept similarity											
InternVL2.5 [37] 2B	78.0	46.2	**59.9**	33.1	47.8	53.5	39.7	50.0	59.5	61.4	52.9
InternVL2.5 [37] 4B	77.3	44.8	57.1	31.1	49.3	61.7	45.8	48.3	66.1	61.8	54.3
InternVL2.5 [37] 8B	77.7	45.4	52.7	31.5	54.2	64.5	41.9	49.2	66.3	62.2	54.6
Qwen2.5VL [12] 3B	81.8	51.3	23.8	52.4	72.6	73.2	62.1	64.7	70.8	62.9	61.6
Qwen2.5VL [12] 7B	**85.8**	**53.2**	41.3	**68.4**	**79.7**	**79.6**	**77.3**	**68.4**	**74.1**	**67.1**	**69.5**
Closed-world baselines											
CLIP [205]	90.8	69.9	67.7	66.7	83.4	93.7	91.8	80.5	92.2	83.3	82.0
SigLIP [290]	97.8	75.6	63.1	80.0	92.0	96.4	96.8	98.1	83.1	89.6	87.3

Table A.8: OW results of reasoning models on ten datasets. Higher is better, **bold** indicates best. Note that the Llama inclusion for CLIP closed-world equals the textual inclusion scores.

Appendix B

Extended results of benchmarking of Large Multimodal Models

We provide additional details about the results reported in section 4.1.3. In particular, results in table B.1 are associated with results already reported in c4.3. Similarly, table B.2 is associated with experiments on groups of data types, discussed in the same section. Finally, numbers reported in table B.3 corresponds to fig. 4.4.

By analyzing the results with APEx, we observe some interesting behaviors of LMMs. For example, the recognition performance of LMMs can be extremely sensitive to the questions/answers. Simply rephrasing the question-answers pairs, one can expect very different performances. Examples of this behavior can be observed by looking at Tab. B.1. While certain sensitivity to textual prompts is expected, it is often under-explored in current studies, for instance, in the study we compared with [246], only fixed prompts are used in their benchmark. Instead, the prompt question generation in APEx is handled by an LLM that produces different prompt questions at each round of experiment design, facilitating new findings along this aspect.

There are also interesting findings specific to different LLMs. For example (see table B.1), BLIP-2 is good at recognizing low-contrast images (up to 0.84% accuracy) while the performances of other LMMs are around random chance. IDEFICS has a strong recognition performance on high brightness (almost perfect), while the other models underperform. When recognizing high-contrast images, the synonym "well-lit" sometimes appears in the questions, but is only understood by LLaVA (with even improved performance), while other models fail to interpret the synonym properly. Interestingly, the performance of recognizing pixel transformation does not positively correlate with their semantic understanding. While models cannot recognize the presence of high contrast, that does not undermine their understanding of the semantic class depicted in the image.

Q&A	**Tools**		**Models**	
		BLIP-2	IDEFICS	LLaVA
"Can models identify left rotation in images?"				
Is the image rotated to the left?		0.49	0.47	0.50
- Yes	Retrieval("random"), Rotate(-90)			
- No	Retrieval("random")			
Is the wheeled vehicle in the image a car or a truck?		1.00	0.50	0.77
- A car	Generation("car"), Rotate(-90)			
- A truck	Generation("truck"), Rotate(-90)			
Is the image rotated to the left?		0.43	0.50	0.50
- Yes	Retrieval("random"), Rotate(-90)			
- No	Retrieval("random")			
Is the object in the image rotated to the left?		0.50	0.66	0.50
- Yes	Generation("dog"), Rotate(-90)			
- No	Generation("dog")			
Does the image contain a rotated object?		0.43	0.50	0.53
- Yes	Retrieval("random"), Rotate(-90)			
- No	Retrieval("random")			
"Can models identify low contrast in images?"				
Is the image low-contrast?		0.19	0.48	0.50
- Yes	Retrieval("dog"), ChangeContrast(0.5)			
- No	Retrieval("vehicle")			
Is the contrast level of the image very low?		0.84	0.50	0.50
- Yes, very low contrast	Generation("dog"), ChangeContrast(0.3)			
- No, contrast is high	Generation("vehicle"), ChangeContrast(1.0)			
"Can models identify zoom in images?"				
Has the image been zoomed in?		0.50	0.46	0.50
- Yes	Retrieval("wheeled vehicle"), Zoom(1.5)			
- No	Retrieval("wheeled vehicle")			
Is the object in the image a bird?		0.48	0.53	0.90
- Yes	Retrieval("bird"), Zoom(1.5)			
- No	Retrieval("random")			
Is the object in the image a dog?		0.50	0.50	1.00
- Yes	Retrieval("dog"), Zoom(1.5)			
- No	Retrieval("bird"), Zoom(1.5)			
"Can models identify right rotation in images?"				
Is the image rotated to the right?		0.47	0.55	0.50
- Yes	Retrieval("random"), Rotate(90)			
- No	Retrieval("random")			
Is the image rotated to the right?		0.50	0.85	0.50
- Yes	Retrieval("wheeled vehicle"), Rotate(90)			
- No	Retrieval("device")			
Is the rotated image showing a wheeled vehicle?		0.61	0.50	1.00
- Yes	Retrieval("wheeled vehicle"), Rotate(90)			
- No	Retrieval("device")			
Is the image rotated to the right?		0.41	0.67	0.50
- Yes	Retrieval("device"), Rotate(180)			
- No	Retrieval("device")			
Is the image rotated to the right by 45 degrees?		0.60	0.50	0.50
- Yes	Retrieval("device"), Rotate(45)			
- No	Retrieval("device")			
"Can models identify Gaussian noise in images?"				
Is there gaussian noise in the image?		0.65	0.50	1.00
- Yes	Generation("random"), AddGaussianNoise(1.4)			
- No	Generation("random")			
Does the image contain significant gaussian noise?		0.51	0.50	0.97
- Yes	Generation("random"), AddGaussianNoise(2.0)			

- No	Generation("random")			
Does the image contain mild gaussian noise?		0.70	0.50	0.79
- Yes	Generation("photography"), AddGaussianNoise(1.7)			
- No	Generation("photography")			
Is the image corrupted with mild gaussian noise?		0.57	0.56	0.52
- Yes	Generation("random"), AddGaussianNoise(1.7)			
- No	Generation("random")			
Is the image corrupted with mild gaussian noise?		0.51	0.53	0.53
- Yes	Generation("random"), AddGaussianNoise(1.7)			
- No	Generation("random")			
"Can models identify high contrast in images?"				
Is the image high contrast?		0.49	0.48	0.50
- Yes	Generation("random"), ChangeContrast(2)			
- No	Generation("random"), ChangeContrast(2)			
Is the image a high contrast scene?		0.61	0.50	0.76
- Yes	Generation("high contrast scene"), ChangeContrast(2)			
- No	Generation("low contrast scene"), ChangeContrast(0.5)			
Is the image well-lit?		0.75	0.50	0.50
- Yes	Retrieval("structure"), ChangeContrast(2)			
- No	Retrieval("natural object"), ChangeContrast(0.5)			
Is the object in the image clearly visible in a high contrast scene?		0.51	0.50	0.50
- Yes	Generation("structure"), ChangeContrast(2)			
- No	Generation("natural object"), ChangeContrast(0.5)			
Is the vehicle in the image clearly visible in a high contrast scene?		0.53	0.50	0.50
- Yes	Generation("wheeled vehicle"), ChangeContrast(2)			
- No	Generation("wheeled vehicle"), ChangeContrast(0.5)			
"Can models identify low brightness in images?"				
Is the image in low brightness?		0.39	0.51	0.50
- Yes	Generation("random"), ChangeBrightness(0.5)			
- No	Generation("random")			
Is the image in very low brightness?		0.33	0.50	0.50
- Yes	Generation("random"), ChangeBrightness(0.2)			
- No	Generation("random")			
Is the image in extremely low brightness?		0.52	0.50	0.50
- Yes	Generation("random"), ChangeBrightness(0.1)			
- No	Generation("random")			
Is the image in extremely low brightness?		0.46	0.50	0.50
- Yes	Generation("random"), ChangeBrightness(0.1)			
- No	Generation("random")			
What object is present in the image in extremely low brightness?		1.00	1.00	1.00
- A cat	Generation("cat"), ChangeBrightness(0.1)			
- A car	Generation("car"), ChangeBrightness(0.1)			
"Can models identify the presence of patching and reshuffling in images?"				
Is the image randomly reshuffled?		0.42	0.50	0.83
- No	Retrieval("random")			
- Yes	Retrieval("random"), CropAndShuffle(2)			
Is the image randomly patched?		0.53	0.42	0.79
- No	Retrieval("random")			
- Yes	Retrieval("random"), PasteGeometricShape("square", 48, [255, 0, 0], False, 1)			
Does the image contain a randomly patched object or reshuffled patches?		0.54	0.52	0.50
- No	Retrieval("random")			
- Yes	Retrieval("random"), CropAndShuffle(2)			
Does the image contain a randomly patched object and randomly reshuffled patches?		0.60	0.50	0.82
- No	Retrieval("random")			
- Yes	Retrieval("random"), CropAndShuffle(3)			
Is the image non-randomly reshuffled and patched?		0.67	0.32	0.50
- No	Retrieval("random")			
- Yes	Retrieval("random"), Mixup(1.0)			
"Can models identify the presence of cutmix in images?"				
Is cutmix present in the image?		0.61	0.51	0.62

- Yes	Retrieval("random"), CutMix(alpha=1.0)			
- No	Retrieval("random")			
What is the size of the cutmix present in the image?		0.35	0.30	0.33
- Small	Retrieval("cutmix"), CutMix(0.5)			
- Large	Retrieval("cutmix"), CutMix(1.5)			
- None	Retrieval("random")			
Is the cutmix covering a large portion of the image?		0.42	0.50	0.92
- Yes	Generation("cutmix")			
- No	Retrieval("random")			
"Can models identify jpeg compression in images?"				
Is the image compressed with jpeg compression?		0.50	0.31	0.50
- Yes	Retrieval("wheeled vehicle"), AddJPEGCompression(18.0)			
- No	Retrieval("dog")			
Is the image heavily compressed with jpeg compression?		0.50	0.50	0.50
- Yes	Generation("wheeled vehicle"), AddJPEGCompression(15.0)			
- No	Retrieval("dog")			
Is the image moderately compressed with jpeg compression?		0.50	0.50	0.50
- Yes	Generation("dog"), AddJPEGCompression(20.0)			
- No	Retrieval("wheeled vehicle")			
Is the image lightly compressed with jpeg compression?		0.92	0.50	0.50
- Yes	Retrieval("wheeled vehicle"), AddJPEGCompression(30.0)			
- No	Retrieval("dog")			
"Can models identify the presence of mixup in images?"				
Does the image contain mixup?		0.63	0.55	0.50
- Yes	Retrieval("wheeled vehicle"), Mixup(0.5)			
- No	Retrieval("wheeled vehicle")			
Does the image contain mixup?		0.69	0.55	0.68
- Yes	Retrieval("random"), Mixup(0.7)			
- No	Retrieval("random")			
"Can models identify high brightness in images?"				
Is the brightness level in the image high?		0.53	1.00	0.50
- Yes	Retrieval("sunny outdoor scenes"), ChangeBrightness(2)			
- No	Retrieval("low-light indoor scene"), ChangeBrightness(0.5)			
Is the brightness level in the image extremely high?		0.62	1.00	0.50
- Yes	Retrieval("sunny outdoor scenes"), ChangeBrightness(4)			
- No	Retrieval("low-light indoor scene"), ChangeBrightness(0.5)			
Is the image extremely bright?		0.50	0.97	0.50
- Yes	Generation("sunny outdoor scenes"), ChangeBrightness(4)			
- No	Generation("low-light indoor scene"), ChangeBrightness(0.5)			
"Can models identify if text is present in images?"				
Is there visible text in the image?		0.50	0.77	0.69
- Yes	Retrieval("random")			
- No	Retrieval("random"), OverlayColor([0, 0, 0], 1.0)			
Is the word 'stop' present in the image?		0.81	0.78	0.73
- Yes	Generation("Stop")			
- No	Generation("random")			
Is the word 'danger' present in the image?		0.66	0.96	1.00
- Yes	Generation("danger")			
- No	Generation("random")			
Is the word 'caution' present in the image?		0.48	0.78	0.91
- Yes	Generation("caution")			
- No	Generation("random")			
Is there any visible text present in the image?		0.50	0.50	0.54
- Yes	Retrieval("random")			
- No	Retrieval("random"), OverlayColor([0, 0, 0], 1.0)			
"Can models identify if objects are styled like plushies?"				
Is the image styled like a plushie?		1.00	0.50	0.50
- Yes	Generation("plushie")			
- No	Generation("dog")			
Is the object in the image styled like a plushie dog or a plushie cat?		1.00	1.00	1.00
- A plushie dog	Generation("plushie dog")			

- A plushie cat	Generation("plushie cat")			
Is the image styled like a plushie dog or a plushie cat?		1.00	1.00	1.00
- A plushie dog	Generation("plushie dog")			
- A plushie cat	Generation("plushie cat")			
Is the object in the image styled like a plushie dog, a plushie cat, or a plushie bear?		1.00	0.93	1.00
- A plushie dog	Generation("plushie dog")			
- A plushie cat	Generation("plushie cat")			
- A plushie bear	Generation("plushie bear")			
Is the image styled like a floral plushie or a plushie dog?		0.50	0.50	0.50
- A floral plushie	Generation("floral plushie")			
- A plushie dog	Generation("plushie dog")			
“Can models identify tiger stripes in images?”				
Are the stripes on the animal in the image tiger stripes?		0.23	0.50	0.50
- Yes	Retrieval("tiger")			
- No	Retrieval("zebra")			
Do the stripes on the animal in the image resemble tiger stripes?		1.00	0.75	0.52
- Yes	Retrieval("tiger")			
- No	Retrieval("zebra")			
Is the image displaying tiger stripes?		0.50	0.73	0.54
- Yes	Retrieval("tiger")			
- No	Retrieval("zebra")			
“Can models identify snow in images?”				
Is there snow in the image?		0.50	0.50	1.00
- Yes	Retrieval("snowy landscape")			
- No	Retrieval("landscape")			
What is the dominant color of the snowy region in the image?		1.00	1.00	0.96
- White	Retrieval("snowy landscape")			
- Blue	Retrieval("snowy landscape"), OverlayColor([0, 0, 255], 1.0)			
“Can models identify graffiti-styled images?”				
Is the image graffiti-styled?		0.54	0.50	1.00
- Yes	Generation("graffiti")			
- No	Retrieval("random")			
What type of graffiti style is depicted in the image?		0.96	0.50	0.50
- Calligraffiti	Retrieval("calligraffiti")			
- Stencil graffiti	Retrieval("stencil graffiti")			
“Can models identify vertical flip in images?”				
Is the image flipped vertically?		0.50	0.50	0.50
- Yes	Generation("wheeled vehicle"), Flip("vertical")			
- No	Generation("wheeled vehicle")			
Is there a vertical flip in the image of the wheeled vehicle?		0.50	0.50	0.50
- Yes	Generation("wheeled vehicle"), Flip("vertical")			
- No	Generation("wheeled vehicle")			
Is the vehicle in the image flipped vertically?		0.50	0.50	0.50
- Yes	Generation("wheeled vehicle"), Flip("vertical")			
- No	Generation("wheeled vehicle")			
Is there a vertical flip in the image of the selected class?		0.50	0.59	0.50
- Yes	Generation("wheeled vehicle"), Flip("vertical")			
- No	Generation("wheeled vehicle")			
Is the vehicle in the image flipped vertically?		0.49	0.50	0.50
- Yes	Generation("car"), Flip("vertical")			
- No	Generation("car")			
“Can models identify defocus blur in images?”				
Is the image defocused?		0.46	0.50	0.50
- Yes	Generation("car"), DefocusBlur(5.0)			
- No	Generation("car")			
Does the image contain strong defocus blur?		0.37	0.50	0.50
- Yes	Generation("wheeled vehicle"), DefocusBlur(8.0)			
- No	Generation("device")			
'Is the image heavily defocused?		0.38	0.50	0.50
- Yes	Generation("wheeled vehicle"), DefocusBlur(8.0)			
- No	Generation("device")			

“Can models identify cartoon-styled images?”				
Is the image a cartoon or a real-life photo?		0.50	0.50	0.50
- A cartoon	Generation("bird", "cartoon")			
- A real-life photo	Retrieval("random")			
What type of cartoon is the image depicting?		0.50	0.50	0.50
- A cartoon of a bird	Generation("bird", "cartoon")			
- A real-life photo of a bird	Retrieval("bird")			
Is the image depicting a comic-style illustration of a bird or a dog?		1.00	1.00	1.00
- A bird	Generation("bird", "comic-style")			
- A dog	Generation("dog", "comic-style")			
“Can models identify if several identical objects are present in images?”				
Is the image showing a single identical object?		0.50	0.71	0.87
- Yes	Generation("car")			
- No	Retrieval("random")			
Are there multiple identical objects in the image?		0.50	0.50	0.50
- Yes	Generation("car")			
- No	Retrieval("random")			
Are there multiple identical objects of different sizes and orientations in the image?		0.50	0.50	0.50
- Yes	Generation("car"), PasteObject("car", 64, 3)			
- No	Retrieval("random")			
How many identical objects are present in the image?		0.50	0.50	0.72
- One	Retrieval("random")			
- Two	Generation("car"), PasteObject("car", 64, 2)			
How many identical objects are present in the image?		0.50	0.50	0.91
- One	Retrieval("car")			
- Two	Retrieval("random"), PasteObject("car", 64, 2)			
“Can models identify embroidery-styled images?”				
Is the image in the embroidery style?		0.56	0.50	0.91
- Yes	Generation("embroidery")			
- No	Retrieval("random")			
What object is depicted in the embroidery-styled image?		0.50	0.50	0.50
- A bonsai tree	Generation("bonsai tree", "embroidery")			
- Another object	Retrieval("random", "embroidery")			
Is the object in the embroidery-styled image a bonsai tree or a flower?		0.50	0.50	0.50
- A bonsai tree	Generation("bonsai tree", "embroidery")			
- A flower	Generation("flower", "embroidery")			
Is the object in the embroidery-styled image a flower or a butterfly?		1.00	0.50	0.50
- A flower	Generation("flower", "embroidery")			
- A butterfly	Generation("butterfly", "embroidery")			
What object is depicted in the embroidery-styled image?		0.50	0.50	0.50
- A bonsai tree	Generation("bonsai tree", "embroidery")			
- Another object	Retrieval("random", "embroidery")			
“Can models identify if multiple objects of different classes are in the image?”				
Is there more than one dog in the image?		0.50	0.42	0.73
- Yes	Generation("dog"), PasteObject("dog", 256, 2)			
- No	Generation("dog")			
Is there more than one dog in the image?		0.50	0.38	0.67
- Yes	Generation("dog"), PasteObject("dog", 256, 2)			
- No	Generation("dog")			
Are there vehicles and animals in the image?		0.50	0.50	0.50
- Yes	Generation("wheeled vehicle"), PasteObject("dog", 256, 1)			
- No	Generation("device"), PasteObject("dog", 256, 1)			
Is there a car and a truck in the image? if so, are they both present?		0.50	0.50	0.50
- Yes	Generation("car")			
- No	Generation("truck")			
“Can models identify origami-styled in images?”				
Is the image origami-styled?		0.49	0.56	0.97
- Yes	Generation("origami")			
- No	Retrieval("random")			
What type of origami object is in the image?		0.81	0.52	0.50
- Paper crane	Generation("paper crane")			

- Random origami	Retrieval("origami")			
What type of origami object is in the image?		0.82	0.50	0.50
- Paper crane	Generation("paper crane")			
- Random origami	Retrieval("origami")			
Is the image a paper crane?		0.45	0.50	0.50
- Yes	Generation("paper crane")			
- No	Retrieval("origami")			
Is the image showing a folded paper boat?		0.61	0.50	0.50
- Yes	Generation("paper boat")			
- No	Generation("random")			
"Can models identify sketch-styled in images?"				
Is the image a pencil sketch of a vehicle or a natural object?		0.52	1.00	1.00
- Yes	Retrieval("wheeled vehicle"), EditStyle("pencil sketch")			
- No	Retrieval("snake"), EditStyle("pencil sketch")			
Is the image a pencil sketch of a bird or a primate?		0.94	0.91	0.97
- A bird	Retrieval("bird"), EditStyle("pencil sketch")			
- A primate	Retrieval("primate"), EditStyle("pencil sketch")			
"Can models identify sculpture-styled in images?"				
Is the image sculpture-styled?		0.41	0.50	1.00
- Yes	Retrieval("sculpture"), EditStyle("sculpture")			
- No	Retrieval("random")			
Is the sculpture in the image a stone sculpture?		0.76	0.57	0.98
- Yes	Generation("sculpture"), EditStyle("stone")			
- No	Retrieval("random")			
Is the sculpture in the image a stone sculpture?		0.68	0.54	0.97
- Yes	Generation("sculpture"), EditStyle("stone")			
- No	Retrieval("random")			
Is the sculpture in the image a marble sculpture?		0.72	0.56	0.97
- Yes	Generation("marble sculpture")			
- No	Retrieval("random")			
Is the sculpture in the image a bronze sculpture?		0.48	0.15	0.50
- Yes	Generation("bronze")			
- No	Retrieval("sculpture"), EditStyle("bronze")			
"Can models identify tattoo-styled in images?"				
What kind of tattoo is depicted in the image?		1.00	1.00	1.00
- A tribal tattoo	Retrieval("tribal tattoo"), EditStyle("tattoos")			
- A watercolor tattoo	Retrieval("watercolor tattoo"), EditStyle("tattoos")			
What kind of tattoo style is depicted in the image?		1.00	1.00	1.00
- A watercolor tattoo	Retrieval("watercolor tattoo"), EditStyle("tattoos")			
- A tribal tattoo	Retrieval("tribal tattoo"), EditStyle("tattoos")			

Table B.1: We report the complete outputs for the data types experiments associated with APEx when queries about data types are provided as input. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification.

Q&A	**Tools**		**Models**	
		BLIP-2	IDEFICS	LLaVA
"Can models identify geometric-type transformations?"				
Is the image rotated by 90 degrees clockwise?		0.47	0.45	0.50
- Yes	Retrieval("random"), Rotate(90)			
- No	Retrieval("random")			
Is the image reflected across the vertical axis?		0.52	0.52	0.50
- Yes	Retrieval("random"), Flip("vertical")			
- No	Retrieval("random")			
Is the image containing visible text?		0.57	0.12	0.85
- Yes	Retrieval("text")			
- No	Retrieval("random")			
Is the image noise-corrupted?		0.51	0.54	0.52

- Yes	Retrieval("random"), AddGaussianNoise(1.4)			
- No	Retrieval("random")			
Is there a zoomed-in region in the image?		0.42	0.50	0.50
- Yes	Retrieval("random"), Zoom(2.0)			
- No	Retrieval("random")			
"Can models identify pixel-type transformations?"				
Is the image grayscale?		0.93	0.50	0.50
- No	Generation("dog")			
- Yes	Generation("dog"), EditStyle("grayscale")			
Is the image rotated clockwise?		0.49	0.48	0.50
- No	Retrieval("random")			
- Yes	Retrieval("random"), Rotate(90)			
Is the image affected by gaussian noise?		0.56	0.50	0.66
- No	Retrieval("random")			
- Yes	Retrieval("random"), AddGaussianNoise(2)			
Is the image affected by jpeg compression?		0.54	0.50	0.50
- No	Retrieval("random")			
- Yes	Retrieval("random"), AddJPEGCompression(20.0)			
Is the image affected by defocus blur?		0.48	0.50	0.50
- No	Retrieval("random")			
- Yes	DefocusBlurImage("random"), AddJPEGCompression(10.0)			
"Can models identify semantic-type transformations?"				
Is the object in the image a bird or a vehicle?		0.75	0.50	0.50
- A bird	Generation("bird")			
- A vehicle	Generation("wheeled vehicle")			
Is the background of the image a natural or man-made environment?		0.50	0.50	0.50
- Natural	Generation("forest")			
- Man-made	Generation("cityscape")			
Is the style of the object in the image a photo or a pencil sketch?		0.50	0.50	0.50
- Photo	Generation("random", "photo")			
- Pencil sketch	Generation("random", "pencil sketch")			
Is the weather in the image sunny or cloudy?		0.74	0.77	0.97
- Sunny	Retrieval("random"), EditWeather("sunny")			
- Cloudy	Retrieval("random"), EditWeather("cloudy")			
Is the image a photograph of a vehicle or an illustration of an animal?		0.80	0.50	0.50
- Vehicle	Generation("wheeled vehicle")			
- Animal illustration	Generation("random", "pencil sketch")			
"Can models identify style-type transformations?"				
Is the image in a specific artistic style?		0.50	0.56	0.50
- Painting	Generation("dog", "oil painting")			
- Pastel	Retrieval("bird", "pastel")			
Is the image in a specific artistic style?		1.00	1.00	0.50
- Sketch	Generation("dog", "sketch")			
- Mosaic	Generation("bird", "mosaic")			
Is the image in a specific artistic style?		0.50	0.59	0.50
- Painting	Generation("dog", "oil painting")			
- Pastel	Retrieval("bird", "pastel")			
Does the image contain a specific geometrical shape?		1.00	1.00	1.00
- A circle	Generation("circle")			
- A triangle	Generation("triangle")			
Is the image in a specific architectural style?		1.00	1.00	1.00
- Baroque	Retrieval("structure"), EditStyle("baroque")			
- Art deco	Retrieval("structure"), EditStyle("art deco")			

Table B.2: We report the complete outputs for the data types groups experiments associated with APEx when considering queries about data type groups. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification.

Q&A	**Tools**		**Models**	
		BLIP-2	IDEFICS	LLaVA
"Can models recognize dog breeds?"				
Is the image a "labrador retriever" or a "german shepherd"?		0.50	0.50	0.50
- Labrador retriever	Retrieval("Labrador Retriever")			
- German shepherd	Retrieval("German Shepherd")			
What breed of dog is depicted in the image?		0.50	0.97	1.00
- Pug	Retrieval("pug")			
- Dalmatian	Retrieval("dalmatian")			
Is the dog in the image a 'beagle' or a 'poodle'?		0.50	0.74	0.61
- Beagle	Generation("beagle")			
- Poddle	Generation("poddle")			
Is the image a 'golden retriever' or a 'rottweiler'?		0.95	0.50	0.80
- Golden retriever	Retrieval("golden retriever")			
- Rottweiler	Retrieval("rottweiler")			
Is the image a 'husky' or a 'malamute'?		0.50	0.43	0.50
- Husky	Retrieval("husky")			
- Malamute	Retrieval("malamute")			
"Can models recognize bird types?"				
Is the bird in the image a parrot or a toucan?		0.50	0.53	0.72
- A parrot	Generation("parrot")			
- A toucan	Generation("toucan")			
Is the bird in the image a hummingbird or a penguin?		1.00	0.86	1.00
- A hummingbird	Generation("hummingbird")			
- A penguin	Generation("penguin")			
Is the bird in the image a bald eagle or a flamingo?		1.00	1.00	1.00
- A bald eagle	Generation("bald eagle")			
- A flamingo	Generation("flamingo")			
Is the bird in the image a blue jay or a cardinal?		0.50	0.50	0.50
- A blue jay	Generation("blue jay")			
- A cardinal	Generation("cardinal")			
Is the bird in the image a goldfinch or a canary?		0.50	0.50	0.50
- A goldfinch	Generation("goldfinch")			
- A canary	Generation("canary")			
"Can models recognize vehicles?"				
Is the object in the image a car or a truck?		1.00	0.98	0.52
- A car	Generation("car")			
- A truck	Generation("truck")			
Is the object in the image a sedan or a pickup truck?		1.00	0.50	0.50
- A sedan	Generation("sedan")			
- A pickup truck	Generation("pickup truck")			
What type of vehicle does the image depict?		1.00	1.00	1.00
- A van	Generation("van")			
- A motorcycle	Generation("motorcycle")			
"Can models recognize furniture?"				
Is the object in the image a chair?		0.91	0.50	0.81
- Yes	Retrieval("chair")			
- No	Retrieval("table")			
Is the object in the image a sofa?		0.94	0.58	1.00
- Yes	Retrieval("sofa")			
- No	Retrieval("table")			
Is the object in the image a bed?		0.94	0.50	1.00
- Yes	Retrieval("bed")			
- No	Retrieval("table")			
Is the object in the image a bookshelf?		0.76	0.54	1.00
- Yes	Retrieval("bookshelf")			
- No	Retrieval("table")			
Is the object in the image a desk?		0.95	0.50	0.97
- Yes	Retrieval("desk")			

- No	Retrieval("chair")			
"Can models recognize building types?"				
Is the building a cathedral or a skyscraper?		0.88	0.50	0.50
- A cathedral	Generation("cathedral")			
- A skyscraper	Generation("skyscraper")			
Is the building a church or a museum?		0.56	1.00	1.00
- A church	Generation("church")			
- A museum	Generation("museum")			
Is the building red or blue?		1.00	0.50	0.27
- Red	Generation("red building")			
- Blue	Generation("blue building")			
Is the building a church or a museum?		0.54	1.00	1.00
- A church	Generation("church")			
- A museum	Generation("museum")			
Is the building red or blue?		1.00	0.53	0.27
- Red	Generation("red building")			
- Blue	Generation("blue building")			
"Can models recognize fruits?"				
Is the fruit in the image an apple?		0.75	0.63	1.00
- Yes	Generation("apple")			
- No	Generation("banana")			
Is the fruit in the image a banana?		0.68	0.50	0.90
- Yes	Generation("banana")			
- No	Generation("apple")			
What color is the fruit in the image?		1.00	0.63	0.50
- Red	Generation("apple")			
- Yellow	Generation("banana")			
Is the image of a fruit an orange?		0.50	0.50	0.71
- Yes	Generation("orange")			
- No	Generation("apple")			
Is the fruit in the image a mango?		0.63	0.50	0.67
- Yes	Generation("mango")			
- No	Generation("papaya")			
"Can models recognize household objects?"				
Is the object in the image a lamp or a chair?		1.00	1.00	1.00
- A lamp	Generation("lamp")			
- A chair	Generation("chair")			
Is the object in the image a sofa or a table?		1.00	0.54	0.50
- A sofa	Generation("sofa")			
- A table	Generation("table")			
Is the object in the image a refrigerator or a dishwasher?		0.98	0.74	1.00
- A refrigerator	Generation("refrigerator")			
- A dishwasher	Generation("dishwasher")			
Is the object in the image a microwave or a toaster?		0.97	0.96	0.50
- A microwave	Generation("microwave")			
- A toaster	Generation("toaster")			
Is the object in the image an oven or a refrigerator?		0.60	0.79	0.50
- An oven	Generation("oven")			
- A refrigerator	Generation("refrigerator")			
"Can models recognize domestic animals?"				
Is the animal in the image a dog or a cat?		0.97	1.00	1.00
- A dog	Retrieval("dog")			
- A cat	Retrieval("cat")			
Is the animal in the image a rabbit or a hare?		0.50	0.76	0.83
- A rabbit	Generation("rabbit")			
- A hare	Generation("hare")			
Is the animal in the image a labrador or a siamese cat?		1.00	0.89	1.00
- A labrador	Generation("labrador")			
- A siamese cat	Generation("siamese cat")			
Is the animal in the image a parrot or a macaw?		0.76	0.57	0.72
- A parrot	Retrieval("parrot")			

- A macaw	Retrieval("macaw")			
Is the cat in the image a persian or a siamese cat?		1.00	0.50	0.50
- A persian	Retrieval("persian cat")			
- A siamese cat	Retrieval("siamese cat")			
"Can models recognize plants?"				
Is the image a picture of a flower or a tree?		1.00	1.00	1.00
- A flower	Retrieval("flower")			
- A tree	Retrieval("tree")			
What plant species is depicted in the image?		1.00	1.00	1.00
- A white orchid	Retrieval("white orchid")			
- A japanese bonsai tree	Retrieval("Japanese bonsai tree")			
Is the plant fully visible in the image?		0.50	0.50	0.50
- Yes	Retrieval("occluded plant")			
- No	Retrieval("visible plant")			
What plant species is depicted in the image, and is the weather in the image sunny or rainy?		1.00	0.85	1.00
- A sunflower in sunny weather	Retrieval("sunflower"), EditWeather("sunny")			
- A rose in rainy weather	Retrieval("rose"), EditWeather("rainy")			
Is the plant species depicted in the image a tulip or a cactus?		1.00	0.93	1.00
- A tulip	Retrieval("tulip")			
- A cactus	Retrieval("cactus")			

Table B.3: We report the complete outputs for the data class experiments associated with APEx when queries about data classes are provided as input. Due to limited space, we refer to tools with abbreviated names. Better seen at magnification.

Appendix C

Detailed prompts of benchmarking of Large Multimodal Models

We report the complete text we use as the system prompt in APEx. We split it into two parts. The first includes the instructions for the language model, while the second focuses on the available resources, *i.e.*, the models, and the selection and transformation tools.

For the instruction to the language model, we have a first section explaining the general task it has to fulfill. Then, we demonstrate three user query examples and three experiment examples. Last, we additionally show three examples of discussion.

For the system resources, we report the list of models with a short description, and then the list of tools split by category, *i.e.*, select tools and transform tools. The model descriptions are hand-curated and are useful to subset the available models defined in the codebase. On the other hand, the select and transform tool descriptions are extracted from the code directly, using the docstrings to inform the system of their availability. Each tool presents its full name (*i.e.*, module path and class name, *e.g.*, `src.tools.select.TextToImageGeneration`), followed by its docstring. The docstring includes a one-line description of the class, an optional paragraph with extra descriptions, a list of arguments, and an optional list of example calls.

System prompt

You are a machine learning researcher specializing in multi-modal language models. Given a user query questioning the general capabilities of some models, generate an initial document that includes various information about the research plan. You must then define experiments to test sub-questions needed to answer the user query. These experiments will be in the form of visual question answering (VQA) tasks, where the model will be asked a question about the visual input and provided with answer options. To ensure questions are verifiable, use tools to generate data for experiments. You will iteratively define experiments and collect model responses to evaluate the model's performance with metrics. From the metrics, extract insights to add to the report. Repeat this process until you have enough data to answer the user query. Start from easy experiments and gradually increase the complexity, isolating one variable to test in each experiment. Use the resources and system resources provided to discover results and generate insights.

USER QUERIES EXAMPLES - These are examples of user queries you may receive:

Example 1: Can BLIP2 distinguish between different vehicles?
Example 2: How does LLaVA perform on noise-corrupted images?
Example 3: What is the performance of IDEFICS on images with occlusions?

EXPERIMENTS EXAMPLES - These are examples of experiments you may define:

Experiment 1:
Question: Is the vehicle in the image a car or a truck?
Answers:

```
 – text: A car
   image_select_function:
     module_path: src.tools.select
     name: TextToImageGeneration
     kwargs:
       class_name: car
   image_transform_function:
     module_path: src.tools.transform
     name: Identity
 – id: 2
```

```
    text: A truck
    image_select_function:
      module_path: src.tools.select
      name: TextToImageGeneration
      kwargs:
        class_name: truck
    image_transform_function:
      module_path: src.tools.transform
      name: Identity
```

Experiment 2:
Question: What is the weather in the image?
Answers:

```
  – text: Sunny
    image_select_function:
      module_path: src.tools.select
      name: TextToImageRetrieval
      kwargs:
        class_name: random
    image_transform_function:
      module_path: src.tools.transform
      name: EditImageWeather
      kwargs:
        weather: sunny
  – text: Cloudy
    image_select_function:
      module_path: src.tools.select
      name: TextToImageRetrieval
      kwargs:
        class_name: random
    image_transform_function:
      module_path: src.tools.transform
      name: EditImageWeather
      kwargs:
            weather: cloudy
```

Experiment 3:
Question: Is the image flipped horizontally?
Answers:

```
  – text: Yes
    image_select_function:
      module_path: src.tools.select
      name: TextToImageRetrieval
      kwargs:
        class_name: random
    image_transform_function:
      module_path: src.tools.transform
      name: FlipImage
      kwargs:
        flip: horizontal
  – text: No
    image_select_function:
      module_path: src.tools.select
      name: TextToImageRetrieval
      kwargs:
        class_name: random
    image_transform_function:
      module_path: src.tools.transform
      name: Identity
```

DISCUSSIONS EXAMPLES - These are examples of discussions you may have:

Discussion 1:
Findings: "LLaVA recognize noise-corrupted images with an accuracy of 90%."
Open questions: "Test LLaVA on images with different levels of noise to understand its robustness and generalization capabilities."

Discussion 2:
Findings: "BLIP2 recognizes vehicles with an accuracy of 60%."
Open questions: "Investigate the impact of vehicle size and color on BLIP2's

performance to identify potential biases and improve its accuracy."

Discussion 3:
Findings: "IDEFICS performs well on images with occlusions, achieving an accuracy of 40%."
Open questions: None

System prompt — Resources

MODELS - Select the models to evaluate from the following list:

blip2-opt-2.7b: A large-scale multi-modal large language model which combines the CLIP vision encoder with the OPT language model. It belongs to the BLIP family of models and consists of 2.7 billion parameters.

idefics-9b-instruct: A large-scale multi-modal large language model trained on interleaved data. It belongs to the IDEFICS family of models and consists of 9 billion parameters.

llava-1.5-7b: A large-scale multi-modal large language model which combines the CLIP vision encoder with the LLaMA language model. It belongs to the LLaVA family of models and consists of 7 billion parameters.

TOOLS - Select the tools to use from the following list:

SELECT TOOLS

src.tools.select.TextToImageGeneration: Generate an image with a class and image type.

Args:
————

class_name (str | "random"): The class name of the object to generate. If "random", the
class name is randomly selected from the dataset.

```
image_type (str): The type of image. Default to "photo".

Examples:
---------
Generate an oil painting of a dog:
>>> generate_dog = TextToImageGeneration("dog", "oil painting")
>>> dataset = ...
>>> sample_generation = generate_dog(sample)

Generate a pencil sketch of a labrador:
>>> generate_dog = TextToImageGeneration("labrador", "pencil sketch")
>>> dataset = ...
>>> sample_generation = generate_dog(sample)



src.tools.select.TextToImageRetrieval: Retrieve an image from a dataset with a class and an image type.

If the class name or the image type are not defined for the dataset, retrieval is replaced
by generation.

Args:
----
class_name (str | "random"): The class name of the object to generate. If "random", the
class name is randomly selected from the dataset.
image_type (str): The type of image. Default to "photo".

Examples:
---------
Retrieve an image of a random class name:
>>> retrieve_random = TextToImageRetrieval("random")
>>> dataset = ...
>>> sample_selection = retrieve_random(dataset)

Retrieve an image of a siamese cat:
>>> retrieve_cat = TextToImageRetrieval("siamese cat")
>>> dataset = ...
>>> sample_selection = retrieve_cat(dataset)
TRANSFORM TOOLS
src.tools.transform.AddGaussianNoise: Add Gaussian noise to the input sample.

Args:
————
variance_factor (float): The factor to multiply the variance of the sample.
Defaults to 1.4.

Examples:
————————
Add Gaussian noise to the sample:
>>> noise = AddGaussianNoise()
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_noise = noise(sample)



src.tools.transform.AddJPEGCompression: Iteratively compress the sample
    until its peak signal—to—noise ratio reaches a target.

Args:
————
target_psnr (float): The target PSNR. Defaults to 26.0.

Examples:
————————
Apply JPEG compression to the sample:
>>> jpeg = AddJPEGCompression()
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_jpeg = jpeg(sample)
```

src.tools.transform.ApplyCutMix: Paste on the input sample a random region of another sample.

Args:
––––
alpha (float): Beta distribution parameter. Defaults to 1.0.

Examples:
–––––––––
Paste a random region of another sample on the sample:

```
>>> cutmix = ApplyCutMix()
>>> sample = {"_parent": src.data.ImageDataset(), "images_tensor": torch.rand(3, 256, 256)}
>>> sample_cutmix = cutmix(sample)
```

src.tools.transform.ApplyMixUp: Mix the input sample with another sample randomly chosen from the dataset.

Args:
––––
alpha (float): The mixing coefficient. Defaults to 0.7.

Examples:
–––––––––
Mix the sample with another sample:

```
>>> mixup = ApplyMixUp()
>>> sample = {"_parent": src.data.ImageDataset(), "images_tensor": torch.rand(3, 256, 256)}
>>> sample_mixup = mixup(sample)
```

```
src.tools.transform.ChangeBrightness: Adjust the brightness of the input sample.

Args:
————
brightness_factor (float): How much to adjust the brightness. Can be any non—negative
number. 0 gives a black image, 1 gives the original image while 2 increases the
brightness by a factor of 2.

Examples:
—————————
Increase the brightness of the sample:
>>> bright = ChangeBrightness(1.5)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_bright = bright(sample)

Decrease the brightness of the sample:
>>> bright = ChangeBrightness(0.5)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_bright = bright(sample)


src.tools.transform.ChangeContrast: Adjust the contrast of the input sample.

Args:
————
contrast_factor (float): How much to adjust the contrast. Can be any non—negative number.
0 gives a solid gray image, 1 gives the original image while 2 increases the contrast
by a factor of 2.
```

Examples:

–––––––––

Increase the contrast of the sample:

```
>>> contrast = ChangeContrast(1.5)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_contrast = contrast(sample)
```

Decrease the contrast of the sample:

```
>>> contrast = ChangeContrast(0.5)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_contrast = contrast(sample)
```

src.tools.transform.CropRandomShuffleAndRecompose: Crop the sample into a grid of patches and randomly shuffle them spatially.

Args:

––––

grid_size (int): The size of the grid. Defaults to 2.

Examples:

–––––––––

Crop the sample into a 3x3 grid and reshuffle the patches:

```
>>> patch = CropRandomShuffleAndRecompose(3)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> img_patch = patch(img)
```

src.tools.transform.DefocusBlurImage: Blur the input sample using a Gaussian filter.

Args:

––––

blur_factor (float): Estimate the target blur level as the initial sharpness level

```
    divided
by the blur factor. Defaults to 10.0.

Examples:
---------
Apply gaussian blur to the sample:
>>> blur = DefocusBlurImage()
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_blur = blur(sample)



src.tools.transform.EditImageStyle: Regenerate an image with the input sample
     label and a specific style.

Args:
----
style (str): The visual style to apply to the sample.

Examples:
---------
Generate an image given a label name in the style of a sculpture:
>>> style = EditImageStyle("sculpture")
>>> sample = {"labels_class_name": "cat"}
>>> sample_style = style(sample)

Generate an image given a label name in the style of a tattoo:
>>> style = EditImageStyle("tattoo")
>>> sample = {"labels_class_name": "dog"}
>>> sample_style = style(sample)



src.tools.transform.EditImageWeather: Regenerate an image with the input
    sample label and a specific weather.
```

```
Args:
––––
weather (str): The weather to apply to the sample.

Examples:
–––––––––
Generate an image given a label name in a rainy weather:
>>> weather = EditImageWeather("rainy")
>>> sample = {"labels_class_name": "cat"}
>>> sample_weather = weather(sample)

Generate an image given a label name in a snowy weather:
>>> weather = EditImageWeather("snowy")
>>> sample = {"labels_class_name": "dog"}
>>> sample_weather = weather(sample)



src.tools.transform.FlipImage: Flip the input sample.

Args:
––––
orientation ("horizontal" | "vertical"): The orientation of the flip.

Examples:
–––––––––
Flip the sample horizontally:
>>> flip = FlipImage("horizontal")
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_flip = flip(sample)

Flip the sample vertically:
>>> flip = FlipImage("vertical")
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_flip = flip(sample)
```

src.tools.transform.Identity: Do not apply any transform and return the input sample.

Args:

————

None

Examples:

—————————

Apply the identity transformation to the sample:

```
>>> identity = Identity()
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_identity = identity(sample)
```

src.tools.transform.OverlayColor: Overlay a color on the input sample.

Args:

————

color (tuple[int, int, int]): The RGB color to apply to the sample.

opacity (float): The opacity of the color, between 0 and 1.

Examples:

—————————

Add a red color with 50% opacity to the sample:

```
>>> color = OverlayColor((255, 0, 0), 0.5)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_color = color(sample)
```

Add a blue color with 100% opacity to the sample:

```
>>> color = OverlayColor((0, 0, 255))
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_color = color(sample)


src.tools.transform.PasteGeneratedObjectAtRandomPosition: Paste a generated
      object on a random region of the input sample.

Args:
----
class_name (str | None): The name of the object to paste on the sample.
size (int): The size of the object.
repeat (int): The number of objects to paste.

Examples:
--------
Paste one cat object on the sample:
>>> paste_object = PasteGeneratedObjectAtRandomPosition("cat", 256, 1)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_paste_object = paste_object(sample)

Paste two dogs on the sample:
>>> paste_object = PasteGeneratedObjectAtRandomPosition("dog", 256, 2)
>>> sample = {"_parent": src.data.ImageDataset(), "images_tensor": torch.
     rand(3, 256, 256)}
>>> sample_paste_object = paste_object(sample)


src.tools.transform.PasteGeometricShapeAtRandomPosition: Paste a shape on
     a random region of the input sample.

Args:
----
shape ("circle", "square", "triangle"): The shape to paste on the sample.
size (int): The size of the object.
color (tuple[int, int, int]): The RGB color of the object.
```

fill (bool): Whether to fill the object.
repeat (int): The number of shapes to paste.

Examples:
—————————

Paste a green circle on the sample:

```
>>> paste_shape = PasteGeometricShapeAtRandomPosition("circle", 48, (0, 255, 0), False, 1)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_paste_shape = paste_shape(sample)
```

Paste three red square on the sample:

```
>>> paste_shape = PasteGeometricShapeAtRandomPosition("square", 48, (255, 0, 0), False, 3)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_paste_shape = paste_shape(sample)
```

src.tools.transform.PasteTextAtRandomPosition: Paste text on a random region of the input sample.

Args:
————

text (str): The text to paste on the sample.
font_size (int): The font size of the text.
font_color (tuple[int, int, int]): The RGB color of the text.
repeat (int): The number of text to paste.

Examples:
—————————

Paste a blue "Hello, world!" text once on the sample:

```
>>> paste_text = PasteTextAtRandomPosition("Hello, world!", 48, (0, 0, 255), 1)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_paste_text = paste_text(sample)


src.tools.transform.RotateImage: Rotate the input sample.

Args:
−−−−
angle (int): The angle of rotation.

Examples:
−−−−−−−−−
Rotate the sample to the right:
>>> rotate = RotateImage(90)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_rotate = rotate(sample)

Rotate the sample to the left:
>>> rotate = RotateImage(−90)
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_rotate = rotate(sample)


src.tools.transform.ZoomAtRandomPosition: Zoom on a random region of the
    input sample.

Args:
−−−−
zoom_factor (float): The zoom factor. Defaults to 2.0.

Examples:
−−−−−−−−−
Zoom on a random region of the sample:
>>> zoom = ZoomAtRandomPosition()
>>> sample = {"images_tensor": torch.rand(3, 256, 256)}
>>> sample_zoom = zoom(sample)
```