Converging Pathways: Structured Visual Representations with Multi-Task Learning

Thesis by
Shikun Liu

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy in Computing

IMPERIAL

Imperial College of Science, Technology and Medicine
London, England

(Completed 30th April 2024)

Statement of Originality

The research presented in this thesis is my own and all external content and information has been referenced appropriately.

Abstract

Human cognition is inherently adept at concurrently managing multiple tasks, and efficiently leveraging past experiences to acquire new skills. In machine learning, multi-task learning is the learning paradigm that seeks to emulate this cognitive capability by learning shared features from related tasks to improve training efficiency. However, the straightforward application of multi-task learning may lead to sub-optimal performance, owing to the variations in task-specific complexity and distinct training objectives.

In this thesis, we conduct a comprehensive investigation of multi-task learning within the domain of computer vision, resulting in novel solutions in multi-task learning for constructing *structured visual representations* — a foundational building block for a wide range of applications, spanning from visual perception and object recognition to vision-based control and reasoning.

Our research includes a spectrum of training strategies and optimisation methods, all intricately designed to enhance the efficacy of structured visual representations. Specifically, we delve into three critical dimensions: i) the design strategy of harnessing multi-task relationships to improve computer vision model performance, ii) the creation of auxiliary tasks to improve computer vision model generalisation, and iii) the utilisation of multi-task knowledge within pre-trained experts to improve open-ended visual reasoning.

We introduce a series of multi-task learning frameworks to address these research questions and showcase their effectiveness in improving the generalisation, training efficiency, and interpretability of computer vision systems. Through this comprehensive exploration of multi-task learning and its implications, we aim to contribute to the development of more intelligent and versatile systems for challenging real-world applications.

Acknowledgments

First and foremost, I would like to extend my deepest gratitude to my Ph.D. advisors, Prof. Andrew J. Davison and Prof. Edward Johns. Over the span of six years — from my M.Res. to my Ph.D. — your invaluable mentorship has laid the foundation for both my academic and personal growth. Andy, thank you for always reminding me to focus on long-term impact and offering me remarkable research freedom to explore. Ed, thank you for your meticulous hands-on guidance in turning my chaotic thoughts into clear, organised papers and presentations. Being one of your first Ph.D. students is a privilege I deeply cherish. Our many discussions, meetings, and (of course) disagreements have not only shaped my research skills but have also my taste and vision as to become an independent AI researcher today. Additionally, I am grateful to my thesis examiners Prof. Yingzhen Li and Prof. Hakan Bilen. Your constructive feedbacks have greatly improved my thesis.

A sincere thank you to all the remarkable colleagues and friends who have been part of my Master's and Ph.D. journey. Specifically, special appreciation goes to Iosifina Pournara for her indispensable role in ensuring the smooth functioning of the lab, as well as for her support and encouragement. To Stefan Leutenegger, Robert Lukierski, John McCormac, Michael Bloesch, and Fabian Falck — thank you for always backing me up during the early stages of my research career. You've all become Dyson legends. And to Talfan Evans, Stephen James, Charlie Houseago, Shuaifeng Zhi, Raluca Scona, Kentaro Wada, Zoe Landgraf, Tristan Laidlow, Binbin Xu, Daniel Lenton, Dorain Hennings, Edgar Sucar, and Joseph Ortiz — thank you for keeping me sane during the challenging times of COVID and sharing the most core memory during my Ph.D. Finally, a special shout-out to Ignacio Alzugaray-Lopez, Gwangbin Bae, Callum Rhodes, Marwan Taher, Hide Matsuki, Seth Nabarro, Aalok Patwardhan, Xin Kong, Kirill Mazur, Eric Dexheimer, Ivan Kapelyukh, Riku Murai, Kamil Dreczkowski, Norman Di Palo, Vitalis Vosylius, Georgios Papagiannis, Pietro Vitiello, Yifei Ren, and the rest of the current lab crew for always having my back. All your help and friendship have meant the world to me. From ping-pong and bowling

battles to hotpot feasts and countless late nights chasing CVPR deadlines, I'll cherish every moment we've shared. Thank you for the laughs, the inspirations, and the endless memories.

I have been fortunate to have two research internship experiences during my Master's and Ph.D. Thanks to Zhe Lin, Yilin Wang, Jianming Zhang, and Federico Perazzi for hosting me at Adobe Research. It was my very first experience in the San Francisco Bay Area. I still remember the never-ending sunny days as a stark contrast to the luxury in London. Thanks to Jim Fan, Zhiding Yu, Chaowei Xiao, and Anima Anandkumar for hosting me at Nvidia Research. Thank you for putting in your best effort to make this remote internship interesting during COVID.

Besides, I am really thankful for all the friends I met in London. Thanks to Danlin Peng, Xiaoqing Huang, Linkun Geng, Tianjing Xu, Qinyi Liu, Chengxiao Ge, Wei Wang, Guosheng Hu, Wei-Hong Li, Hengyi Wang and Jingwen Wang. You've added spice to my life outside the lab. Thanks for being part of my life.

Finally, I would like to express my sincere gratitude to my family who offered financial and mental support throughout my 24 years of education. The final special thanks goes to my wife, Shan Huang. Thank you for making sacrifices and supporting me every step of the way, sharing in all the joys and sorrows, dreams and fears, triumphs and failures.

The work presented in this thesis is generously funded by Dyson Ltd.

Sincerely,
Shikun Liu 刘诗昆

Nomenclature

Symbols & Notations

$\mathcal{X}$	input domain
$\mathcal{Y}$	output domain
X	input dataset
Y	output dataset
x_n	single data point
y_n	single output label
$\hat{y}_n$	predicted model output on input x_n
ϵ	a random variable
λ	task weighting
τ	temperature
$f_\theta(\cdot)$	a function f with parameters θ
$\mathcal{N}(\cdot)$	Gaussian (normal) distribution
$\mathcal{B}(\cdot)$	Bernoulli distribution
$\|\cdot\|$	norm-1 distance
$\|\cdot\|_2$	norm-2 distance (Euclidean norm)
$[\cdot;\cdot]$	concatenation

NN	Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
MTL	Multi-Task Learning
TL	Transfer Learning
AL	Auxiliary Learning
RL	Reinforcement Learning
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
RAE	Relative Absolute Error
e.g.	exempli gratia (for example)
i.e.	id est (that is)
s.t.	such that
w.r.t.	with respect to

Contents

Abstract ◆ iv

Acknowledgments ◆ v

Nomenclature ◆ vii

1 Introduction and Background ◆ 1

1.1 Artificial Intelligence and Machine Learning ◆ 1

1.2 Tasks: Definitions and Applications ◆ 3

1.3 Learning Strategies with Multiple Tasks ◆ 8

1.4 Research Direction and Contribution ◆ 10

2 Preliminaries ◆ 14

2.1 Advances in Neural Architecture Design ◆ 15

2.2 Multi-Task Neural Architectures ◆ 20

2.3 Multi-Task Optimisation Strategies ◆ 23

2.4 Interpretability and Task Relationships ◆ 26

3 Exploring Task Relationships with Automated Weightings ◆ 30

3.1 Rethinking Multi-Task Relationships: Static or Dynamic? ◆ 30

3.2 Related Work ◆ 31

3.3 Auto-λ: Unifying Multi-Task and Auxiliary Learning ◆ 33

3.4 Experiments ◆ 35

3.5 Visualisations and Interpretability of Task Relationships ◆ 41

3.6 Robustness and Ablation Analysis ◆ 45

3.7 Conclusions, Limitations and Discussions ◆ 48

4 Exploring Semantic Relationships with Contrastive Learning ◆ 50
4.1 The Challenge of Semantic Segmentation ◆ 51
4.2 Related Work ◆ 51
4.3 ReCo: Regional Contrast Learning for Semantic Segmentation ◆ 53
4.4 Experiments ◆ 58
4.5 Ablation Study on Hyper-Parameters and Training Details ◆ 65
4.6 Visualisations and Interpretability of Class Relationships ◆ 66
4.7 Conclusion, Limitations and Discussions ◆ 69

5 Self-Supervised Generalisation with Meta Auxiliary Learning ◆ 70
5.1 Understanding Auxiliary Learning with Semantic Complexity ◆ 70
5.2 Related Work ◆ 73
5.3 MAXL: Self-Supervised Auxiliary Learning for Image Classification ◆ 74
5.4 Experiments ◆ 79
5.5 Visualisations of Generated Auxiliary Knowledge ◆ 82
5.6 Conclusions, Limitations and Discussions ◆ 85

6 Vision-Language Reasoning with Multi-Task Experts ◆ 87
6.1 Breaking Down Vision-Language Reasoning ◆ 87
6.2 Related Work ◆ 88
6.3 Prismer: Unifying Multi-Task Experts for Vision-Language Reasoning ◆ 90
6.4 Experiments ◆ 96
6.5 Learning Strategy and Utility Analysis of Multi-Task Experts ◆ 102
6.6 Ablation Study on Architecture Design and Training Details ◆ 104
6.7 Conclusions, Limitations and Discussion ◆ 106

7 Conclusions and Future Works ◆ 108
7.1 Summary of Contributions ◆ 108
7.2 Future Works ◆ 110

1

Introduction and Background

In this chapter, we lay the foundations and motivations of this thesis. We first describe the pivotal role of tasks and multi-task learning in the advancement of general-purpose AI (Section 1.1). Then, we delve into the fundamental task definitions, providing examples and showcasing applications of multi-task learning in different domains (Section 1.2). We introduce the machine learning strategies designed to enhance generalisation by sharing features among tasks (Section 1.3). Finally, we outline our research contribution and present a roadmap of this thesis (Section 1.4).

1.1 Artificial Intelligence and Machine Learning

Tasks play a fundamental role in AI and machine learning, representing the specific problems, challenges, or objectives that AI systems aim to address or accomplish. These tasks span a wide spectrum, ranging from relatively straightforward ones such as image classification and language translation to more intricate undertakings, such as mastering video games and making medical diagnoses. Each task is accompanied by its unique set of intricacies, requiring *specialised design* for effective solutions — the strategies devised for one task may not readily translate to another.

Historically, the field of AI has predominantly revolved around *single-task learning*, where systems are meticulously engineered and trained to excel in a single pre-defined task. This approach often necessitates the design of hand-crafted features and algorithms tailored

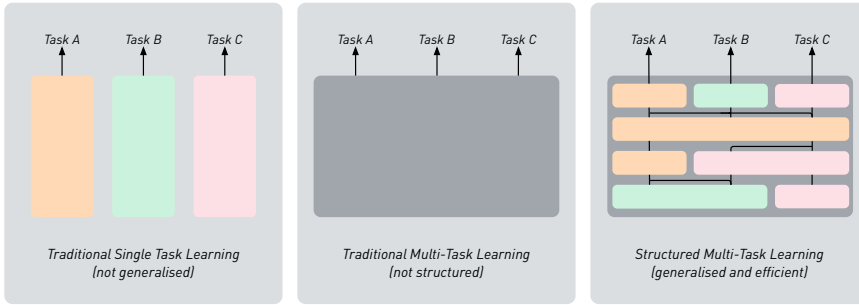


Figure 1.1. Designing single-task learning and multi-task learning systems. *Traditional single-task learning* requires designing task-specific solutions tailored for each particular task and lacks generalisation to new tasks. *Traditional multi-task learning* trains a single network for all tasks concurrently, but may lead to inferior performance in some tasks due to the inherent differences in complexity and learning strategies required for each individual task. *Structured multi-task learning* can leverage shared features from related tasks in a single network, while also considering the relationships between tasks, leading to better generalisation and improved interpretability.

specifically to the task at hand or, alternatively, requires extensive data for end-to-end training. While these methods can yield high accuracy within the target task, they often lack the ability to generalise to other tasks or environments without substantial human intervention and re-engineering.

Recognising the limitations of single-task learning, the concept of *multi-task learning* (MTL) has gained prominence. In a seminal study [Car97], MTL is described as an inductive transfer mechanism whose principle goal is to improve generalisation performance. It does this by training tasks in parallel while using a shared representation. The fundamental premise is that knowledge acquired from one task can contribute to improved generalisation on other related tasks. However, merely grouping all tasks together for end-to-end training is often insufficient. It is essential to consider the *relationships* between these tasks thoughtfully to ensure that the model effectively leverages shared knowledge. Otherwise, the performance of some tasks could be inferior to what can be achieved with single-task learning, given the inherent differences in complexity for learning each individual task.

In response to the limitations of these learning strategies, we propose that *structured multi-task learning* could serve as a promising alternative. Structured multi-task learning seeks to enhance the generalisation of multi-task models by harnessing *structured representations* acquired through an understanding of task relationships. These structured representations are designed to capture the inherent structure within the data, allowing the model to learn more efficiently and generalise more effectively.

By enabling AI systems to learn a diverse set of skills and tasks concurrently and efficiently, we can push the boundaries of what AI is capable of. These systems become less rigid and more adaptable, with the ability to creatively apply their knowledge to new and uncharted settings. This holistic development of flexible intelligence represents a significant milestone on the journey towards achieving more advanced and capable AI.

1.2 Tasks: Definitions and Applications

The definition of a task can be thought of as a *fundamental philosophy of grouping data* that shares common characteristics or attributes, with the aim of enabling the AI systems to learn and make predictions or decisions based on this data. This grouping of data is crucial in defining and delineating tasks, as it influences how AI models are trained, their performance, and their ability to generalise [Car97].

Let's consider a scenario where we have a set of images containing cats and dogs. One natural task in this context is the classification of these images into two categories: "cat" and "dog". By grouping the cat and dog images together, we create a task that focuses on distinguishing between these specific animal categories. However, the concept of a task can expand beyond this binary classification. When we introduce more animal images, such as lions, tigers, elephants, and giraffes, we move into a broader category: animal classification. In this case, the task includes more data, broadening the model's understanding and enabling it to discriminate between various animal species.

The concept of tasks doesn't stop at data grouping; it extends to the *relationships* between these tasks. The nature of these relationships can vary widely, influencing the effectiveness of the learning system. In our previous example, we can observe the hierarchy and interplay between tasks. The "cat and dog classification" task is a subtask of the broader "animal classification" task. A structured multi-task learning system can inherit knowledge from the more specialised task, leading to more efficient learning compared to a single-task learning system that learns from scratch. This hierarchical relationship between tasks is just one aspect of task interdependence. Tasks can be interconnected in various ways, ranging from being subtasks of one another, mutually related to being entirely unrelated [ZY18].

Understanding the structure of tasks is critical in achieving a successful multi-task learning system. When tasks are well-defined and their relationships are carefully considered, it can enhance data efficiency, promote better generalisation, and improve interpretability. By recognising how tasks relate to one another, we can optimise the sharing of knowledge and resources, ultimately resulting in more capable and versatile AI systems.

Tasks in Computer Vision

In the field of computer vision, scene understanding is a critical area that aims to endow machines with the ability to comprehend and interpret visual scenes as humans do. One prominent subset of scene understanding [VGVG+21] involves *pixel-level dense prediction tasks*. These tasks involve taking an input image and producing detailed, fine-grained predictions by assigning labels to each pixel in the image. This pixel-level granularity enables machines to capture intricate details and nuances present within the visual content.

Pixel-level dense prediction tasks encompass a diverse array of challenges, each marked by unique goals and applications. Despite sharing the common input of an image, these tasks yield distinct outputs tailored to their specific objectives, collectively contributing to a more comprehensive grasp of the visual world. For example, *semantic segmentation* [HZG20] involves classifying pixels within an image into predefined categories, which equips machines to comprehend the *arrangement and composition* of scenes; *object detection* [ZZXW19] entails identifying and localising multiple objects within an image to let the machines not only determine the *presence* of objects but also their *precise locations*; *depth estimation* [MMFY21] revolves around inferring the distance of objects from the camera in a given image, which facilitates the creation of 3D models from 2D images, leading to an improved *spatial understanding*; and *surface normal estimation* [YCK92] involves determining the orientation of surfaces in an image, providing insights into the *geometry* of the scene.

Pixel-level dense prediction tasks play a pivotal role in enhancing machine perception of visual data, empowering machines to not only recognise objects and regions but also understand their spatial relationships and finer intricacies. A unified multi-task learning system that can perform all these tasks simultaneously would be able to achieve a holistic understanding of the visual world with strong robustness, and lays the foundation for a wide spectrum of industrial applications, such as autonomous driving.

Tasks in Natural Language Processing

Natural Language Processing (NLP) includes a wide range of tasks that cater to various aspects of language understanding and generation, such as *machine translation (English → French)* [Sta20]: I want to eat salad. → Je veux manger de la salade.; *sentiment analysis* [MHK14]: I am excited to see the Oppenheimer movie! → Positive.; and sometimes also involve machine languages like *code completion* [RVY14]: Revert a string `s` → `s[::-1]`.

By dividing and parametrising the input and output language sequences into smaller computational units (usually tokens), the problem becomes more manageable, and deep learning

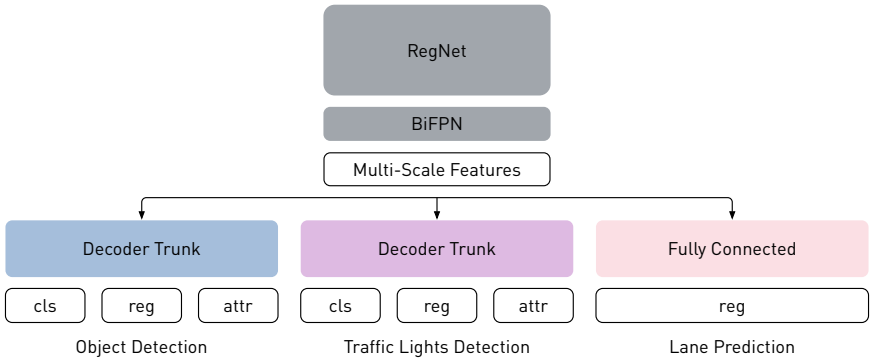


Figure 1.2. Tesla FSD system [Tes22] with multi-task perception. Tesla's Full Self-Driving (FSD) system represents a paradigm of vision-centric multi-task learning, adeptly predicting tasks including object detection, traffic light identification, and lane prediction, all within a singular, unified network.

models like recurrent neural networks [SP97, HS97] or transformer models [VSP+17] can be used to learn the relationships between the input and output tokens. As we may treat both the input and output as sequences, we may simply re-formulate all language understanding tasks as *sequence-to-sequence learning* tasks, where the goal is to learn a mapping from one sequence to another. Unlike computer vision, where different tasks may require very different data representations and processing pipelines, NLP tasks can often be expressed using similar tokenisation techniques. This makes the multi-task learning problem in NLP more structured and amenable to a unified framework.

Multi-task learning has reshaped how modern language models are designed, contributing to the development of very powerful large language models like OpenAI's GPT-4 [Ope23] and Google's PaLM [CND+22]. These large language models (LLMs) are firstly pre-trained by a large corpus of textual data, and then fine-tuned on a multi-task mixture of NLP tasks described using *instructions*. After multi-task instruction tuning, the models showcase a powerful ability of *zero-shot generalisation* on novel tasks, where tasks here can be effectively defined with just a handful of prompt examples. Finally, we may optionally train another reward function to align the language models more closely with human preferences. This technique, known as Reinforcement Learning with Human Feedback (RLHF) [CLB+17], facilitates the generated responses that better align with intricate human values — encourages the generation of more elaborate answers, while also empowering the model to discern and decline inappropriate queries or those that fall beyond its knowledge scope. The combination of these techniques has significantly contributed to the remarkable commercial success observed in models like OpenAI's ChatGPT [Ray23].

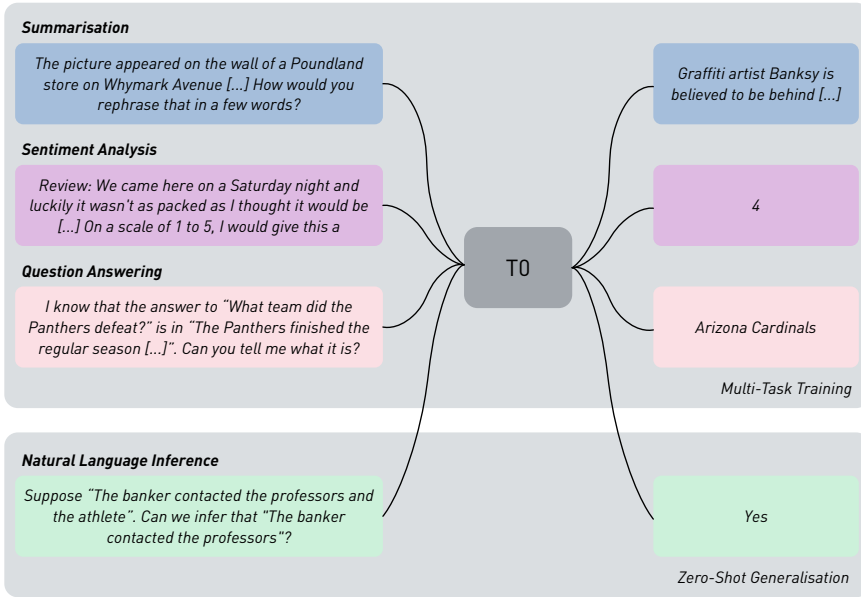


Figure 1.3. Multi-task training on large language models enables zero-shot generalisation. T0 [SWR+22] is an encoder-decoder language model that consumes textual inputs and produces target responses. It is trained on a multi-task mixture of NLP datasets partitioned into different tasks. Each dataset is associated with multiple prompt templates that are used to format example instances to input and target pairs. After training on a diverse mixture of tasks (top), the model demonstrated zero-shot generalisation capabilities to tasks that are not seen during training (bottom).

Tasks in Robotics

Tasks in robotics involve the *execution of actions to achieve specific goals*. These tasks include fundamental actions like grasping an object or navigating to a pre-defined location, and are progressively expanding into more intricate and complex ones, such as cooking elaborate meals or performing surgical procedures. Unlike CV and NLP, where tasks can be explicitly described through data representation or language instructions, defining robotic tasks can be more challenging due to the potential *ambiguity of actions* [ZRH+19]. The ambiguity may arise from the complexity of the physical world, the variability in how tasks can be executed, and the nuanced definitions required for successful completions.

Nowadays, the *high-level task representation* for robotics typically takes one of the following forms: *language instructions*: Like in NLP, robots can be given high-level language instructions. This approach enables humans to communicate tasks to robots using natural

language, by translating the instructions into actionable tasks. By jointly training the robot to understand a variety of instructions, it becomes more versatile in its abilities and can perform different tasks based on the provided instructions; *goal state / image*: Providing the target state or visual representation of the desired outcome can help guide the robot's actions. For example, showing a robot an image of a completed puzzle can guide it to place the pieces in the correct positions; *demonstrations*: Demonstrations involve showing the robot how to perform a task by physically guiding its actions. This provides *the most accurate* description of tasks, where the robot learns by mimicking human actions.

Once the high-level task representations are established, they need to be translated into *low-level action representations*, which describe the specific actions a robot should take. These representations can be *continuous* or *discrete*. Continuous representations involve specifying the robot's actions using a set of continuous variables, such as the robot's joint angles, velocities or torques. These provide fine-grained control and are suitable for tasks that require smooth, continuous motion. Discrete representations, on the other hand, involve specifying the robot's actions using a set of discrete waypoints, within the intermediate trajectories that are interpolated or generated with path-planning algorithms. These representations are beneficial for tasks that involve discrete actions, like picking and placing objects. It's worth noting that even when provided with the same high-level task description, the choice of action representation can significantly impact the robot's performance and its ability to generalise. Thus, the selection of the appropriate action representation is a critical consideration in the successful execution of robotic tasks.

Tasks with Multi-Modalities

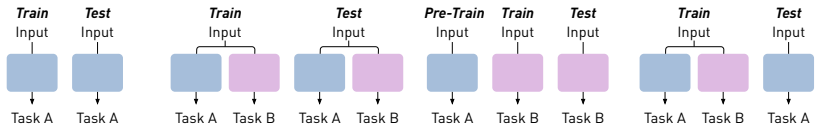
In addition to tasks within the same domain, tasks can also be extended across multiple domains, representing correlations and shared concepts across modalities, which we refer to as *multi-modal tasks*. Compared to single-modal tasks, multi-modal learning involves processing and integrating information from multiple sensory modalities or data sources such as text, images, audio, video, and more, in a *unified* manner. Multi-modal learning often intersects with multi-task learning, where diverse tasks necessitate the processing of data from different modalities. For example, multi-task robotic systems frequently involve object recognition (vision) and the comprehension of natural language (text and audio).

The concepts of multi-modal learning and multi-task learning can also mutually enrich one another. Multi-modal learning extends the multi-task learning process by providing a broader range of input data, while multi-task learning enhances multi-modal systems by facilitating the sharing of knowledge and features across various tasks.

In the following section, we will delve into commonly used learning strategies related to multiple tasks, aimed at enhancing the generalisation of both single- and multi-modal machine learning systems.

1.3 Learning Strategies with Multiple Tasks

Multi-task learning, transfer learning, and auxiliary learning [ZY18, ZQD+20] are commonly used learning strategies aimed at improving the generalisation of neural networks by leveraging *shared features from related tasks*. They involve training models on multiple tasks *simultaneously or sequentially* to enhance the learned representations. Throughout this thesis, we will extensively explore and address all three of these learning strategies. In this section, we briefly introduce these learning strategies and how they are related.



(a) Single Task Learning (b) Multi-Task Learning (c) Transfer Learning (d) Auxiliary Learning

Figure 1.4. A visual diagram of different learning strategies. *Single task learning* (a) involves training and evaluating a model on a single task. *Multi-task learning* (b) involves training and evaluating a model on multiple tasks. *Transfer learning* (c) involves training a model on a source task and then transferring and evaluating the learned knowledge to a target task. *Auxiliary learning* (d) involves training a model on multiple tasks, while only evaluating on the primary tasks.

Multi-Task Learning

In multi-task learning as illustrated in Fig. 1.4b, a single model is trained to perform multiple tasks simultaneously. This approach fosters the sharing and leveraging of information across tasks, resulting in enhanced generalisation across *all tasks*. The motivation behind multi-task learning lies in the recognition that the knowledge and representations acquired for one task can be utilised to improve the performance of other related tasks. Consequently, multi-task learning fosters a symbiotic relationship among these tasks, promoting *mutual benefits* and mitigating *task conflicts*.

Tasks that share similar objectives or features can benefit from collaborative learning, while tasks with conflicting objectives can be managed through careful regularisation, neural architecture designs and task weighting strategies.

Transfer Learning

Transfer learning as illustrated in Fig. 1.4c involves training a model on a source task and then *transferring the learned knowledge to a target task*. Unlike multi-task and auxiliary learning which only requires one-step training, transfer learning often involves two-step training: pre-training on a source task and fine-tuning on a target task. The idea is that the knowledge acquired during the initial training can serve as a foundation for the new task, thereby accelerating learning and improving generalisation. Transfer learning is particularly beneficial in domains where annotated (down-steam) datasets are scarce or costly to acquire. By leveraging pre-trained models trained on large-scale datasets, practitioners can bootstrap learning on new tasks with limited labelled data, saving time and resources.

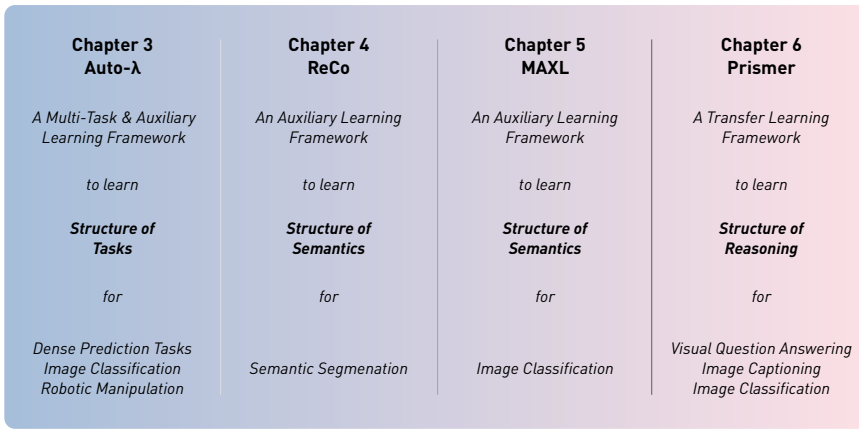
Auxiliary Learning

Auxiliary learning as illustrated in Fig. 1.4d, also known as auxiliary task learning or multi-task learning with auxiliary tasks, involves training a model on primary tasks (usually one), while simultaneously training it on one or more auxiliary tasks. These auxiliary tasks (occasionally necessitating input from other modalities) are designed to help the model learn more informative and robust features. It's important to note that both multi-task learning and auxiliary learning often adopt the same neural architecture design. However, unlike multi-task learning, where all tasks are equally important, auxiliary learning solely focuses on *improving the performance of the primary tasks*. The auxiliary tasks are included to act as *regularisers* that encourage the model to learn features that are useful for both the primary and auxiliary tasks. It's also interesting to note that, auxiliary tasks can serve as effective regularisers even when they are not directly related to the primary task. For instance, introducing noise through an additional output in a neural network can enhance generalisation by acting as a regularisation mechanism at the hidden layer, without implying any inherent relationship between the primary and auxiliary tasks [Car97].

Multi-task learning, transfer learning, and auxiliary learning collectively contribute to enhancing the robustness and efficiency of learning in machine learning tasks, and they can also be used in combination to yield even more benefits. A prevalent approach involves multi-task pre-training which initiates with generalised feature learning, followed by task-specific transfer learning [SWR+22, DCLT19]. Moreover, these learning strategies can also be seamlessly integrated with multi-modal learning by incorporating multi-modal data as inputs to the model. While these strategies deliver benefits such as efficient knowledge transfer and adaptability, they also pose challenges related to task interference, complexity, and imbalance, catalyse the focus of our research direction, as outlined in the next section.

1.4 Research Direction and Contribution

In this thesis, our primary focus is on the design of structured multi-task learning systems for visual perception. Our goal is to improve the generalisation of multi-task systems by learning *structured visual representations*. This pursuit has the potential to be not only beneficial for advancing computer vision tasks but also holds significant value for a wide range of vision-related multi-modal tasks — a robust and structured visual representation serves as a foundation for understanding scenes, objects, and their relationships, benefiting applications from classical scene understanding like object recognition and detection to more intricate undertakings like vision-based control and reasoning. The contributions of this thesis are guided by the following research questions, and are structured as follows:



Research Question 1: How can we understand the structure between tasks and leverage them to improve the generalisation and interpretability of neural networks?

We introduce the *Auto-*λ framework in Chapter 3 and in [LJDJ22]. Auto-λ is an optimisation framework that learns task relationships parameterised by task weightings, termed λ. Auto-λ learns continuous and dynamic multi-task relationships, allowing optimising on any choice of combination of tasks, and therefore effectively unifying multi-task learning and auxiliary learning into a single optimisation problem. Auto-λ has exhibited its effectiveness in improving generalisation across both computer vision and vision-based robotics tasks and has outperformed optimisation strategies that were specifically tailored for each of these tasks, showcasing its versatility and potential in diverse domains. Moreover, Auto-λ showcases the ability to learn task relationships that align with human intuition, thereby enhancing the interpretability of multi-task models.

Research Question 2: How can we understand the structure between semantic classes and leverage them to improve the generalisation and interpretability of neural networks?

We introduce the *ReCo* framework in Chapter 4 and in [LZJD21]. ReCo is an auxiliary learning framework that operates by conducting pixel-level contrastive learning on the learned representations within a segmentation model. It incorporates active sampling strategies guided by semantic class relationships, which helps in the adaptive selection of informative training samples. ReCo consistently yields improvements across a variety of scenarios, including both semi-supervised and supervised semantic segmentation methods. The most substantial impact is observed in the context of semi-supervised learning, particularly when dealing with very limited labelled data. With ReCo, we achieve high-quality semantic segmentation models requiring only five examples for each semantic class.

Research Question 3: How can we generate structured auxiliary tasks automatically to improve the generalisation of neural networks?

We introduce the *meta auxiliary learning* (MAXL) framework in Chapter 5 and in [LDJ19]. MAXL represents an auxiliary learning strategy that focuses on the direct generation of useful, structured auxiliary tasks such that any supervised learning task can be improved without requiring access to any further data. This approach involves the training of two neural networks: a label-generation network, responsible for predicting auxiliary labels; and a multi-task network that trains the primary tasks concurrently with the generated auxiliary tasks. This label-generation network's purpose is to produce structured auxiliary tasks that are inherently valuable, leading to improved generalisation of primary tasks by training them alongside the generated auxiliary tasks in a standard multi-task training setting. Remarkably, MAXL exhibits the capability to achieve performance improvements that are on par with, and in some cases, even surpass those achieved with auxiliary tasks designed by humans.

Research Question 4: How can we effectively transfer multi-task knowledge encoded within pre-trained expert models to improve open-ended vision-language reasoning?

We introduce the *Prismer* model in Chapter 6 and in [LFJ+24]. Prismer stands as a multi-modal model that harnesses and transfers the power of an ensemble of specialised pre-trained task experts. Prismer shows an effective way to scale-down multi-modal learning by breaking down a complex vision-language reasoning task into structured domain-specific reasoning. This decomposition leads to improved training efficiency, as the model can focus on integrating specialised skills and domain-specific knowledge within each expert,

rather than attempting to master all aspects simultaneously within a single model. As a result, Prismer achieves fine-tuned and few-shot vision-language reasoning performance which is competitive with current state-of-the-arts, while requiring up to two orders of magnitude less training data.

A substantial portion of the research presented in this thesis is featured in the following publications:

1. **Shikun Liu**, Andrew J. Davison, and Edward Johns (2019). "Self-Supervised Generalisation with Meta Auxiliary Learning." In *Advances in Neural Information Processing Systems (NeurIPS)*.

2. **Shikun Liu**, Shuaifeng Zhi, Edward Johns, and Andrew J. Davison (2022). "Bootstrapping Semantic Segmentation with Regional Contrast." In *International Conference on Learning Representations (ICLR)*.

3. **Shikun Liu**, Stephen James, Andrew J. Davison, and Edward Johns (2022). "Auto-Lambda: Disentangling Dynamic Task Relationships." In *Transactions on Machine Learning Research (TMLR)*.

4. **Shikun Liu**, Linxi Fan, Edward Johns, Zhiding Yu, Chaowei Xiao, and Anima Anandkumar (2024). "Prismer: A Vision-Language Model with Multi-Task Experts." In *Transactions on Machine Learning Research (TMLR)*.

Additionally, the author has made contributions to the following publications:

1. Edgar Sucar, **Shikun Liu**, Joseph Ortiz, and Andrew J. Davison (2021). "iMAP: Implicit Mapping and Positioning in Real-Time." In *International Conference on Computer Vision (ICCV)*.

2. Xin Kong, **Shikun Liu**, Marwan Taher, and Andrew J. Davison (2023). "vMAP: Vectorised Object Mapping for Neural Field SLAM." In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

3. Xin Kong, **Shikun Liu**, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J. Davison (2024). "EscherNet: A Generative Model for Scalable View Synthesis." In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

In addition to the main contributions highlighted above, this thesis includes a preliminary section, which serves as a technical foundation, establishing the fundamental concepts and necessary background knowledge for the content presented throughout the thesis (Chapter 2). After presenting our main findings, we conclude the thesis and outline intriguing avenues for future research (Chapter 7).

2

Preliminaries

In this chapter, we provide an extensive exploration of multi-task learning techniques in computer vision tasks. We start with the introduction of modern neural architecture designs (Section 2.1), and then proceed to introduce and categorise multi-task architecture designs (Section 2.2). We survey various MTL optimisation strategies aimed at mitigating the issue of task conflict and balancing task influences (Section 2.3). Finally, we delve into how multi-task learning can enhance the interpretability of neural network learning processes and uncover the relationships between tasks (Section 2.4).

Notations

In multi-task learning, we adopt a uniform notation scheme as follows. We denote a muli-task network to be $f(\cdot;\boldsymbol{\theta}) : \mathcal{X} \mapsto \mathcal{Y}$, where $\boldsymbol{\theta}$ denotes the network parameters. These parameters are trained to map inputs $X_{1:K} \in \mathcal{X}$ to their corresponding labels $Y_{1:K} \in \mathcal{Y}$ across a total of K learning tasks. The multi-task network comprises a combination of task-shared parameters denoted as θ_{sh} and K task-specific parameters denoted as $\theta_{1:K}$, collectively represented as $\boldsymbol{\theta} = \{\theta_{sh}, \theta_{1:K}\}$. Each task is allocated a task-specific weight denoted as $\boldsymbol{\lambda} = \{\lambda_{1:K}\}$. We express the task spaces using a collection of task-specific inputs and outputs pairs, denoted as $\boldsymbol{T} = \{T_{1:K}\}$, with each $T_i = (X_i, Y_i)$.

The design of the task spaces can be further divided into two different settings: a single-domain setting (where all inputs are the same $X_i = X_j, i \neq j$, *i.e.*, one-to-many mapping),

and a multi-domain setting (where all inputs are different: $X_i \neq X_j, i \neq j$, *i.e.,* many-to-many mapping). We want to optimise $\boldsymbol{\theta}$ for all tasks $\boldsymbol{T}$ and obtain a good performance in some pre-selected primary tasks $\boldsymbol{T}^{pri} \subseteq \boldsymbol{T}$. If we choose $\boldsymbol{T}^{pri} = \boldsymbol{T}$, we are in the multi-task learning setting, otherwise, we are in the auxiliary learning setting.

2.1 Advances in Neural Architecture Design

The field of computer vision has witnessed remarkable progress over the past few decades, driven by the development of increasingly sophisticated neural network architectures. These advancements have reshaped the way we perceive, interpret, and analyse visual information. Among the many architectural innovations that have fuelled this evolution, multi-layer perceptrons, convolutional neural networks, and transformers stand out as transformative milestones. In this section, we will briefly explore how these architectural designs have evolved, and their unique properties and limitations in learning visual representations.

Multi-Layer Perceptrons: The Pioneers of Deep Learning

The journey through the evolution of neural architecture in computer vision begins with Multi-Layer Perceptrons (MLPs), or fully-connected feed-forward neural networks. MLPs are the fundamental building blocks of deep learning and represent a significant departure from shallow neural networks. Unlike other supervised learning methods, such as logistic regression and support vector machines [HDO+98], which were limited by their capacity to model simple linear relationships, MLPs are characterised by multiple hidden layers, allowing them to capture intricate, non-linear patterns within data.

MLPs are capable of approximating any continuous function, known as *universal function approximators* [HSW89]. They are consisted of inter-connected layers of artificial neurons, and these neurons process information by applying weighted sums, followed by activation functions. Each layer refines the features learned in the previous layer, ultimately culminating in a final layer that produces the network's output. The depth of an MLP, achieved by stacking multiple layers, enables it to learn increasingly abstract and complex representations of data.

However, MLPs do not naturally handle the *spatial structure* of images. They treat each pixel as an independent feature, which ignores the important *local relationships* between neighbouring pixels. Moreover, they tend to struggle with high-resolution images as they may require a large number of neurons in the input layer to process all the pixels, which can lead to a large computational cost.

Convolutional Neural Networks: Spatial Hierarchies and Translation Invariance

The emergence of Convolutional Neural Networks (CNNs) marked a significant turning point in computer vision. CNNs were purposefully designed to address the challenges that MLPs encountered in computer vision tasks.

CNNs were first proposed in [LBBH98] which leverage a unique architecture that incorporates convolutional layers, pooling layers, and fully connected layers, and later popularised by AlexNet [KSH12] and VGGNet [SZ15] which achieved remarkable success in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [DDS+09].

Convolutional layers apply learnable filters to input data, enabling the network to automatically learn spatial hierarchies of features. This characteristic is especially beneficial in vision tasks, where detecting local patterns and structures is essential. Pooling layers, on the other hand, downsample the data to reduce computational complexity while retaining important features. The use of convolutional and pooling layers in CNNs equips them with translation invariance, making them particularly adept at handling images.

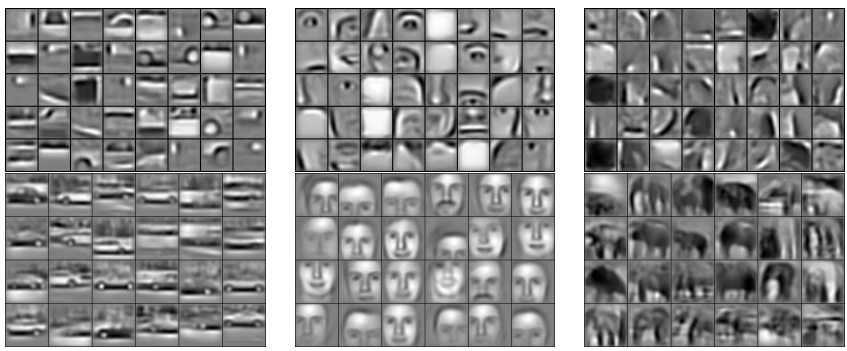


Figure 2.1. Visualisation of the feature maps in convolutional deep belief networks [LGRN09]. We observe the emergence of basic features, such as edges and textures in earlier layers (top), and feature maps become more complex and begin to form recognisable object parts in the later layers (bottom).

In addition to their capacity for spatial hierarchies and translation invariance, CNNs employ techniques like weight sharing and local receptive fields to significantly reduce the number of parameters in comparison to MLPs. This reduction in parameters enables the model's ability to generalise from large-scale data, making CNNs easier to scale and highly efficient for computer vision tasks.

CNNs have revolutionised the field of computer vision, leading to significant breakthroughs in areas such as image classification, object detection, and medical image analysis. As computer vision tasks have become increasingly diverse and complex, the evolution of CNN architectures has played a pivotal role in addressing these challenges.

Traditional convolutional layers rely on fixed grid structures, which may not be the most effective approach for all computer vision tasks. For instance, in tasks like semantic segmentation or object detection, the fixed grid structure struggles to capture long-range spatial relationships between pixels. To tackle this issue, two notable techniques were introduced. Dilated convolution, also known as atrous convolution [CPK+17], popularised by DeepLab architectures [CPK+17], enhances the receptive field of convolution by introducing gaps between kernel elements. These gaps allow CNNs to capture long-range dependencies, making it particularly valuable for tasks requiring an extensive spatial context. Deformable convolution takes flexibility in the receptive field to a new level. It allows the network to dynamically adjust the shape and position of the receptive field for each spatial location. This flexibility enables CNNs to learn complex spatial relationships and deformations in the data, making it highly effective in tasks that demand precise localisation.

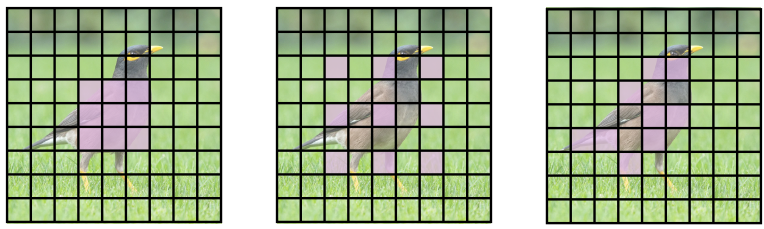


Figure 2.2. Visualisation of different $[3 \times 3]$ **convolutional kernel designs.** Standard convolution employs a fixed grid structure confined to neighbouring pixels, while dilated and deformable convolution introduces gaps and flexibility to the grid structure to capture long-range relationships.

In addition to innovations in convolutional layers, advancements in CNN architecture designs have been instrumental in enhancing the performance and capabilities of computer vision systems. ResNet [HZRS16] introduced residual connections, facilitating the training of deeper neural networks. This architecture mitigates the *vanishing gradient problem* and enables the development of extremely deep networks. DenseNet [HLVDMW17] introduced dense connections between layers, enhancing information flow and gradient propagation. UNet [RFB15] is an encoder-decoder architecture tailored for image segmentation tasks. Its U-shaped structure combines a contracting path for feature extraction and an expans-

ive path for precise localisation, making it a popular choice for semantic segmentation. SENet [HSS18] focuses on feature recalibration by introducing a "squeeze-and-excitation" mechanism. It adaptively recalibrates the importance of different channels within each feature map, enhancing the model's capability to focus on relevant features.

These architectural innovations represent the continuous evolution of computer vision, unlocking new possibilities and pushing the boundaries of what machines can achieve in terms of visual perception and understanding. As computer vision tasks continue to grow in complexity and diversity, these advancements play a critical role in enabling AI systems to excel in various domains.

Transformers: Representing Images as Sequences

The rise of transformers, originally introduced in the seminal work [VSP+17], represents a paradigm shift in neural architecture design. Originally developed for natural language processing, transformers have been adapted to computer vision, known as Vision Transformers (ViTs) [DBK+20]. ViTs represent a fresh and compelling alternative to the long-standing dominance of CNNs.

In the ViT framework, images are transformed into sequences of smaller, fixed-size patches, a departure from the traditional pixel-based approach applied in CNNs. These patches serve as the fundamental units of analysis in the ViT model, enabling it to capture both local and global information efficiently. The heart of the ViT architecture lies in its multi-head self-attention mechanism, a critical component that empowers the model to make sense of the patch-based image representations. This self-attention mechanism operates through a process known as scaled dot-product attention. In this process, each patch within the image sequence interacts with every other patches. This interaction is achieved by computing the dot product between a query, often representing one patch, and a set of keys, representing all other patches. The resulting dot products are then scaled to prevent the gradients from becoming too large or too small during training, ensuring stable and effective learning.

The scaled dot-product attention essentially evaluates the similarity between different patches, which allows the ViT model to discern complex relationships and dependencies within the image. By learning how each patch relates to every other patches, the model can grasp both global context and fine-grained details, transcending the limitations of local analysis typically associated with CNNs. Consequently, ViTs have achieved competitive results in image classification, object detection, and semantic segmentation, challenging the traditional dominance of CNNs in these domains.

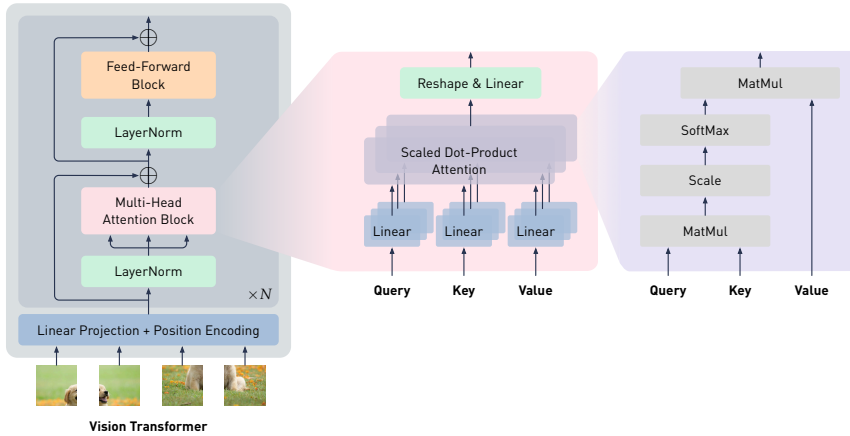


Figure 2.3. A visual diagram of the Vision Transformer architecture. ViT represents images as sequences of patches, which are then processed through a multi-head self-attention mechanism which performs scaled dot-product attention.

While ViTs emerge as a compelling alternative to CNNs, it's essential to recognise that these two paradigms need not be mutually exclusive. In fact, they can be combined to harness the strengths of both and achieve even more impressive results in computer vision tasks. Notable examples include CvT [WXC+21], CoatNet [DLLT21], and CeiT[YGL+21]. These models adopt a hybrid architecture that combines elements from both CNNs and transformers. This integration introduces the desirable properties of CNNs, such as shift, scale, and distortion invariance, into the transformer architecture, while preserving the merits of transformers, such as dynamic attention, variable token length, and global context, which eventually translates into models that are both powerful and resource-efficient, which is a crucial consideration in modern machine learning applications.

Interestingly, recent research [THK+21, SBBD23] highlights that the most important factors determining the performance of a sensibly designed model are the compute and data available for training rather than the architectural designs. This observation challenges conventional wisdom and highlights the critical role of resources in achieving optimal model performance. However, it's important to recognise that architectural choices still play a crucial role in certain contexts, particularly in scenarios that demand adaptability across different modalities. This adaptability is particularly valuable in learning multi-modal tasks, such as image captioning, visual question answering, text-to-image generation, and audio-visual processing, where different types of data, such as images, text, and audio, need to be processed in a cohesive manner.

2.2 Multi-Task Neural Architectures

Multi-task neural architecture design is one of the most important directions in MTL research and typically employs two main paradigms to learn shared representations: *hard parameter sharing* and *soft parameter sharing*, incorporating explicit or implicit feature-sharing strategies for modelling cross-task interaction.

In this section, we will primarily focus on multi-task architectures that are constructed using CNNs, as they have established themselves as the predominant architectural choice for addressing multi-task learning in computer vision tasks.

Hard Parameter Sharing

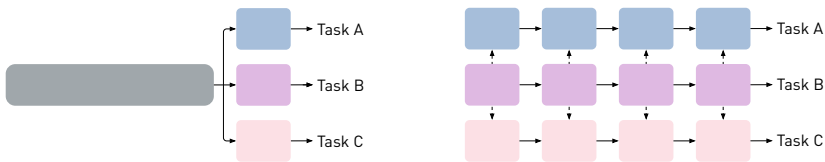
In hard parameter sharing, multiple tasks share the same neural network architecture, and initial hidden representations are identical across tasks. These shared representations then branch into independent task-specific representations at a later stage. The most straightforward implementation, illustrated in Fig. 2.4a, is a neural network with a shared backbone and task-specific heads at the final hidden layer that is responsible for making task-specific predictions [KGC18, SK18, CBLR18, GHH+18]. Despite its simplicity and lower computational requirements, this vanilla hard parameter-sharing approach might not perform optimally when tasks have significantly different data distributions or require varying levels of complexity.

The inherent limitations of the straightforward branching approach in hard parameter sharing have spurred interest in designing multi-task neural architectures that facilitate finely branched structures. In Multi-Task Network Cascades [DHS16], the output of each task-specific branch is appended to the input of the subsequent task-specific branch, creating a "cascade" of information flow to learn *causal task relationships*. UberNet [Kok17] presents another example of a hard-parameter sharing model that allows for branching at multiple layers. This model adopts a multi-scale approach, where at each scale, the network branches at multiple layers, with each branch dedicated to a specific task.

However, in these models, the branching points within the network are typically predetermined based on prior task knowledge, which can still result in sub-optimal task groupings. To achieve even more optimal MTL architecture design, recent works like [VGDBVG20, GLU20], as illustrated in Fig. 2.5a, have proposed efficient design methodologies that automatically determine *where to share or branch* within the network.

Soft Parameter Sharing

In contrast to hard parameter sharing, where task-specific parameters $\theta_{1:K}$ are completely isolated, soft parameter sharing allows the parameters of different tasks to interact with each other. The vanilla soft parameter sharing approach is designed to achieve such interaction *implicitly* by imposing regularisation or loss components that penalise the differences between the parameters of the task-specific models, as illustrated in Fig.2.4b. This regularisation term can be designed as a simple L_2-norm [DCBC15] or the trace norm [YH17].



(a) Vanilla Hard Parameter Sharing (b) Vanilla Soft Parameter Sharing

Figure 2.4. A visual diagram of vanilla hard parameter sharing and soft parameter sharing design in multi-task learning. *Vanilla hard parameter sharing* (a) relies on a single shared neural network architecture for all tasks, with each task has its separate set of output layers that are task-specific. *Vanilla soft parameter sharing* (b) involves separate networks for each task with regularisation techniques to encourage parameter similarity.

Alternatively, *explicit* interaction can be introduced through feature fusion modules. Examples include the use of cross-stitch units [MSGH16], which apply linear combinations of activations, and NDDR modules [GMZ+19], which employ $[1 \times 1]$ convolutions. These modules are integrated into each layer of task-specific networks to facilitate soft feature fusion, as illustrated in Fig. 2.5b. MTAN [LJD19], on the other hand, achieves feature fusion through a learnable soft attention mask applied over a global feature pool θ_{sh} to construct task-specific networks, as illustrated in Fig. 2.5c.

These aforementioned methods are designed to make predictions for all tasks directly from the same input in a single-step process. This design, focusing on feature fusion within the encoder part of the network, is also commonly referred to as an *encoder-focused* architecture, following the categorisation introduced in the survey [VGVG+21]. To promote knowledge sharing directly in the task space (*e.g.* depth discontinuities are usually aligned with semantic edges), some recent works employ multi-task networks to make initial task predictions and then leverage features from these initial predictions to further refine each task's output in a recursive manner [BV16, XOWS18, VGVG20]. As these MTL approaches exchange information during the decoding stage to facilitate fine-grained task refinement, they are referred to as *decoder-focused* architectures.

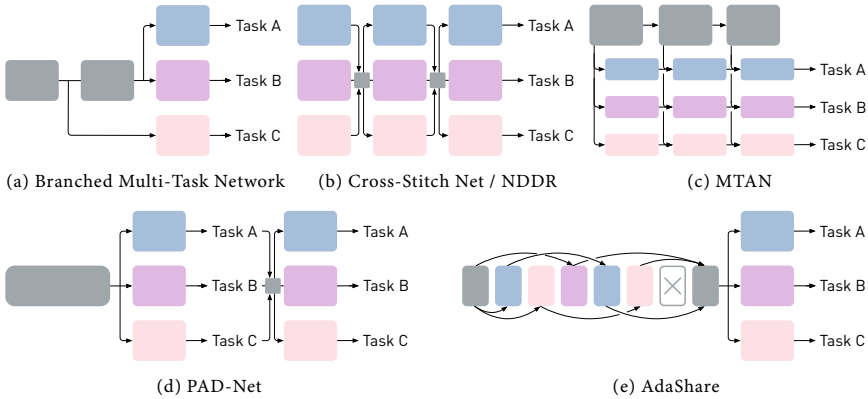


Figure 2.5. A visual diagram of modern multi-task neural architecture design. Modern multi-task neural architectures employ more advanced feature fusion strategies to promote cross-task interaction. *Branched Multi-Task Network* (a) automatically determines where to share or branch within the network. *Cross-Stitch Net / NDDR* (b) and *MTAN* (c) explicitly introduce feature fusion modules to facilitate soft feature fusion. *PAD-Net* (d) is a decoder-focused architecture that leverages initial task predictions to further refine each task's output in a recursive manner. *AdaShare* (e) is a NAS-based method with a bounded network parameter count that automatically determines which layers to share across which tasks.

For instance, PAD-Net [XOWS18], depicted in Fig. 2.5d, is one of the pioneering decoder-focused architectures. In this network, an input image first goes through an off-the-shelf backbone network with task-specific heads to produce initial predictions for each task, as in the vanilla hard parameter-sharing approach. These initial task predictions are recombined to extract cross-task information, which is then fed into another set of task-specific heads to generate the final task predictions. Similar techniques used in advanced hard parameter sharing approaches can also be applied in decoder-focused architectures to further improve performance, promoting feature fusion with a multi-scale design [VGVG20].

Decoder-focused architectures have demonstrated significantly improved performance compared to encoder-focused architectures and have achieved state-of-the-art performance when further combined with more sophisticated design components within the transformer architectures [YX22b, YX22a].

Neural Architecture Search

While both hard and soft parameter sharing approaches have exhibited promising results in multi-task learning benchmarks, they are somewhat constrained by the manual design

of the network architecture[1] and can become computationally inefficient — the model size scales proportionally with the number of tasks.

To address these constraints, recent research has delved into the utilisation of neural architecture search (NAS) techniques. These NAS methods aim to automatically determine *which layers to share across which tasks* in multi-task learning. For instance, AdaShare [SPFS20], as depicted in Fig. 2.5e, employs a differentiable NAS framework to jointly learn a feature-sharing policy and network weights. This approach adaptively selects network layers to execute for a given task. Consequently, the network's parameter count remains *bounded* by the original backbone network, and thereby the number of parameters becomes *independent* of the number of tasks. The exploration of what parameters to share across tasks is further advanced in AutoMTL [ZLG22], which introduces a multi-task super-model compiler to automatically transform backbone network layers into a multi-task structure encoded with a pre-defined search space, affording greater flexibility in terms of choosing the backbone model. These NAS-driven approaches offer a promising avenue for enhancing the efficiency and adaptability in designing more advanced multi-task neural architectures.

2.3 Multi-Task Optimisation Strategies

In addition to neural architecture design, balancing the joint learning of multiple tasks is another important research challenge in MTL, as we need to ensure that no single task dominates the influence on the network weights. To address this issue, various optimisation strategies have been proposed, which can be categorised into two main directions.

Single Objective Optimisation

In this approach, the goal is to minimise a linearly combined single-valued loss for all tasks, with each task i being assigned a task-specific weighting λ_i:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{K} \lambda_i \cdot L_i \left(f \left(x_i; \theta_{sh}, \theta_i \right), y_i \right). \tag{2.1}$$

To balance the influence of each task on the shared network parameters θ_{sh}, suitable task weightings $\boldsymbol{\lambda}$ can be determined. These weightings can be manually chosen based on the prior task knowledge, set to be equal $\lambda_{1:K} = 1/K$ as a common practice, or learned adaptively. Adaptive weightings, also known as *weighting*-based methods, learn to prioritise more

[1] Branched multi-task networks [VGDBVG20, GLU20] introduced as a hard parameter sharing approach can also be considered as a form of NAS-based methods.

difficult tasks, as measured by heuristics such as task uncertainty (uncertainty weighting) [KGC18], task learning speed (dynamic weight average, DWA) [LJD19], or relative task loss value (dynamic task prioritisation, DTP) [GHH+18].

Gradient-based methods, on the other hand, aim to find an aggregated gradient by linearly combining individual task gradients under various constraints. These methods offer a more direct and nuanced means of balancing tasks compared to weighting-based methods, as they operate on task-specific gradients, enabling fine-grained adjustments to shared network parameters by generating suitable gradient weights $\lambda^g_{1:K}$[2],

$$\theta_{sh} = \theta_{sh} - \eta \sum_{i=1}^{K} \lambda_i^g \cdot \nabla_{\theta_{sh}} L_i \left(f\left(x_i; \theta_{sh}, \theta_i\right), y_i \right). \tag{2.2}$$

These constraints can include equal gradient magnitude (GradNorm) [CBLR18], equal gradient projection (IMTL) [LLK+21], conflicting gradient dropout (GradDrop) [CNH+20], or aligning the principal components of the gradient matrix (Aligned-MTL) [SPKK23].

Multi-Objective Optimisation

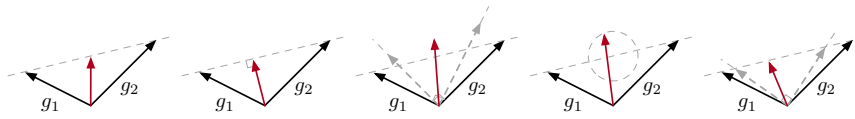
Multi-task learning can also be formulated as a multi-objective optimisation problem, where the objective is to minimise a vector-valued loss,

$$\min_{\boldsymbol{\theta}} \left[L_i \left(f\left(x_i; \theta_{sh}, \theta_i\right), y_i \right)_{i=1:K} \right]^\top. \tag{2.3}$$

This formulation proves to be more attractive because, by recasting multi-task learning as a multi-objective optimisation problem, the process becomes more tractable to optimisation techniques — their convergence to a Pareto optimal solution is guaranteed under mild conditions. In contrast, seeking a global optimum in the context of a single-objective optimisation problem often proves to be very challenging and computationally demanding. A Pareto optimal solution implies that, for the given set of tasks, no further improvements can be made in the loss for one task without sacrificing the loss in at least one of the other tasks. In other words, it's an equilibrium where we've reached the best trade-off between competing objectives.

A crucial characteristic of multi-objective optimisation is that, since there's no natural linear ordering on vectors, it's not always possible to compare solutions or determine a

[2] The gradient weights $\lambda^g_{1:K}$ will only affect the task-shared parameters θ_{sh} but not task-specific parameters $\theta_{1:K}$, each of which is updated by the i-th task gradient $\nabla_{\theta_i} L_i \left(f\left(x_i; \theta_{sh}, \theta_i\right), y_i \right)$. As such, gradient-based methods would typically require a longer training time compared to weighting-based methods.



(a) Equal (b) MGDA (c) PCGrad (d) CAGrad (e) Aligned MTL

Figure 2.6. Geometric visualisation of diverse multi-task optimisation methods. We show the updated gradient direction (coloured red) obtained by various multi-task optimisation methods in a two-dimensional parameter space for a simple two-task learning problem, labelled as g_1 and g_2. Each method has its own way of adjusting the gradients to effectively balance the parameter space.

clear optimal value. Consequently, selecting between different Pareto optimal solutions can be challenging without additional assumptions or prior task preferences [NSFC20].

This challenge has been observed in optimisation methods like MGDA (Multi-Gradient Descent Algorithm) [SK18] and PCGrad [YKG+20], which are designed to converge to an arbitrary point on the Pareto set depending on the network initialisation, without explicit control over the specific point of convergence. To address this issue, [LZL+19] extends MGDA to generate a set of well-representative Pareto solutions from which a preferred solution can be selected. CAGrad [LLJ+21] explores an update direction in a neighbourhood of the average gradient that maximises the worst improvement of any task, using gradient conflict to regularise the optimisation trajectory. Nash-MTL [NSA+22] proposes a Nash bargaining solution to find a Pareto optimal solution that is also proportionally fair, ensuring that any alternative solution would have a negative average relative change.[3]

These methods aim to facilitate the selection of a single Pareto optimal solution from the set of possible solutions, enhancing the practicality of multi-objective optimisation in multi-task learning.

Meta Learning

Meta-learning, often referred to as "learning to learn," represents the process of improving a learning algorithm over multiple learning episodes, rather than just optimising model predictions over multiple data instances, as in traditional machine learning [VD02, HAMS20]. The process of meta-learning involves two key components: base learning and meta-learning. During base learning, an inner or lower learning algorithm solves a task defined

[3] While operating under distinct optimisation objectives, it's worth noting that multi-objective optimisation methods can be viewed as gradient-based methods in the sense that they are designed to iteratively adjust the direction of task-shared parameters with the same formulation of Eq. 2.2, with the aim of converging toward a Pareto optimal solution.

by a specific dataset and objective function. During meta-learning, an outer or upper algorithm updates the inner learning algorithm such that the learned model improves an outer objective, which typically involves optimising the model's ability to adapt to new tasks quickly and effectively.

The essence of meta-learning lies in its ability to address *unknown future tasks*, a goal that differs slightly from traditional multi-task learning, which focuses on solving *a pre-determined set of known tasks*. However, the principles of meta-learning can be effectively integrated into multi-task learning frameworks to enhance performance and adaptability. For instance, meta-learning can be used to learn a shared network initialisation that can be adapted to new tasks with few gradient steps and minimal data [FAL17, NS18], to capture the relatedness between tasks via hyper-gradients [FDFP17], or to adaptively prioritise among multiple tasks [LBKH19]. Meta learning can also be directly employed as an optimisation strategy to solve multi-task learning problems, such as within a multi-objective formulation [YLY+21] or to balance the worst-performing tasks via a min-max game [MRY21].

2.4 Interpretability and Task Relationships

Multi-task learning can improve model accuracy, memory efficiency, and inference speed, when compared to training tasks individually. However, it often requires careful selection of *which tasks should be trained together*, to avoid *negative transfer*, where irrelevant tasks produce conflicting gradients and complicate the optimisation landscape. As such, without prior knowledge of the underlying relationships between the tasks, and hence which tasks should be trained together, multi-task learning can sometimes have worse prediction performance than single-task learning.

While various multi-task optimisation strategies introduced earlier have been designed to address this issue, training non-related or conflicting tasks together can still lead to sub-optimal model performance. Moreover, a model that is aware of the relationships among tasks can require less supervision, consume fewer computational resources, and provide insights into the structure of learning in a more interpretable manner.

Inferring Task Relationships by Task Grouping

The concept of the *relationship* among tasks is a *relative* metric that can be defined as *how much a group of tasks can benefit from the representations learned by another group of tasks*. The task *affinity* score is an *absolute* metric used to quantify the relationship between these tasks, which can be defined differently depending on the learning strategies introduced in

Section 1.3. The term *order* refers to the number of tasks for which we want to compute this relationship. It's important to note that different learning strategies can infer different task relationships [SZC+20a].

- In transfer learning, we say that task A is more related to task B than task C, if the performance of task A is higher when training task A with the representations learned from task B, compared to when training task A with the representations learned from task C. The task affinity score between two tasks is determined by the average performance improvements observed during both forward and backward transfer, relative to when the tasks are trained individually.

- In multi-task and auxiliary learning, we say that task A is more related to task B than task C, if the performance of task A is higher when training tasks A and B together, compared to when training tasks A and C together. The task affinity score between two tasks is determined based on the average performance improvements observed when training the two tasks as a pair, relative to when the tasks are trained individually.

	Depth	Normal	Keypoint	Edge
Sem. Seg.	-0.62%	-1.39%	+0.25%	-15.78%
Depth		-0.54%	+2.43%	+1.42%
Normal			+0.67%	+3.95%
Keypoint				-1.95%

(a) Multi-Task Learning

	Depth	Normal	Keypoint	Edge
Sem. Seg.	+1.74%	+1.83%	+0.72%	+0.70%
Depth		+1.92%	+0.41%	+0.47%
Normal			+0.09%	+0.12%
Keypoint				+0.23%

(b) Transfer Learning

Table 2.1. Pairwise task affinity discovered in multi-task learning and transfer learning strategies. We can observe that there appears to be no correlation between pairwise task affinities in multi-task learning and transfer learning in Taskonomy dataset [ZSS+18], as observed in the results obtained from [ZSS+18] in the transfer learning setting and [SZC+20a] in the multi-task learning setting.

In this thesis, we mainly focus on the task relationships defined in the multi-task learning setting, to determine which tasks should be trained together. One straightforward but computationally expensive approach would be to exhaustively search over all possible task groupings, where tasks within a group are equally weighted, and all other tasks are ignored.[4] However, this requires training $2^{|\mathcal{T}|} - 1$ multi-task networks for a set of tasks $\mathcal{T}$, and the computational cost for this search can be intractable when $|\mathcal{T}|$ is large. To address this challenge, prior works have developed efficient task grouping frameworks based on heuristics to speed up training, such as using an early stopping approximation [SZC+20a] and com-

[4] Task grouping can be considered as a special form of weighting-based methods, by finding fixed and binary task weightings indicating which tasks should be trained together.

puting a lookahead loss averaged across a few training steps [FAZ+21]. Nevertheless, these task-grouping techniques are subject to two notable limitations. First, they are typically two-stage methods, requiring an initial search for the optimal task structure and subsequent re-training of the multi-task network with the identified structure. Second, these methods are primarily designed for lower-order task relationships, making it challenging to directly obtain higher-order task relationships for three or more tasks. In practice, higher-order relationships are approximated using combinations of lower-order relationships. But as the number of training tasks increases, even evaluating these combinations can also become prohibitively computationally expensive.

Inferring Task Relationships by Neural Architecture Design

Task relationships can also be implicitly inferred from the feature-sharing strategy employed in the design of multi-task neural architectures. For instance, in hard parameter-sharing approaches, the branching points in the network can be seen as a form of task grouping, indicating that tasks within the same branch are considered to have closer relationships. In soft parameter sharing approaches, the feature fusion modules can be considered as a way of task grouping, as tasks that share the same feature fusion module are considered to have closer relationships. Therefore, the feature-sharing strategy, which automatically determines where to share or branch within the network, can also be regarded as a means of inferring the underlying task structures learned by a neural network. This strategy implicitly captures the network's understanding of how tasks are related to each other.

	Sem. Seg.	Normal	Depth	Keypoint	Edge
Sem. Seg.	1.00	0.61	0.43	0.58	0.00
Normal		1.00	0.90	0.56	0.44
Depth			1.00	0.49	0.47
Keypoint				1.00	0.67
Edge					1.00

(a) AdaShare

	Sem. Seg.	Normal	Depth	Keypoint	Edge
Sem. Seg.	1.00	0.65	0.26	0.62	0.69
Normal		1.00	0.69	0.64	0.56
Depth			1.00	0.04	0.00
Keypoint				1.00	0.89
Edge					1.00

(b) AutoMTL

Table 2.2. Pairwise task affinity discovered in AdaShare and AutoMTL methods. We can observe that both methods obtained similar task relationships, despite being optimised with different backbone architecture and search spaces. This suggests that the task relationships they've identified are robust and not highly dependent on specific model architectures or search strategies.

In Table 2.2, we present the pairwise task affinity scores derived from the NAS methods AdaShare [SPFS20] and AutoMTL [ZLG22], as discussed in Section 2.2. The pairwise task affinity score here is defined as the normalised cosine similarity between the final task-specific policy logits discovered by the NAS methods. Interestingly, it is evident that

both methods, despite being optimised with different backbone architectures and search spaces, have yielded similar task relationships. For instance, we can observe that in both methods, 2D tasks such as keypoints prediction and edge detection are highly related, and 3D tasks like surface normal prediction and depth estimation are also highly related.

However, it's important to note that the task affinity scores derived from the feature-sharing strategies provide insights into the task structure learned by the multi-task neural network during architecture search, they might not *directly reflect optimal task performance*. Task grouping methods, which explore various combinations of tasks to identify global optimal structures, offer a more direct approach to uncovering the true task relationships and thereby can obtain optimal performance based on any selection of primary tasks.

3

Exploring Task Relationships with Automated Weightings

Understanding the structure of multiple tasks allows for multi-task learning to improve the generalisation ability of one or all of them. However, it usually requires training each pairwise combination of tasks together in order to capture accurate task relationships, at an extremely high computational cost, as discussed in Section 2.4.

In this chapter, we introduce a novel weighting framework called Auto-λ to automate the discovery of multi-task relationships. Unlike previous methods that assume fixed task relationships, Auto-λ dynamically explores continuous task relationships via task-specific weightings. It has the flexibility to optimise any combination of tasks, making it a versatile tool for various multi-task and auxiliary learning problems in computer vision and robotics. The results show that Auto-λ achieves state-of-the-art performance, even when compared to optimisation strategies tailored for specific problems and data domains. Finally, we observe that Auto-λ can reveal intriguing learning behaviours, providing new insights in understanding multi-task relationships.

3.1 Rethinking Multi-Task Relationships: Static or Dynamic?

Imagine a determined student who wants to learn advanced mathematics topics like calculus and linear algebra. However, before diving into these complex subjects, it is essential

to establish a strong foundation in basic arithmetic and algebra. Let's consider learning advanced and fundamental mathematics as two individual tasks, we argue that a better learning strategy is to craft a curriculum that dynamically tailors itself to the learner's current knowledge and skill level. Instead of rigidly following a fixed curriculum, the learning system adjusts the emphasis on different mathematical concepts based on the learner's progress. This approach ensures a solid foundation before moving on to more advanced topics, improving the overall learning experience and comprehension of mathematics.

Building upon this design insight, we present a novel perspective on the relationships among tasks: it's *dynamic, continually evolving* based on the *current state* of the multi-task network during training, as a form of *automated curriculum learning* [BLCW09]. We propose that task relationships can be deduced within a single optimisation problem that iteratively operates throughout training, automatically refining the significance of each task in line with our optimisation goals. In this way, we aspire to merge multi-task learning and auxiliary learning into a *singular, unified* framework.

3.2 Related Work

Multi-task Network Design Multi-Task Learning (MTL) aims at simultaneously solving multiple learning problems while sharing information across tasks. The techniques used in multi-task architecture design can be categorised into hard-parameter sharing [Kok17, HMBS21], soft-parameter sharing [MSGH16, LJD19, MRK19], and neural architecture search [GBJ+20, SPFS20, RKR18]. Please refer to Sec. 2.2 for a detailed review.

Multi-task and Auxiliary-task Optimisation In an orthogonal direction to advance architecture design, significant efforts have been invested to improve multi-task optimisation strategies. Although this is a multi-objective optimisation problem [SK18, LZL+19, YLY+21], a single surrogate loss consisting of linear combination of task losses are more commonly studied in practice. Notable works have investigated finding suitable task weightings based on different criteria, such as task uncertainty [KGC18], task prioritisation [GHH+18] and task loss magnitudes [LJD19]. Other works have focused on directly modify task gradients [CBLR18, CNH+20, YKG+20, JV22, LLJ+21, NSA+22]. Please refer to Sec. 2.3 for a detailed review.

Similar to multi-task learning, there is a challenge in choosing appropriate tasks to act as auxiliaries for the primary tasks. [DCJ+18] proposed to use cosine similarity as an adaptive task weighting to determine when a defined auxiliary task is useful. [NAM+21] applied neural networks to optimally combine auxiliary losses in a non-linear manner.

Our approach is essentially a weighting-based optimisation framework by parameterising these task relationships via learned task weightings. Though these multi-task and auxiliary learning optimisation strategies are encoded to each problem, Auto-λ is designed to solve multi-task learning and auxiliary learning in a unified framework.

Understanding Task Grouping and Relationships These optimisation methods typically assume all training tasks are somewhat related, and the problem of which tasks should be trained together is often overlooked. In general, task relationships are often empirically measured by human intuition rather than prescient knowledge of the underlying structures learned by a neural network. This motivated the study of task relationships in the transfer learning setting [ZSS+18, DR19]. However, [SZC+20a] showed that transfer learning algorithms do not carry over to the multi-task learning domain and instead propose a multi-task specific framework to approximate exhaustive search performance. Further work improved the training efficiency for which the task groupings are computed with only a single training run [FAZ+21]. Rather than exploring fixed relationships, our method instead explores dynamic relationships directly during training.

Meta Learning for Multi-task Learning Meta learning [VD02, HAMS20] has been often used in the multi-task learning setting, such to generate auxiliary tasks in a self-supervised manner [LDJ19, NAM+21] and improve training efficiency on unseen tasks [FAL17, WZL21]. Our work is also closely related to [KS+20, LWS+20] which proposed a task scheduler to learn a task-agnostic features for supervised pre-training, whilst ours learns features that adapt specifically to the primary task; [YLY+21] which applied meta learning to solve multi-objective problems, whilst ours focuses on single-objective problems; [MRY21] which applied meta learning to balance worst-performing tasks, whilst ours balances multi-task learning by finding optimal task relationships. Related to meta learning, our framework is learning to generate suitable and *unbounded* task weightings as a *lookahead* method, as a form of gradient-based meta learning.

Meta Learning for Hyper-parameter Optimisation Since Auto-λ's design models multi-task learning optimisation as learning task weightings λ dynamically via gradients, we may also consider Auto-λ as a meta learning-based hyper-parameter optimisation framework [MDA15, FFS+18, BCC+20] by treating λ as hyper-parameters. Similar to these frameworks, we also formulate a bi-level optimisation problem. However, different to these frameworks, we offer training strategies specifically tailored to the problem of multi-task learning whose goal is not only to obtain good primary task performance, but also explore interesting learning behaviours of Auto-λ from the perspective of task relationships.

3.3 Auto-λ: Unifying Multi-Task and Auxiliary Learning

We now introduce our simple but powerful optimisation framework called Auto-λ, which explores dynamic task relationships through task-specific weightings.

The Design Philosophy. Auto-λ is a gradient-based meta learning framework, a unified optimisation strategy for both multi-task and auxiliary learning problems, which learns task weightings, based on *any combination* of primary tasks. The design of Auto-λ borrows the concept of *lookahead* methods in meta learning literature [FAL17, NAS18], to update parameters at the current state of learning, based on the observed effect of those parameters on a future state. A recently proposed task grouping method [FAZ+21] also applied a similar concept, to compute the relationships based on how gradient updates of one task can affect the performance of other tasks, additionally offering the option to couple with other gradient-based optimisation methods. Auto-λ however is a standalone framework and encodes task relationships *explicitly* with a set of task weightings associated with training loss, directly optimised based on the validation loss of the primary tasks.

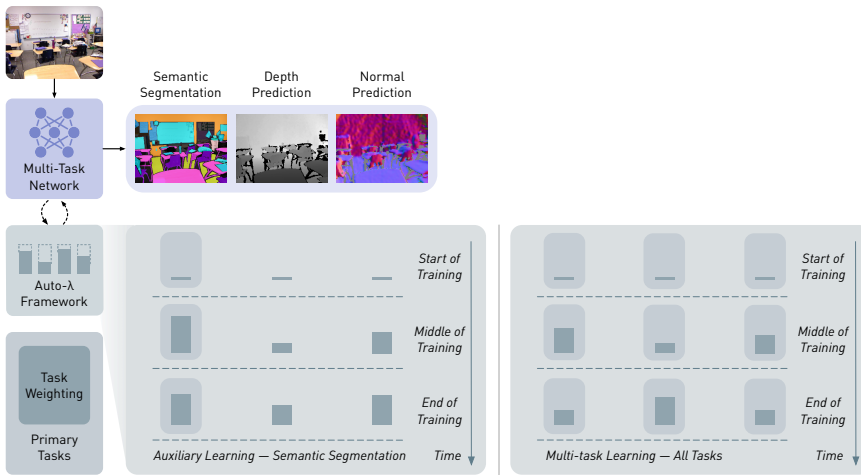


Figure 3.1. Auto-λ framework overview. In Auto-λ, task weightings are dynamically changed along with the multi-task network parameters, in joint optimisation. The task weightings can be updated in both the auxiliary learning setting and the multi-task learning setting. In this example, in the auxiliary learning setting, semantic segmentation is the primary task which we are optimising for. During training, task weightings provide interpretable dynamic task relationships, where high weightings emerge when tasks are strongly related (e.g. normal prediction to segmentation) and low weightings when tasks are weakly related (e.g. depth prediction to segmentation).

Bi-level Optimisation. Let us denote $\boldsymbol{P}$ as the set of indices for all primary tasks defined in $\boldsymbol{T}^{pri}$; (x_i^{val}, y_i^{val}) and $(x_i^{train}, y_i^{train})$ are sampled from the validation and training sets of the i^{th} task space, respectively. The goal of Auto-λ is to find optimal task weightings $\boldsymbol{\lambda}^*$, which minimise the validation loss on the primary tasks, as a way to *measure generalisation*, where the optimal multi-task network parameters $\boldsymbol{\theta}^*$ are obtained by minimising the $\boldsymbol{\lambda}^*$ weighted training loss on all tasks. This implies the following bi-level optimisation problem:

$$\begin{aligned} &\min_{\boldsymbol{\lambda}} \quad \sum_{i \in \boldsymbol{P}} L_i(f(x_i^{val}; \theta_{sh}^*, \theta_i^*), y_i^{val}) \\ &\text{s.t.} \quad \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{K} \lambda_i \cdot L_i(f(x_i^{train}; \theta_{sh}, \theta_i), y_i^{train}). \end{aligned} \tag{3.1}$$

Approximation via Finite Difference. Now, we may rewrite Eq. 3.1 with a simple approximation scheme by updating $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ iteratively with one gradient update each[1]:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{K} \lambda_i \cdot L_i(f(x_i^{train}; \theta_{sh}, \theta_i), y_i^{train}), \tag{3.2}$$

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \beta \nabla_{\boldsymbol{\lambda}} \sum_{i \in \boldsymbol{P}} L_i(f(x_i^{val}; \theta_{sh}', \theta_i'), y_i^{val}), \tag{3.3}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{K} \lambda_i \cdot L_i(f(x_i^{train}; \theta_{sh}, \theta_i), y_i^{train}), \tag{3.4}$$

for which α, β are manually defined learning rates.

The above optimisation requires computing second-order gradients which can produce large memory and slow down training speed. Therefore, we apply finite difference approximation to reduce complexity, similar to other gradient-based meta learning methods [FAL17, LSY19]. For simplicity, let's denote $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}), \mathcal{L}^{pri}(\boldsymbol{\theta}, \boldsymbol{\lambda})$ represent $\boldsymbol{\lambda}$ weighted loss produced by all tasks and primary tasks respectively. The gradient to update $\boldsymbol{\lambda}$ can be approximated by:

$$\begin{aligned} \nabla_{\boldsymbol{\lambda}} \mathcal{L}^{pri}(\boldsymbol{\theta}^*, \mathbb{1}) &\approx \nabla_{\boldsymbol{\lambda}} \mathcal{L}^{pri}(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}), \mathbb{1}) \\ &= \nabla_{\boldsymbol{\lambda}} \mathcal{L}^{pri}(\boldsymbol{\theta}', \mathbb{1}) - \alpha \nabla^2_{\boldsymbol{\theta}, \boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}) \nabla_{\boldsymbol{\theta}'} \mathcal{L}^{pri}(\boldsymbol{\theta}', \mathbb{1}) \\ &\approx \mathbb{0} - \alpha \frac{\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\theta}^+, \boldsymbol{\lambda}) - \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\theta}^-, \boldsymbol{\lambda})}{2\epsilon}, \end{aligned} \tag{3.5}$$

where $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda})$ denotes the updated network weights for a one-step forward model, and $\boldsymbol{\theta}^{\pm} = \boldsymbol{\theta} \pm \epsilon \cdot \nabla_{\boldsymbol{\theta}'} \mathcal{L}^{pri}(\boldsymbol{\theta}', \mathbb{1})$, with a small constant ϵ. $\mathbb{1}$ indicates that all primary

[1] It's easy to extend such formulation for more than one gradient update, while we did not observe significant performance gain.

tasks are of equal importance, but we can also apply different constants based on prior knowledge.[2] We optimise $\boldsymbol{\lambda}$ with vanilla ADAM optimiser [KB14] without any additional regularisation.

Swapping Training Data. In practice, instead of splitting training data into training and validation sets as in the standard meta learning setup, we sample training and validation data to be the different batches in the same training dataset and refresh them for every iteration. We find that this simple swapping training data strategy can learn similar weightings compared to sampling batches in different datasets, making Auto-λ a single-stage framework with end-to-end optimisation.

Stochastic Task Sampling. Eq. 3.2 requires to compute gradients for all training tasks. This can lead to significant GPU memory consumption, particularly in the multi-domain setting for which the task-shared parameters are accumulating gradients for all training tasks. To further save memory, we optimise $\boldsymbol{\lambda}$ in multiple steps, and for each step, we only compute gradients for $K' \ll K$ tasks sampled uniformly. This design allows Auto-λ to be optimised with a *constant memory*, independent of the number of training tasks. In practice, we choose the largest possible K' in each dataset that fits in a GPU to speed up training, and we observe that the performance is robust to a wide range of selections of K'.

3.4 Experiments

To validate the generalisation of Auto-λ, we experiment on both single-domain and multi-domain computer vision and robotics datasets, in multi-task learning and auxiliary learning settings, with various choices of multi-task architectures.

Baselines. In multi-task experiments, we compare Auto-λ with various weighting-based multi-task optimisation methods introduced in Section 2.3: i) **Equal** weighting, ii) **Uncertainty** weighting [KGC18], and iii) **DWA** (Dynamic Weight Average) [LJD19]. In auxiliary learning experiments, we only compare with **GCS** (Gradient Cosine Similarity) [DCJ+18] due to the limited work for this setting.

Optimisation Strategies. By default, we consider each single task as the primary task in the auxiliary learning setting, unless labelled otherwise. In all experiments, Auto-λ's task weightings are initialised to 0.1, a small weighting which assumes that all tasks are *equally*

[2] Note that, $\boldsymbol{\lambda}$ is only applied on the training loss not validation loss, otherwise, we would easily reach trivial solutions $\boldsymbol{\lambda} = \mathbb{0}$. In addition, assuming $\boldsymbol{\theta}' = \boldsymbol{\theta}^*$ is also not applicable, otherwise we have $\nabla_{\boldsymbol{\lambda}} = \mathbb{0}$.

not related. The learning rate to update these weightings is hand-selected for each dataset. For a fair comparison, the optimisation strategies used in all baselines and our method are the same with respect to each dataset and in each data domain.

Results on Dense Prediction Tasks

Training Setup. First, we evaluate Auto-λ with dense prediction tasks in **NYUv2** [NSF12] and **CityScapes** [COR+16], two standard multi-task datasets in a single-domain setting. In NYUv2, we train on 3 tasks: 13-class semantic segmentation, depth prediction, and surface normal prediction, with the same experimental setting as in [LJD19]. In CityScapes, we train on 3 tasks: 19-class semantic segmentation, disparity (inverse depth) estimation, and a recently proposed 10-class part segmentation [dGML+21], with the same experimental setting as in [KGC18]. In both datasets, we train on two multi-task architectures: **Split**: the standard multi-task learning architecture with the vanilla hard parameter sharing, which splits at the last layer for the final prediction for each specific task; **MTAN** [LJD19]: a state-of-the-art multi-task architecture based on task-specific feature-level attention. Both networks are based on ResNet-50 [HZRS16] as the backbone architecture.

Evaluation Metrics. We evaluate segmentation, depth and normal via mean intersection over union (mIoU), absolute error (aErr.), and mean angle distances (mDist.), respectively. Following [MRK19], we also report the overall relative multi-task performance Δ_{MTL} of model m averaged with respect to each single-task baseline b:

$$\Delta_{\text{MTL}} = \frac{1}{K}\sum_{i=1}^{K}(-1)^{l_i}(M_{m,i} - M_{b,i})/M_{b,i}, \tag{3.6}$$

where $l_i = 1$ if lower means better performance for metric M_i of task i, and 0 otherwise.

Noise Prediction as Sanity Check. In auxiliary learning, we additionally train with a *noise prediction* task along with the standard three tasks defined in a dataset. The noise prediction task is generated by assigning a random noise map sampled from a Uniform distribution for each training image. This task is designed to test the effectiveness of different auxiliary learning methods in the *presence of useless gradients.* In all experiments, we train from scratch for a fair comparison among all methods, following the same training setup used in prior works [LJD19, SPFS20, KGC18].

Results. Table 3.1 shows results for CityScapes and NYUv2 datasets in both Split and MTAN multi-task architectures. We can observe that Auto-λ outperforms all baselines in multi-task and auxiliary learning settings across both multi-task networks, and has a

NYUv2	Method	Sem. Seg. [mIoU ↑]	Depth [aErr. ↓]	Normal [mDist. ↓]	Δ_{MTL} ↑
Single-Task	-	43.37	52.24	22.40	-
Split Multi-Task	Equal	44.64	43.32	24.48	+3.57%
	DWA	45.14	43.06	24.17	+4.58%
	Uncertainty	45.98	41.26	24.09	+6.50%
	Auto-λ	**47.17**	**40.97**	**23.68**	**+8.21%**
Split Auxiliary-Task	Uncertainty	45.26	42.25	24.36	+4.91%
	GCS	45.01	42.06	24.12	+5.20%
	Auto-λ [3 Tasks]	**48.04**	40.61	23.31	+9.66%
	Auto-λ [1 Task]	47.80	**40.27**	**23.09**	**+10.02%**
MTAN Multi-Task	Equal	44.62	42.64	24.29	+4.27%
	DWA	45.04	42.81	24.02	+4.89%
	Uncertainty	46.41	40.94	23.65	+7.69%
	Auto-λ	**47.63**	**40.37**	**23.28**	**+9.54%**
MTAN Auxiliary-Task	Uncertainty	44.56	42.21	24.26	+4.55%
	GCS	44.28	44.07	24.03	+3.49%
	Auto-λ [3 Tasks]	47.35	40.10	23.41	+9.30%
	Auto-λ [1 Task]	**47.70**	**39.89**	**22.75**	**+10.69%**
CityScapes	Method	Sem. Seg. [mIoU ↑]	Part Seg. [mIoU ↑]	Disp. [aErr. ↓]	Δ_{MTL} ↑
Single-Task	-	56.20	52.74	0.84	-
Split Multi-Task	Equal	54.03	50.18	0.79	-0.92%
	DWA	54.93	50.15	0.80	-0.80%
	Uncertainty	56.06	**52.98**	0.82	+0.86%
	Auto-λ	**56.08**	51.88	**0.76**	**+2.56%**
Split Auxiliary-Task	Uncertainty	55.72	52.62	0.83	+0.04%
	GCS	55.76	52.19	0.80	+0.98%
	Auto-λ [3 Tasks]	56.42	52.42	0.78	+2.31%
	Auto-λ [1 Task]	**57.89**	**53.56**	**0.77**	**+4.30%**
MTAN Multi-Task	Equal	55.05	50.74	0.78	+0.43%
	DWA	54.71	51.07	0.80	-0.35%
	Uncertainty	56.28	**53.24**	0.82	+1.16%
	Auto-λ	**56.57**	52.67	**0.75**	**+3.75%**
MTAN Auxiliary-Task	Uncertainty	56.13	52.78	0.83	+0.38%
	GCS	55.47	52.75	**0.76**	+2.75%
	Auto-λ [3 Tasks]	57.64	52.77	0.78	+3.25%
	Auto-λ [1 Task]	**58.39**	**54.00**	0.78	**+4.48%**

Table 3.1. Performance on NYUv2 and CityScapes datasets with multi-task learning and auxiliary learning methods in Split and MTAN multi-task architectures. Auxiliary learning is additionally trained with a noise prediction task. Results are averaged over two independent runs, and the best results are highlighted in bold.

particularly prominent effect in the auxiliary learning setting, where it doubles the relative overall multi-task performance compared to auxiliary learning baselines.

We show results for two auxiliary task settings: optimising for just one task (Auto-λ [1 Task]), where the other three tasks (including noise prediction) are purely auxiliary, and

optimising for all three tasks (Auto-λ [3 Tasks]), where only the noise prediction task is purely auxiliary. Auto-λ [3 Tasks] has nearly identical performance to Auto-λ in a multi-task learning setting, whereas the best multi-task baseline *Uncertainty* achieves notably worse performance when trained with noise prediction as an auxiliary task. This shows that standard multi-task optimisation is susceptible to negative transfer, whereas Auto-λ can avoid negative transfer due to its ability to minimise $\boldsymbol{\lambda}$ for tasks that do not assist with the primary task. We also show that Auto-λ [1 Task] can further improve performance relative to Auto-λ [3 Tasks], at the cost of task-specific training for each individual task.

Results on Multi-domain Classification Tasks

Training Setup. We now evaluate Auto-λ on image classification tasks in a multi-domain setting. We train on CIFAR-100 [Kri09] and treat each of the 20 "coarse" classes as one domain, thus creating a dataset with 20 tasks, where each task is a 5-class classification over the dataset's "fine" classes, following [RKR18, YKG+20]. For multi-task and auxiliary learning, we train all methods on a VGG-16 network [SZ15] with the vanilla hard parameter sharing (Split), where each task has a task-specific prediction layer.

CIFAR-100	Method	People	Aquatic Animals	Small Mammals	Trees	Reptiles	Avg.
Single-Task	-	55.37	68.65	72.79	75.37	75.84	82.19
Multi-Task	Equal	**57.73**	73.59	74.41	74.64	76.69	82.46
	Uncertainty	54.14	70.62	74.08	74.62	75.62	82.03
	DWA	55.25	71.54	74.12	**75.68**	76.26	82.26
	Auto-λ	57.57	**74.00**	**75.05**	75.15	**77.55**	**83.92**
Auxiliary-Task	GCS	56.45	71.05	72.93	74.45	76.29	82.58
	Auto-λ	**60.89**	**75.70**	**75.64**	**77.38**	**81.75**	**84.92**

Table 3.2. Performance of 20 tasks in CIFAR-100 dataset with multi-task Learning and auxiliary learning methods. We report the performance from 5 domains giving the lowest single-task performance along with the averaged performance across all 20 domains. Results are averaged over two independent runs, and the best results are highlighted in bold.

Results. In Table 3.2, we show accuracy on the 5 most challenging domains that have the lowest single-task performance, along with the average performance across all 20 domains. Multi-task learning in this dataset is particularly demanding, since we optimise with a $\times 20$ smaller parameter space per task compared to single-task learning. We observe that all multi-task baselines achieve similar overall performance to single-task learning, due to limited per-task parameter space. However, Auto-λ is still able to improve the overall performance by a non-trivial margin. Similarly, Auto-λ can further improve performance in

the auxiliary learning setting, with significantly higher per-task performance in challenging domains with around 5 – 7% absolute improvement in test accuracy.

Results on Robot Manipulation Tasks

Finally, to further emphasise the generality of Auto-λ, we also experiment on visual imitation learning tasks within a multi-domain robotic manipulation setting.

Training Setup. Naively applying behaviour cloning (*e.g.* mapping observations to joint velocities or end-effector incremental poses) for robot manipulations tasks often require thousands of demonstrations [JDJ17]. To circumvent that, we first pre-process the demonstrations by running keyframe discovery [JD21]; a process that iterates over each of the demo trajectories and outputs the transitions where interesting things happen, *e.g.* change in gripper state, or velocities approach zero. The result of the keyframe discovery is a small number of end-effector poses and gripper actions for each of the demonstrations, essentially splitting the task into a set of simple stages. The goal of our behaviour cloning setup is to predict these end-effector poses and gripper actions for new task configurations.

To train and evaluate our method, we select 10 tasks (visualised in Fig. 3.2) from the robot learning environment, RLBench [JMAD20]. Training data are then acquired by first collecting 100 demonstrations for each task, and then running keyframe discovery, to split the task into a smaller number of simple stages to create our behavioural cloning dataset.

Architecture Design and Optimisation Strategies. We optimise an encoder-decoder network which takes the inputs of RGB and point clouds captured by three different cameras (left shoulder, right shoulder and wrist camera), and outputs a continuous 6D pose and a discrete gripper action. The visualisation of the architecture design is illustrated in Fig. 3.3. The 6D pose is composed of a 3-dimensional vector encoding spatial position and a 4-dimensional vector encoding rotation (parameterised by a unit quaternion); the gripper action is represented by a binary scalar indicating gripper open and close. The position and rotation are learned through two separate decoders. The position decoder predicts attention maps based on RGB images, and then we apply spatial (soft) argmax [LFDA16] on the corresponding point cloud to output a 3D spatial position of the attended pixel. We additionally optimise a position offset for each stage of the task, so the predicted position will not be bounded by the position only available in the images. The rotation encoder predicts quaternion and gripper action via direct regression. A learnable task-specific embedding is concatenated to the network bottleneck to distinguish among different tasks in multi-task and auxiliary learning.

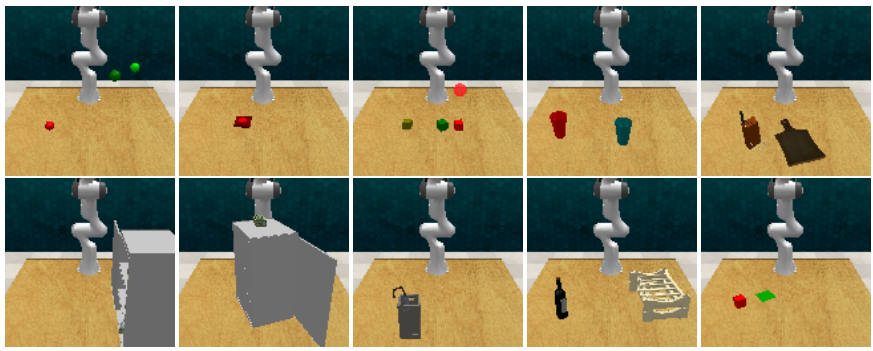


Figure 3.2. A visual illustration of 10 RLBench tasks from the front-facing camera. From top-to-bottom and left-to-right, task names are: *reach target*, *push button*, *pick and lift*, *pick up cup*, *put knife on chopping board*, *take money out of safe*, *put money in safe*, *take umbrella out of umbrella stand*, *stack wine*, and *slide block to target*.

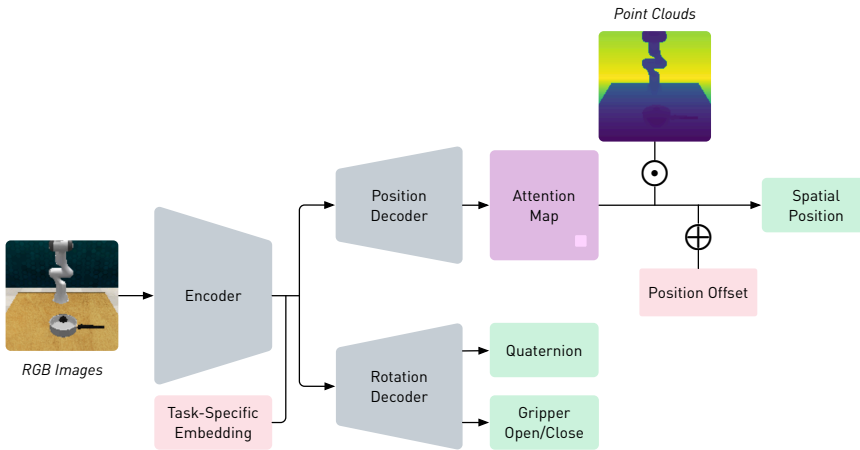


Figure 3.3. Visualisation of the network design for RLBench tasks. We propose a network design that takes in RGB images and point clouds and predicts the next-best end-effector pose and gripper action through two individual decoders. A learnable task-specific embedding is included to distinguish among different tasks in multi-task and auxiliary learning.

Results. In Table 3.3, we report the success rate of each and averaged performance over 10 RLBench tasks. Similar to computer vision tasks, Auto-λ achieves the best performance in both multi-task and auxiliary learning setups, particularly can improve up to 30 – 40% success rate in some multi-stage tasks compared to single-task learning.

RLBench	Method	Reach Target	Push Button	Pick and Lift	Pick Up Cup	Put Knife On Board	Take Money Out Safe	Put Money In Safe	Pick Up Umbrella	Stack Wine	Slide Block To Target	Avg.
Single-Task	-	100	95	82	72	36	38	31	37	23	36	55.0
Multi-Task	Equal	**100**	92	86	69	**40**	57	57	44	16	40	60.1
	Uncert.	**100**	95	75	56	19	60	79	70	16	65	63.5
	DWA	**100**	90	**88**	**82**	35	**66**	57	61	16	66	66.1
	Auto-λ	**100**	95	87	78	31	64	**62**	**80**	**19**	77	**69.3**
Auxiliary-Task	GCS	**100**	97	81	67	42	56	58	60	14	77	65.2
	Auto-λ	**100**	93	**90**	**85**	**49**	**64**	**75**	**74**	**20**	**78**	**72.8**

Table 3.3. Performance of 10 RLBench tasks with multi-task and auxiliary learning methods. We report the success rate with 100 evaluations for each task averaged across two random seeds. Best results are highlighted in bold.

3.5 Visualisations and Interpretability of Task Relationships

In this section, we visualise and analyse the learned weightings from Auto-λ, and find that Auto-λ produces interesting learning strategies with interpretable relationships. Specifically, we focus on using Auto-λ to understand the underlying structure of tasks and transferred task knowledge, introduced next.

Understanding The Structure of Tasks

Task relationships are consistent. Firstly, we observe that the structure of tasks is consistent across the choices of learning algorithms. As shown in Fig. 3.4, the learned weightings with both the NYUv2 and CityScapes datasets are nearly identical, given the same optimisation strategies, *independent* of the network architectures. This observation is also supported by the empirical findings in [ZSS+18, SZC+20a], introduced in Section 2.4, in both transfer and multi-task learning settings.

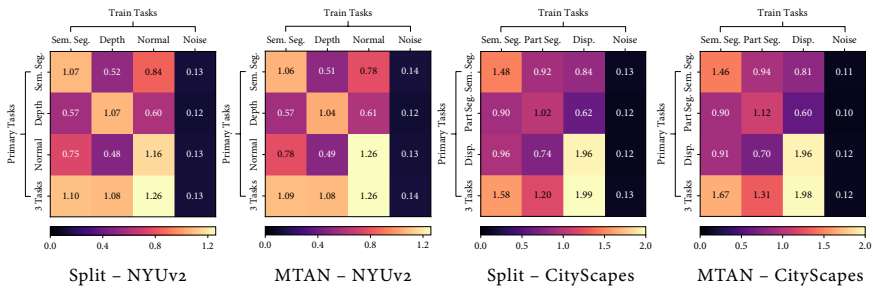


Figure 3.4. Auto-λ explores consistent task relationships in NYUv2 and CityScapes datasets for both Split and MTAN architectures. Higher task weightings indicate stronger relationships and lower task weightings indicate weaker relationships.

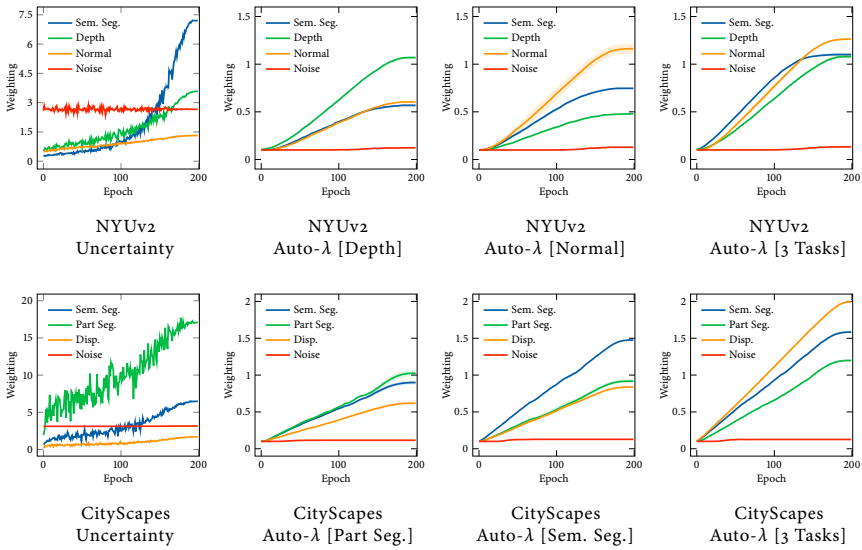


Figure 3.5. Auto-λ learns dynamic relationships based on the choice of primary tasks and can avoid negative transfer. While other baselines such as uncertainty weighting are not able to avoid negative transfer, having a constant weighting on noise prediction task across the entire training stage. $[\cdot]$ represents the choice of primary tasks.

Task relationships are asymmetric. We also find that the task relationships are asymmetric, *i.e.* learning task A with the knowledge of task B is not equivalent to learning task B with the knowledge of task A. A simple example is shown in Fig. 3.5 bottom, where the semantic segmentation task in CityScapes helps the part segmentation task much more than the part segmentation helps the semantic segmentation. This also follows intuition: the representation required for semantic segmentation is a subset of the representation required for part segmentation. This observation is also consistent with recent multi-task learning frameworks focusing on modelling cross-task relationships [LYH16, LYH18, ZSC+20, YKZ21].

Task relationships are dynamic. A unique property of Auto-λ is the ability to explore dynamic task relationships. As shown in Fig. 3.5, we can observe a *weighting cross-over* appears in NYUv2 optimised for 3 tasks auxiliary learning near the end of the training, which can be considered as a learning strategy of *automated curricula*. Further, in Fig. 3.6, we verify that Auto-λ achieves higher per-task performance compared to every combination of fixed task groupings in NYUv2 and CityScapes datasets.

We can also observe that the task relationships inferred by the fixed task groupings are perfectly aligned with the relationships learned with Auto-λ. For example, the performance of semantic segmentation trained with normal prediction (+6.6%) is higher than the performance trained with depth prediction (−6.0%), which is consistent with the fact that the weighting of normal prediction (0.84) is higher than depth prediction (0.52) as shown in Fig. 3.4. In addition, we can observe that the uncertainty weighting [KGC18] is not able to avoid negative transfer from the noise prediction task, having a constant weighting across the entire training stage, which leads to a degraded multi-task performance as in Table 3.1. These observations confirm that Auto-λ is an advanced optimisation strategy, and is able to learn accurate and consistent task relationships.

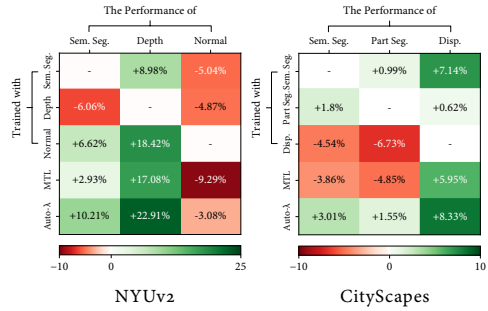


Figure 3.6. Auto-λ achieves best per-task performance compared to every combination of fixed task groupings in NYUv2 and CityScapes trained with Split architecture. This result provides further confirmation that Auto-λ, with its capability to encode dynamic task relationships, excels at learning superior representations.

Understanding Transferred Task Knowledge

Apart from understanding task relationships, we find that Auto-λ can also help us uncover valuable transferred task knowledge. This newfound knowledge can become a valuable resource for making informed decisions when it comes to the manual selection or design of appropriate auxiliary tasks.

Skill v.s. Geometry. In our exploration of robot manipulation tasks, we find that Auto-λ consistently demonstrates a clear preference for optimising weightings based on skills or task trajectories, rather than focussing on object geometry or appearance. An example can be observed in Fig. 3.7, where tasks like "pick up umbrella," "pick up cup," and "pick and lift" emerge as the top three tasks for learning all robot manipulation tasks. This intriguing observation underscores the pivotal importance of the skill of *object grasping* as a fundamental and versatile capability that proves valuable across a variety of robotic manipulation tasks, notwithstanding the differences in the objects involved.

Moreover, in the case of tasks such as "put knife on board" and "put money out safe" benefit significantly from the task of "slide block to target", indicating the critical role of

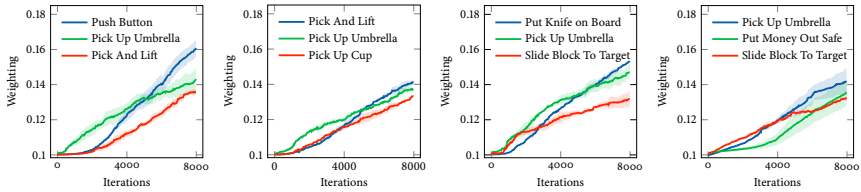


Figure 3.7. Learning dynamics of Auto-λ in the auxiliary learning setting for 4 selected RLBench tasks. Specifically, we highlight the top three tasks with the highest task weightings observed in each experiment. We can observe that Auto-λ exhibits a deliberate emphasis on mastering fundamental tasks such as object grasping and pushing during the initial phases of training. It is only after consolidating these foundational skills that the system gradually shifts its focus towards the primary tasks, providing a clear indication of automated curriculum learning.

object pushing as another fundamental skill in robotic manipulation. Across all these four experiments, it's noteworthy that these fundamental tasks initially carry higher weightings than the primary tasks during the early stages of training and gradually diminish as training progresses, signifying a strategic shift towards prioritising the acquisition of foundational skills early as part of an automated curriculum learning process. This strategy enhances the multi-task model's overall generalisation and ultimately leads to better performance.

In-Domain v.s. Out-of-Domain. In multi-domain classification tasks, our observations reveal that Auto-λ occasionally identifies related domains to the primary task that lack semantic connections. For instance, it recognises a relationship between "Fish" and "Small Mammals" in connection with "Aquatic Mammals," justified by their shared characteristic of small size and their potential commonality in water-related environmental features. However, such connections do not align with human intuition when it comes to domains like "Trees," which appear semantically unrelated.

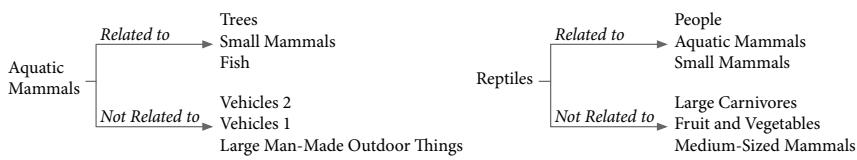


Figure 3.8. Top 3 most related and unrelated domains when classifying Reptiles and Aquatic Mammals in the auxiliary learning setting on the CIFAR-100 dataset. Notably, the domain relationships discovered by Auto-λ do not conform to conventional human intuition. Instead, they suggest that the generalisation capabilities might be rooted in non-obvious and unexpected associations, emphasising the intriguing potential of automated learning systems to unveil novel insights beyond traditional human understanding.

Similarly, Auto-λ establishes a connection between "Aquatic Mammals" and "Small Mammals" in relation to "Reptiles" but not when considering "People". This observation underscores the intriguing notion that domain-specific knowledge sometimes necessitates the inclusion of out-of-domain knowledge to enhance generalisation. This observation highlights the complex and nuanced nature of domain relationships, requiring a broader perspective beyond the immediate domain semantics to improve generalisation effectively.

3.6 Robustness and Ablation Analysis

Finally, we present some additional analyses on NYUv2 dataset with Split multi-task architecture to understand the behaviour of Auto-λ with respect to different hyper-parameters and other types of optimisation strategies.

Robustness on Training Strategies

Here, we evaluate different hyper-parameters trained with Auto-λ [3 Tasks] in the auxiliary learning setting. As seen in Fig. 3.9, we find that Auto-λ optimised with direct second-order gradients offers very similar task weightings compared to when optimised with approximated first-order gradients (< 0.05 averaged difference across training time in all three tasks), resulting a near-identical multi-task performance. In addition, we discover that using first-order gradients may speed up training time roughly $\times 2.3$.

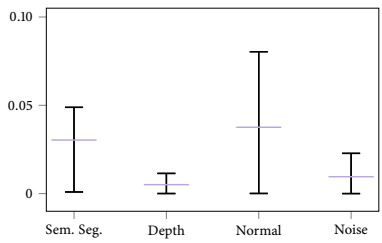


Figure 3.9. Mean and the range of per-task weighting difference for Auto-λ [3 Tasks] optimised with direct and approximated gradients in NYUv2 dataset. We confirm that Auto-λ optimised with direct second-order and approximated first-order gradients would obtain near identical task weightings.

	Task Weightings				Δ_{MTL}
	Sem. Seg.	Depth	Normal	Noise	
$Init = 0.01$	0.97	0.95	1.1	0.02	+8.98%
$Init = 1.0$	2.00	2.11	2.08	1.00	+1.42%
$LR = 3 \cdot 10^{-5}$	0.43	0.37	0.46	0.11	+8.53%
$LR = 3 \cdot 10^{-4}$	3.10	3.34	3.26	0.15	+8.56%
$LR = 1 \cdot 10^{-3}$	10.5	10.5	10.3	0.23	+5.04%
No Swapping	2.67	2.76	2.98	0.20	+8.17%
Our Setting	1.11	1.06	1.26	0.12	**+9.66%**

Table 3.4. Multi-task performance in NYUv2 dataset trained with Auto-λ [3 Tasks] with different hyper-parameters. The default setting is $Init = 0.1$, $LR = 1 \cdot 10^{-4}$ and with training data swapping. Auto-λ is sensitive to hyper-parameters, and initialising with a small weighting and a suitable learning rate is important to achieve a good performance.

In Table 3.4, we show that initialising with a small weighting and a suitable learning rate is important to achieve a good performance. A larger learning rate leads to saturated

weightings which cause unstable network optimisation; and a larger initialisation would not successfully avoid negative transfer. In addition, optimising network parameters and task weightings with different data is also essential (to properly measure generalisation), which otherwise would slightly decrease performance.

Comparison to Gradient-based Methods

Finally, since Auto-λ is a weighting-based optimisation method, it can naturally be combined with gradient-based methods to further improve performance. We evaluate Auto-λ along with the other weighting-based baselines described in Sec. 3.4, when combined with recently proposed state-of-the-art gradient-based methods designed for multi-task learning: GradDrop [CNH+20], PCGrad [YKG+20] and CAGrad [LLJ+21]. We train all methods in NYUv2 dataset with standard 3 tasks in the multi-task learning setup.

	Equal	DWA	Uncertainty	Auto-λ
Vanilla	+3.57%	+4.58%	+6.50%	**+8.21%**
+ GradDrop	+4.65%	+5.93%	+6.22%	**+8.12%**
+ PCGrad	+5.09%	+4.37%	+6.20%	**+8.50%**
+ CAGrad	+7.05%	+8.08%	+9.65%	**+11.07%**

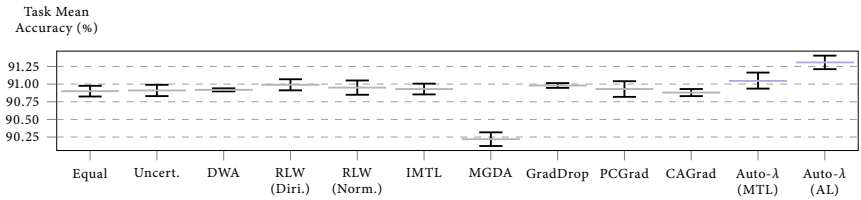
Table 3.5. NYUv2 multi-task performance trained with weighting-based and gradient-based methods in the multi-task learning setting. Auto-λ surpasses other gradient-based methods in the vanilla setting, and can further improve performance when combined with a more advanced gradient-based method.

In Table 3.5, we can observe that Auto-λ remains the best optimisation method even compared to other gradient-based methods in the vanilla setting (with Equal weighting). Further, combined with a more advanced gradient-based method such as CAGrad [LLJ+21], Auto-λ can reach even higher performance.

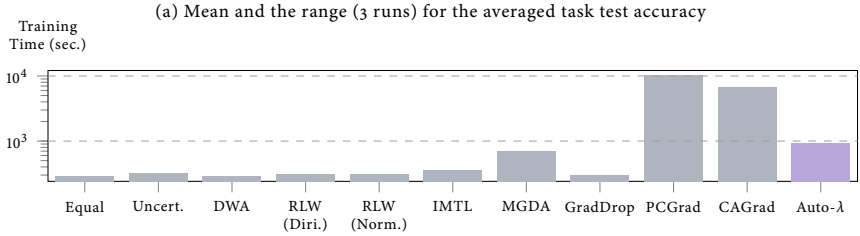
Comparison to Strong Regularisation Methods

Finally, recent works [LYZT22, KDPK+22] suggest that many multi-task optimisation methods can be interpreted as forms of implicit regularisation. They show that when using strong regularisation and stabilisation techniques from single-task learning, training by simply minimising the sum of task losses, or with randomly generated task weightings, can achieve performance competitive with complex multi-task methods.

As such, we now evaluate Auto-λ, along with all multi-task baselines evaluated in Section 3.4, as well as all multi-task methods included in the original work of [KDPK+22], coupled with this strong regularisation on CelebA dataset [LLWT15], for a challenging 40-task classification problem. We train these multi-task methods with the exact same experimental setting in [KDPK+22] for a fair comparison. To conclude, we compare with: Equal (Unit. Scal.) [KDPK+22], DWA [LJD19], RLW (with weights sampled from a Dirichlet and a Normal



(a) Mean and the range (3 runs) for the averaged task test accuracy



(b) Mean per-epoch training time (10 repetitions)

Figure 3.10. Average test accuracy and per-epoch training time of various multi-task optimisation methods trained with strong regularisation on CelebA dataset. Multi-task methods tend to yield equivalent or worse performance compared to equal weighting. However, Auto-λ emerges as the exception to this trend, showcasing superior results. It's worth noting that Auto-λ does demand a longer training time than most weighting-based methods but still runs considerably faster compared to gradient-based methods. Part of the results are directly borrowed from [KDPK+22].

Distribution) [LYZT22], IMTL [LLK+21], MGDA [SK18], GradDrop [CNH+20], PCGrad [YKG+20], and CAGrad [LLJ+21], for a total of 10 multi-task optimsiation methods.

To our surprise, though most methods achieve similar performance, which is consistent with the findings in [KDPK+22], Auto-λ is still able to improve performance (marginally in the multi-task learning setting, and significantly in the auxiliary learning setting) with a clear statistical significance. The improvement is especially pronounced in the auxiliary learning mode, which is the unique learning mode of Auto-λ, showing the multi-task network's generalisation imposed from Auto-λ is more than implicit regularisation.

In addition, we also compare training time across these multi-task methods, and we re-scale the training time in our implementation to [KDPK+22]'s setting for a fair comparison. We can observe that Auto-λ requires three times longer the training time than Equal weighting (Unit. Scal.) [KDPK+22], in consistent with its theoretical design, since Auto-λ needs to compute additional two forward and two backward passes to approximate the second-order gradients. Though Auto-λ requires longer training time, it can outperform other multi-task methods, and is still an order of magnitude faster than some gradient-based methods such as PCGrad [YKG+20] and CAGrad [LLJ+21].

3.7 Conclusions, Limitations and Discussions

In this chapter, we have presented Auto-λ, a unified multi-task and auxiliary learning optimisation framework. Auto-λ operates by exploring task relationships in the form of task weightings in the loss function, which are allowed to dynamically change throughout the training period. This allows optimal weightings to be determined at any one point during training, and hence, a more optimal period of learning can emerge than if these weightings were fixed throughout training. Auto-λ achieves state-of-the-art performance in both computer vision and robotics benchmarks, for both multi-task learning and auxiliary learning, even when compared to optimisation methods that are specifically designed for just one of those two settings.

For transparency, we now discuss some limitations of Auto-λ that we have noted during our implementations, and we discuss our thoughts on future directions with this work.

Advanced Training Strategies To achieve optimal performance, Auto-λ still requires hyper-parameter search (although the performance is primarily sensitive to only one parameter, the learning rate, making this search relatively simple). Some advanced training techniques, such as incorporating weighting decay or bounded task weightings, might be helpful to find a general set of hyper-parameters that work for all datasets.

Training Speed The design of Auto-λ requires computing second-order gradients, which is computationally expensive. To address this, we apply a finite-difference approximation to reduce the complexity, which requires the addition of only two forward passes and two backward passes. However, this may still be slower than alternative optimisation methods.

Single Task Decomposition Auto-λ can optimise on any type of task. Therefore, it is natural to consider a compositional design, where we decompose a single task into a series of smaller sub-tasks, *e.g.* to decompose a multi-stage manipulation task into a sequence of discrete stages. By applying Auto-λ to these individual sub-tasks, we open the door to intriguing possibilities for exploring novel learning behaviours that can enhance the efficiency of single-task learning.

Open-ended Learning Given the dynamic and adaptable nature of task exploration by Auto-λ, it presents an intriguing avenue for investigation into its integration within an open-ended learning system. In such a system, tasks are continuously introduced and incorporated during the training process. The inherent flexibility of Auto-λ to dynamically

optimise task relationships suggests that it could serve as a natural fit for open-ended learning, where this integration could occur seamlessly without the need for manual balancing or adjusting the learning system for each new task addition.

Impact on Future Research Auto-λ has become widely recognised as a prominent and competitive baseline for advancing multi-task and auxiliary learning optimisation research [MVS22, JCP+24, DFL23, SNG+23, SRZP23, LFSL24]. Its applications extends across various domains, including computational pathology [ZWP22] and blind image quality assessment [ZZW+23], where it has demonstrated the capability to improve model generalisation and performance. Additionally, Auto-λ has sparked a line of new research in multi-task robot manipulation, particularly in neural architecture and action space design. These efforts have led to significant breakthroughs in general-purpose multi-modal robotics research guided by natural language [GCG+22, GXGF23, SMF23].

4

Exploring Semantic Relationships with Contrastive Learning

In the previous chapter, we have delved into the intricate dynamics of task relationships within a multi-task learning framework. In this chapter, we shift our focus to the exploration of structured relationships in an auxiliary learning setting, particularly within the domain of semantic segmentation. As such, we introduce a contrastive learning method designed at a regional level named as ReCo, to improve the performance of semantic segmentation models by leveraging the inherent structure of semantic class relationships.

ReCo employs pixel-level contrastive learning, targeting a sparse selection of challenging negative pixels, imposing minimal additional memory footprint. ReCo is designed as an auxiliary learning framework, that can seamlessly complement existing segmentation networks, providing consistent performance improvements across both semi-supervised and supervised semantic segmentation methods, achieving smoother segmentation boundaries and faster convergence. The strongest effect is in a semi-supervised learning setting with a very limited number of labels. Remarkably, ReCo empowers us to achieve high-quality semantic segmentation models to be trained with minimal labelled data, requiring as few as just five labelled examples for each semantic class.

4.1 The Challenge of Semantic Segmentation

Semantic segmentation is an essential part of applications such as scene understanding and autonomous driving, whose goal is to assign a semantic label to each pixel in an image. Significant progress has been achieved by the use of large datasets with high-quality human annotations. However, labelling images with pixel-level accuracy is time-consuming and expensive; for example, labelling a single image in CityScapes can take more than 90 minutes [COR+16]. When deploying semantic segmentation models in practical applications where only limited labelled data are available, high-quality ground-truth annotation is a significant bottleneck.

To reduce the need for labelled data, there is a recent surge of interest in leveraging unlabelled data for semi-supervised learning. Previous methods include improving segmentation models via adversarial learning [HTL+19, MTB19] and self-training [ZYL+19, ZYKW18, ZZW+21]. Others focus on designing advanced data augmentation strategies to generate pseudo image-annotation pairs from unlabelled images [OTPS21, FAL+20].

In both semi-supervised and supervised learning, a semantic segmentation model often predicts smooth label maps, because neighbouring pixels are usually of the same class, and rarer high-frequency regions are typically only found in object boundaries. This learning bias naturally produces blurry contours and regularly mislabels rare objects. After carefully examining the label predictions, we further observe that wrongly labelled pixels are typically confused with very few other classes; *e.g.* a pixel labelled as "rider" has a much higher chance of being wrongly classified as "person", compared to "building" or "bus". By understanding this class structure, learning can be actively focused on the most challenging pixels to improve overall segmentation quality.

4.2 Related Work

Semantic Segmentation The advances of semantic segmentation commonly rely on designing more powerful deep convolutional neural networks. Fully convolutional networks (FCNs) [LSD15] are the foundation of modern segmentation network design. They were later improved with dilated/atrous convolutions with larger receptive fields, capturing more long range information [CPK+17, CZP+18]. Alternative approaches include encoder-decoder architectures [RFB15, KGHD19], sometimes using skip connections [RFB15] to refine filtered details.

A parallel direction is to improve optimisation strategies, by designing loss functions that

better respect class imbalance [LGG+17] or using point-wise rendering strategy to refine uncertain pixels from high-frequency regions improving the label quality [KWHG20]. ReCo is built upon this line of research, a *model-agnostic* framework to improve segmentation by providing additional supervision on hard pixels.

Semi-supervised Classification and Segmentation The goal of semi-supervised learning is to improve model performance by taking advantage of a large amount of unlabelled data during training. Here consistency regularisation and entropy minimisation are two common strategies. The intuition is that the network's output should be invariant to data perturbation and geometric transformation. Based on these strategies, many semi-supervised methods have been developed for image classification [SBC+20, TV17, BCG+19, KMHK20].

However, for segmentation, generating effective pseudo-labels and well-designed data augmentation are non-trivial. Some solutions improved the quality of pseudo-labelling, using adversarial learning [HTL+19, MTB19] and class activation maps [ZZZ+21]; or enforcing consistency from different augmented images [FAL+20, OTPS21], perturbed features [OHT20] and different networks [KQL+20]. In this work, we show that rather than designing a more advanced pseudo-labelling strategy, we can improve the performance of current semi-supervised segmentation methods by jointly training with a suitable auxiliary task.

Contrastive Learning Contrastive learning learns a similarity function to bring views of the same data closer in representation space, whilst pushing views of different data apart. Most recent contrastive frameworks learn similarity scores based on *global representations* of the views, parameterising data with a single vector [HFW+20, CKNH20, KTW+20]. *Dense representations*, on the other hand, rely on pixel-level representations and naturally provide additional supervision, capturing fine-grained pixel correspondence. Contrastive pre-training based on dense representations has recently been explored, and shows better performance in dense prediction tasks, such as object detection and keypoint detection [WZS+21, OPAB+20].

Contrastive Learning for Semantic Segmentation Contrastive learning has been recently studied to improve semantic segmentation, with a number of different design strategies. [ZTRR21] and [ZVM+21] both perform contrastive learning via pre-training, based on the generated auxiliary labels and ground-truth labels respectively, but at the cost of huge memory consumption. In contrast, ours performs contrastive learning whilst requiring much less memory, via active sampling. In concurrent work, [WZY+21, ASF+21] also perform contrastive learning with active sampling. However, whilst both these meth-

ods are applied to a stored feature bank, ours focuses on sampling features on-the-fly. Active sampling in [ASF+21] is further based on learnable, class-specific attention modules, whilst ours only samples features based on relation graphs and prediction confidence, without introducing any additional computation overhead, which results in a simpler and much more memory-efficient implementation.

4.3 ReCo: Regional Contrast Learning for Semantic Segmentation

Here we propose ReCo, a contrastive learning framework designed at a regional level. Specifically, ReCo is a new auxiliary loss that helps semantic segmentation to not only learn from *local context* from neighbouring pixels, but also from *global semantic class relationships* across the entire dataset. ReCo performs supervised or semi-supervised contrastive learning on a pixel-level dense representation, as visualised in Fig. 4.1. For each semantic class in a training mini-batch, ReCo samples a set of pixel-level representations (queries), and encourages them to be close to the class mean averaged across all representations in this class (positive keys), and simultaneously pushes them away from representations sampled from other classes (negative keys).

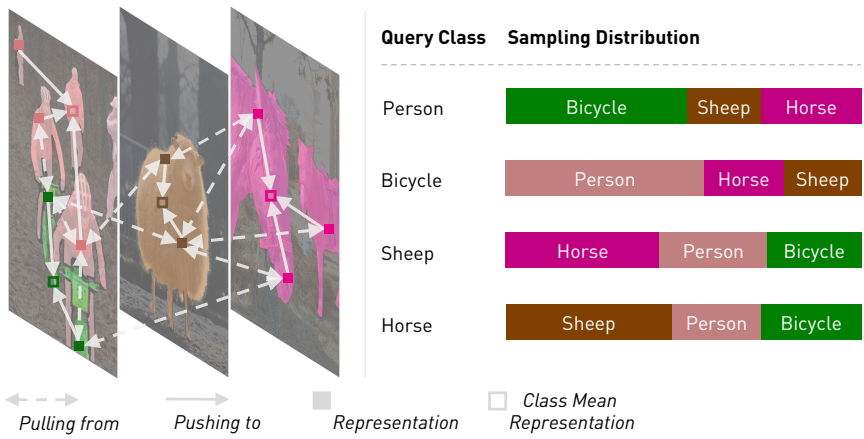


Figure 4.1. ReCo framework overview. ReCo pushes representations within a class closer to the class mean representation, while simultaneously pushing these representations away from negative representations sampled from different classes. The sampling distribution from negative classes is adaptive to each query class, and dynamically updated during training. For example, due to the strong relation between "bicycle" and "person" class, ReCo will sample more representations in "bicycle" class, when learning "person" class, compared to other classes.

Pixel-Level Contrastive Learning

Let $(\mathcal{X}, \mathcal{Y})$ be a training dataset with training images $x \in \mathcal{X}$ and their corresponding C-class pixel-level segmentation labels $y \in \mathcal{Y}$, where y can be either provided in the original dataset (supervised learning setting) or generated automatically as pseudo-labels (semi-supervised learning setting). A segmentation network f is then optimised to learn a mapping $f_\theta : \mathcal{X} \mapsto \mathcal{Y}$, parameterised by network parameters θ. This segmentation network f can be decomposed into two parts: an encoder network: $f^e_\phi : \mathcal{X} \mapsto \mathcal{Z}$, and a decoder classification head $f^d_{\psi_c} : \mathcal{Z} \mapsto \mathcal{Y}$. To perform pixel-level contrastive learning, we additionally attach a decoder representation head $f^d_{\psi_r}$ on top of the encoder network f^e_ϕ, parallel to the classification head, mapping the encoded feature into a higher m-dimensional dense representation with the same spatial resolution as the input image: $f^d_{\psi_r} : \mathcal{Z} \mapsto \mathcal{R}, \mathcal{R} \in \mathbb{R}^m$. This representation head is only applied during training to guide the classifier using the ReCo loss as an auxiliary task, and is removed during inference.[1]

A pixel-level contrastive loss is a function which encourages queries r_q to be similar to the positive key r_k^+, and dissimilar to the negative keys r_k^-. All queries and keys are sampled from the decoder representation head: $r_q, r_k^{+,-} \in \mathcal{R}$. In ReCo, we use a pixel-level contrastive loss in a supervised or semi-supervised manner across all available semantic classes in each mini-batch, with the distance between keys and queries measured by their normalised dot product. The general formation of the ReCo loss $L_{\texttt{reco}}$ is then defined as:

$$L_{\texttt{reco}} = \sum_{c \in \mathcal{C}} \sum_{r_q \sim \mathcal{R}_q^c} -\log \frac{\exp(r_q \cdot r_k^{c,+}/\tau)}{\exp(r_q \cdot r_k^{c,+}/\tau) + \sum_{r_k^- \sim \mathcal{R}_k^c} \exp(r_q \cdot r_k^-/\tau)}, \tag{4.1}$$

for which $\mathcal{C}$ is a set containing all available classes in the current mini-batch, τ is the temperature control of the softness of the distribution, $\mathcal{R}_q^c$ represents a query set containing all representations whose labels belong to class c, $\mathcal{R}_k^c$ represents a negative key set containing all representations whose labels do not belong to class c, and $r_k^{c,+}$ represents the positive key which is the mean representation of class c. Suppose $\mathcal{P}$ is a set containing all pixel coordinates with the same resolution as $\mathcal{R}$, these queries and keys are then defined as:

$$\mathcal{R}_q^c = \bigcup_{[u,v] \in \mathcal{P}} \mathbb{1}(y_{[u,v]} = c)\, r_{[u,v]}, \quad \mathcal{R}_k^c = \bigcup_{[u,v] \in \mathcal{P}} \mathbb{1}(y_{[u,v]} \neq c)\, r_{[u,v]} \tag{4.2}$$

$$r_k^{c,+} = \frac{1}{|\mathcal{R}_q^c|} \sum_{r_q \in \mathcal{R}_q^c} r_q. \tag{4.3}$$

[1] Using the notations introduced in Chapter 2, we are optimising a 2-task auxiliary learning problem with $\theta = \{\phi, \psi_c, \psi_r\}$, where ϕ is the task-shared parameters and $\psi_{r,c}$ are task-specific parameters.

Active Hard Sampling on Queries and Keys

To perform pixel-level contrastive learning on all available pixels in high-resolution training images would be computationally expensive and require massive memory. To address this challenge, we introduce active hard sampling strategies to optimise the selection of only a sparse set of queries and keys, focusing computational efforts on the most informative and challenging pixel pairs.

Active Key Sampling. When classifying a pixel, a semantic network often exhibits uncertainty only for a very few number of candidates among all available classes. This uncertainty typically arises from either close *spatial relationships* (*e.g.* "rider" and "bicycle") or *semantic similarities* (*e.g.* "chair" and "sofa"). To reduce this uncertainty, we propose a novel approach for sampling negative keys in a *non-uniform* manner, based on the relative distance between each negative key class and the query class.

This involves constructing a pair-wise class relationship graph, denoted as G, where $G \in \mathbb{R}^{|\mathcal{C}|\times|\mathcal{C}|}$. The relationship graph is computed and dynamically updated for each mini-batch, serving as a dynamic representation of the relationships and affinities between different classes. This concept aligns with a similar approach introduced in Auto-λ, where dynamic relationships play a crucial role in the learning process, allowing the model to adapt and respond to the evolving nature of the data and tasks.

The pair-wise relationship is measured by the normalised dot product between the mean representation from a pair of two classes and is defined as:

$$G[p,q] = \left(r_k^{p,+} \cdot r_k^{q,+}\right), \quad \forall p,q \in \mathcal{C}, \text{ and } p \neq q. \tag{4.4}$$

We further apply SoftMax to normalise these pair-wise relationships among all negative classes j for each query class c, producing a distribution: $\exp(G[c,i])/\sum_{j\in\mathcal{C},j\neq c}\exp(G[c,j])$. We sample negative keys for each class i based on this distribution, when optimising the corresponding query class c. By leveraging this relationship graph, we can intelligently select negative keys that are more relevant to the query class, effectively reducing uncertainty by focusing on classes that are closely related or similar to the target class.

Active Query Sampling. In semantic segmentation, class imbalance is a natural challenge that can lead to overfitting on common classes, such as "road" and "building" in the CityScapes dataset, or the ubiquitous "background" class in the Pascal VOC dataset. These common classes occupy the majority of pixel space in training images, and as a result,

randomly sampling queries would disproportionately undersample rare classes, providing minimal supervision to these crucial but less frequent classes.

To tackle this issue, we adopt a different strategy by sampling hard queries — those corresponding to pixel prediction confidence levels below a pre-defined threshold. By doing so, we ensure that the ReCo loss guides the segmentation network to provide more targeted and appropriate supervision to these less certain pixels. This approach helps balance the learning process, ensuring that both common and rare classes receive adequate attention and training, ultimately contributing to improved segmentation performance and reducing the risk of overfitting on dominant classes.

The easy and hard queries are defined as follows, and visualised in Fig. 4.2,

$$\mathcal{R}_q^{c,\,easy} = \bigcup_{r_q \in \mathcal{R}_q^c} \mathbb{1}(\hat{y}_q > \delta_s) r_q, \quad \mathcal{R}_q^{c,\,hard} = \bigcup_{r_q \in \mathcal{R}_q^c} \mathbb{1}(\hat{y}_q \leq \delta_s) r_q, \tag{4.5}$$

where $\hat{y}_q$ is the predicted confidence of label c after the SoftMax operation corresponding to the same pixel location as r_q, and δ_s is the user-defined confidence threshold.

(a) Confidence Map (b) Easy Queries (c) Hard Queries

Figure 4.2. Visualisation of easy and hard queries. Easy and hard queries (shown in white) are determined from the predicted confidence map in the Cityscapes dataset. Here we set the confidence threshold $\delta_s = 0.97$. We can observe that most hard queries are concentrated around small objects or object boundaries.

Improving Semantic Segmentation with ReCo

ReCo can easily be added to modern supervised and semi-supervised segmentation methods without changing the training pipeline, with *no additional cost* at inference time. To incorporate ReCo, we simply add an additional representation head ψ_r as described in Section 4.3, and apply the ReCo loss (in Eq. 4.1) to this representation using the sampling strategy introduced in Section 4.3. Following prior contrastive learning methods [HFW+20], we only compute gradients on queries, for better training stabilisation.

In the supervised segmentation setting, where all training data have ground-truth annotations, we apply the ReCo loss on dense representations corresponding to all valid pixels.

The overall training loss is then the linear combination of the supervised cross-entropy loss and the ReCo loss:

$$L_{total} = L_{supervised} + L_{reco}. \tag{4.6}$$

In the semi-supervised segmentation setting, where only part of the training data has ground-truth annotations, we apply the Mean Teacher framework [TV17] following prior state-of-the-art semi-supervised segmentation methods [OTPS21, MTB19]. Instead of using the original segmentation network f_θ (referred to as the student model), we instead use $f_{\theta'}$ (referred to as the teacher model) to generate pseudo-labels from unlabelled images, where θ' is a moving average of the previous state of θ during training optimisation: $\theta'_t = \lambda\theta'_{t-1} + (1-\lambda)\theta_t$, with a decay parameter $\lambda = 0.99$. The teacher model can be viewed as a temporal ensemble of student models across different stages of training, resulting in more stable and consistent predictions for unlabelled images. The student model f_θ is then used to train on the augmented unlabelled images, with pseudo-labels as the ground-truths.

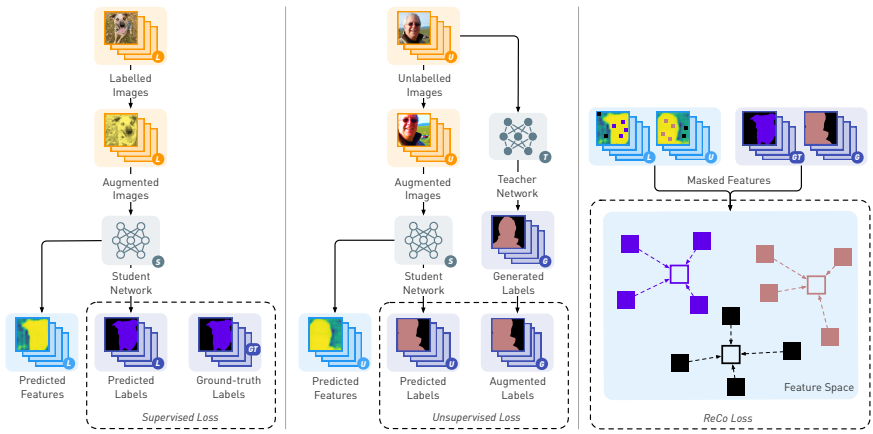


Figure 4.3. Visualisation of the ReCo framework applied to semi-supervised segmentation and trained with three losses. A supervised loss is computed based on labelled data with ground-truth annotations. An unsupervised loss is computed for unlabelled data with generated pseudo-labels. And finally, a ReCo loss is computed based on pixel-level dense representation predicted from both labelled and unlabelled images.

For all pixels with defined ground-truth labels, we apply the ReCo loss similarly to the supervised segmentation setting. For all pixels without such labels, we only sample pixels whose predicted pseudo-label confidence is greater than a threshold δ_w. This avoids sampling pixels that are likely to have incorrect pseudo-labels.

We apply the ReCo loss to a combined set of pixels from both labelled and unlabelled images. The overall training loss for semi-supervised segmentation is then the linear combination of supervised cross-entropy loss (on ground-truth labels), unsupervised cross-entropy loss (on pseudo-labels generated by the teacher model), and ReCo loss:

$$L_{total} = L_{supervised} + \eta \cdot L_{unsupervised} + L_{reco}, \tag{4.7}$$

where η is defined as the percentage of pixels whose predicted confidence is greater than δ_s, a scalar coefficient that regulates the contribution for unsupervised loss, following prior methods [OTPS21, MTB19]. This re-weighting of the unsupervised loss helps ensure that the segmentation network does not become dominated by gradients originating from uncertain pseudo-labels, a scenario that is more prevalent during the early stages of training. The ReCo framework for semi-supervised segmentation is visually represented in Fig. 4.3, providing a detailed overview of how these components interact within the training process.

4.4 Experiments

We evaluate ReCo on supervised and semi-supervised segmentation. We introduce our new benchmark design and datasets, along with their results and visualisations presented in this section. We provide an ablative analysis of important hyper-parameters along with the effect of query and key sampling strategies in Section 4.5.

Experiment Setup

Semi-Supervised Segmentation Benchmark Redesign. We propose two modes of semi-supervised segmentation tasks aimed at different applications.

1. *Partial Dataset Full Labels*: A small subset of the images is trained with complete ground-truth labels, while the remaining training images are unlabelled. When creating the labelled dataset, we sample labelled images based on two conditions: i) Each sampled image must contain a *distinct* number of classes greater than a manually-defined threshold. ii) Each sampled image must contain one of the *least sampled classes* in the previously sampled images.

   These conditions are carefully designed to ensure a consistent class distribution across different random seeds and guarantee the representation of all classes. This setup allows us to evaluate the performance of semi-supervised methods with a very limited number of labelled images, without concerns about the complete absence of rare classes. This

mode assesses the model's ability to generalise to semantic classes with only a few examples while benefiting from accurate boundary information.

2. *Partial Labels Full Dataset*: All images are trained with partial labels, but only a few percentages of labels are provided for each class in each training image. We create the dataset by first randomly sampling a pixel for each class, and then continuously applying a $[5 \times 5]$ square kernel for dilation until we meet the percentage criteria.

   This mode evaluates the model's ability to learn the semantic class completion in the presence of many examples but with limited or no boundary information. It simulates scenarios where semantic class information is required to be inferred or completed from a dataset with minimal annotations.

By introducing these two distinct modes, we can comprehensively assess the performance and capabilities of semi-supervised segmentation methods in different practical scenarios, aligning with the specific challenges and requirements of each application.

Datasets. We experiment on popular segmentation datasets: Cityscapes [COR+16] and Pascal VOC 2012 [EEVG+15] in both partial and full label setting. We also evaluate on a more difficult indoor scene segmentation dataset SUN RGB-D [SLX15] in the full label setting only, mainly due to the low-quality annotations making it difficult to be fairly evaluated in the partial label setting. In the full label setting, all three datasets are evaluated in four cases containing three semi-supervised settings, and one fully supervised setting (training on all labelled images). In a semi-supervised setting, we sample labelled images to make sure the least appeared class has appeared at least in 5, 15 and 50 images respectively, in all three datasets, among which, the labelled images for CityScapes, Pascal VOC and SUN RGB-D contain at least 12, 3 and 1 semantic classes, respectively.

In the partial label setting, both the CityScapes and Pascal VOC datasets are evaluated in four cases, by sampling 1, 1%, 5% and 25% labelled pixels for each semantic class in each training image. An example of the partially labelled dataset is shown in Fig. 4.4.

Strong Baselines. Prior semi-supervised segmentation methods are typically designed with different backbone architectures, and trained with different strategies, which makes it hard to compare them fairly. In this work, we standardise the baselines and implement four strong semi-supervised segmentation methods ourselves: i) *S4GAN* [MTB19]: an adversarial learning based semi-supervised method, ii) *CutOut* [FAL+20]: an image augmentation strategy to generate new data by cutting out a random patch in an image, iii)

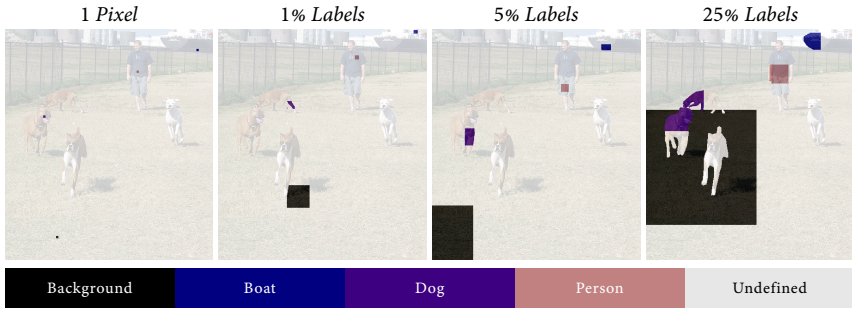


Figure 4.4. Example of training labels for Pascal VOC dataset in Partial Labels Full Dataset setting. 1 Pixel labels are zoomed 5 times for better visualisation.

CutMix [FAL+20]: an image augmentation strategy to generate new data by attaching a random patch extracted from one image to another image, and iv) *ClassMix* [OTPS21]: an image augmentation strategy by attaching random semantic classes extracted from one image to another image. Our implementations for all baselines obtain performance on par with, and most of the time surpassing, the performance reported in each original publication, giving us a set of strong baselines. Finally, we compare our method with standard supervised learning by training purely on labelled data.

Training Strategies. All baselines and our method are implemented on the same segmentation architecture: DeepLabV3+ [CZP+18] with ResNet-101 backbone [HZRS16], trained with the same optimisation strategies, and the same labelled and unlabelled data split.

Results on Pascal VOC, CityScapes and SUN RGB-D (Full Labels)

First, we compare our results to baselines (4 semi-supervised and 1 supervised) in a full-label setting. For semi-supervised learning, we apply ReCo on top of ClassMix, which consistently outperforms other semi-supervised baselines. In fully supervised learning, we simply apply ReCo on top of standard supervised learning.

Table 4.1 shows the mean IoU validation performance in three datasets over three individual runs (different labelled and unlabelled data split). We see that for all cases, applying the ReCo loss improves performance in both the semi-supervised and supervised settings. In the fewest label settings in each dataset, applying ReCo with ClassMix can improve results by an especially significant margin, with up to 5 – 10% relative improvement.

We present qualitative results from the semi-supervised setup with the fewest labels: 20

	Pascal VOC				CityScapes				SUN RGB-D			
Method	60 im.	200 im.	600 im.	all im.	20 im.	50 im.	150 im.	all im.	50 im.	150 im.	500 im.	all im.
Supervised	37.79	53.87	64.04	77.79	38.12	45.42	54.93	70.48	19.79	28.78	37.73	51.06
S4GAN	47.95	61.25	66.21	-	37.65	47.08	56.46	-	20.53	29.79	38.08	-
CutOut	52.96	63.57	69.85	-	42.52	50.15	59.42	-	25.94	34.45	41.25	-
CutMix	**53.71**	66.95	72.42	-	44.02	54.72	62.24	-	27.60	37.55	42.69	-
ClassMix	49.06	67.95	72.50	-	45.61	55.56	63.94	-	28.42	37.55	42.46	-
ReCo	53.31	**69.81**	**72.75**	**78.39**	**49.86**	**57.69**	**65.04**	**71.45**	**29.65**	**39.14**	**44.55**	**52.01**

Table 4.1. mean IoU validation performance for Pascal VOC, CityScapes, and SUN RGB-D datasets. We report the performance averaged over three independent runs for all methods. The number of labelled images shown in the first three columns in each dataset is chosen to make sure the least appeared classes have appeared in 5, 15, and 50 images respectively.

labelled CityScapes and 50 labelled SUN RGB-D datasets in Fig. 4.5, and 60 labelled Pascal VOC in Fig. 4.3. In all cases, we can see the clear advantage of ReCo, where the edges and boundaries of small objects are clearly more pronounced such as in the "person" and "bicycle" classes in CityScapes, "boat" and "horse" classes in Pascal VOC, and the "lamp" and "pillow" classes in SUN RGB-D. More interestingly, we find that in SUN RGB-D, though all methods may confuse ambiguous class pairs such as "table" and "desk" or "window" and "curtain", ReCo still produces consistently sharp and accurate object boundaries compared to the Supervised and ClassMix baselines where labels are noisy near object boundaries.

To further justify the effectiveness of ReCo, we also include results on existing benchmarks in Table 4.2. Here, all baselines are re-implemented and reported in the PseudoSeg setting [ZZZ+21], where the labelled images are sampled from the original PASCAL dataset, with a total of 1.4k images. In both benchmarks, ReCo shows state-of-the-art performance, and specifically can reach PseudoSeg's performance, while *requires only half the labelled data*.

Pascal VOC	1/16 [92]	1/8 [183]	1/4 [366]	1/2 [732]
AdvSemSeg [HTL+19]	39.69	47.58	59.97	65.27
Mean Teacher [TV17]	48.70	55.81	63.01	69.16
CCT [OHT20]	33.10	47.60	58.80	62.10
GCT [KQL+20]	46.04	54.98	64.71	70.67
VAT [MMKI18]	36.92	49.35	56.88	63.34
CutMix [FAL+20]	55.58	63.20	68.36	69.87
PseudoSeg [ZZZ+21]	57.60	65.50	69.14	72.41
ReCo	**64.78**	**72.02**	**73.14**	**74.69**

Table 4.2. mean IoU validation performance for Pascal VOC with data partition and training strategy proposed in PseudoSeg [ZZZ+21]. The percentage and the number of labelled data are listed in the first row. ReCo achieves best performance in all cases, and with significantly less labelled data.

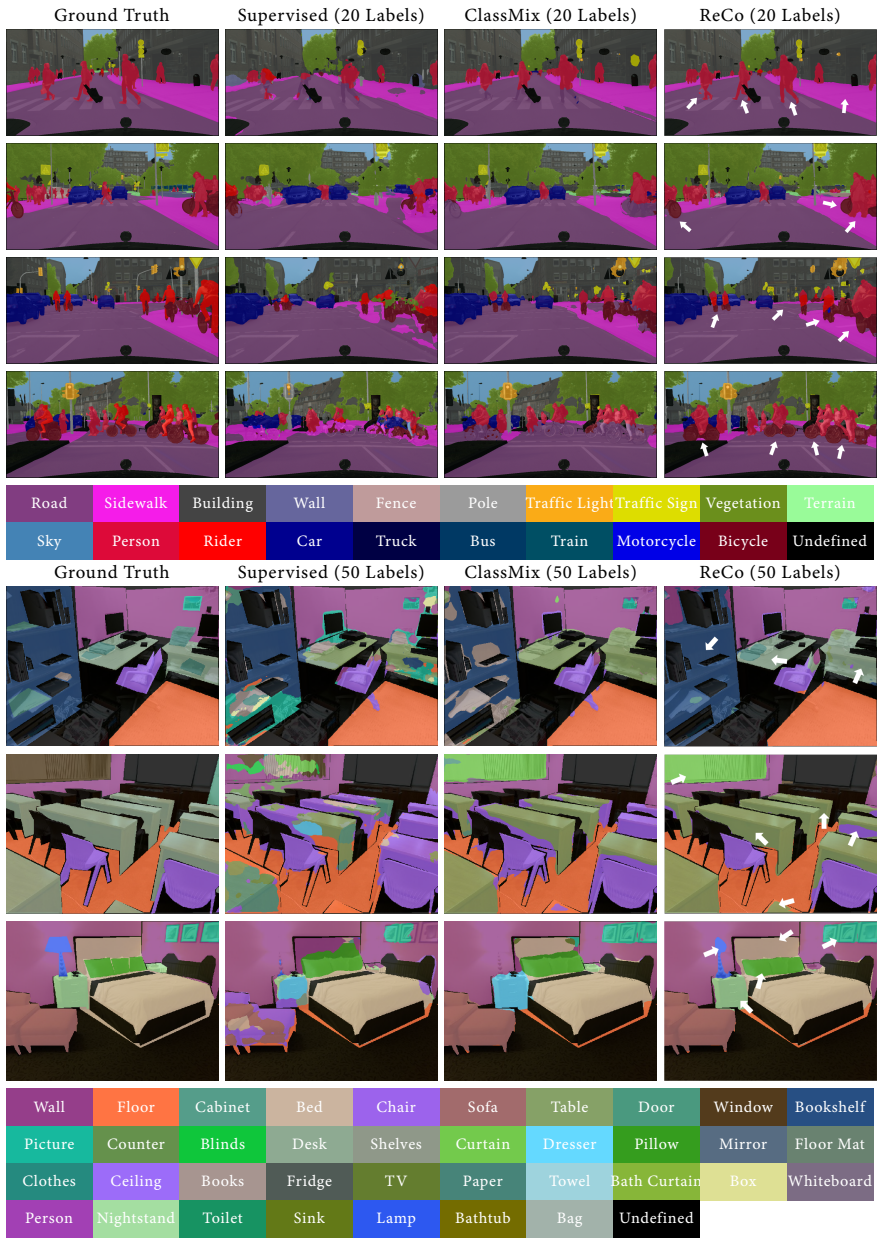


Figure 4.5. Visualisation of Cityscapes (top) and SUN RGB-D (bottom) validation set trained on 20 and 50 labelled images respectively. Interesting regions are shown in white arrows.

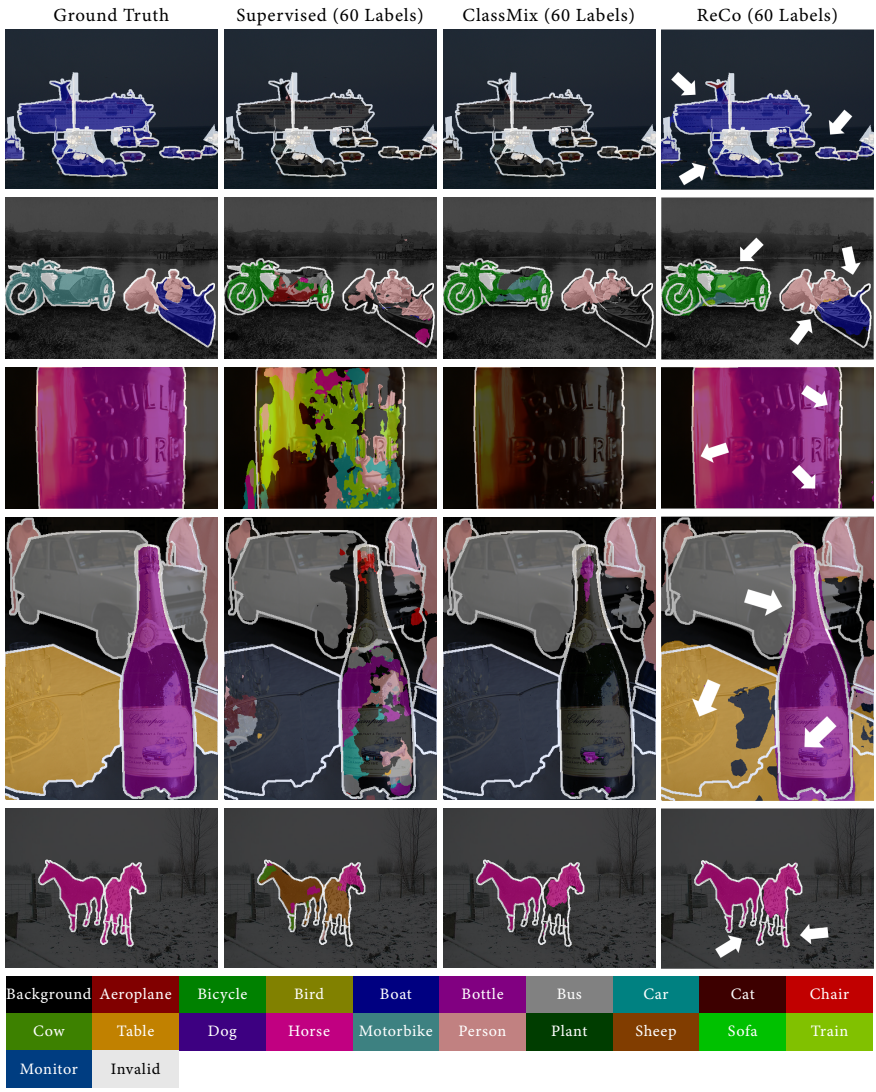


Table 4.3. Visualisation of Pascal VOC validation set trained on 60 labelled images. Interesting regions are shown in white arrows.

Results on Pascal VOC and CityScapes (Partial Labels)

In the partial label setting, we evaluate on the CityScapes and Pascal VOC datasets. We show qualitative results on the Pascal VOC dataset trained on 1 labelled pixel per class per

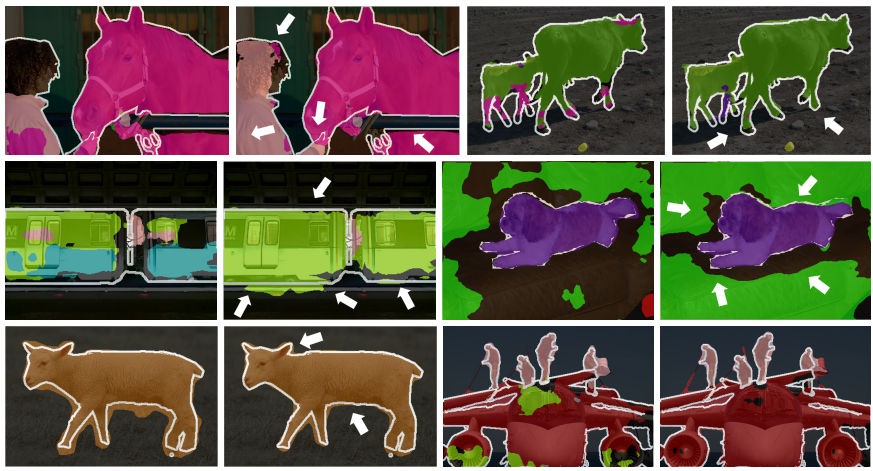


Figure 4.6. Visualisation of Pascal validation set with ClassMix (left) vs. with ReCo (right) trained on 1 labelled pixel per class per image. Interesting regions are shown in white arrows.

image in Fig. 4.6. As in the full label setting, we see smoother and more accurate boundary predictions from ReCo.

Table 4.4 compares ReCo to the two best semi-supervised baselines and a supervised baseline. Once again, we observe that ReCo consistently enhances performance in all cases when applied with ClassMix, resulting in approximately 1 – 5% relative improvement.

	Pascal VOC					CityScapes			
Method	1 pixel	1% labels	5% labels	25% labels	Method	1 pixel	1% labels	5% labels	25% labels
Supervised	60.33	66.17	69.16	73.75	Supervised	44.08	52.89	56.65	63.43
CutMix	63.50	70.83	73.04	75.64	CutMix	46.91	54.90	59.69	65.61
ClassMix	63.69	71.04	72.90	75.79	ClassMix	47.42	56.68	60.96	66.46
ReCo	**66.11**	**72.67**	**74.09**	**75.96**	ReCo	**49.66**	**58.97**	**62.32**	**66.92**

Table 4.4. mean IoU validation performance for Pascal VOC and Cityscapes datasets trained on 1, 1%, 5% and 25% labelled pixels per class per image. We report the performance averaged over three independent runs for all methods.

However, it is worth noting that the extent of performance improvement is somewhat less pronounced than in the full-label setting. This disparity can be attributed to the inherent challenge of very limited ground-truth annotations in this scenario. In such cases, ReCo may occasionally receive inaccurate supervision signals, potentially leading to

confusion in the learning process. This limitation highlights the crucial role of accurate object boundaries. Nonetheless, the fact that ReCo still manages to deliver improvements in this challenging evaluation setting underscores its effectiveness in semi-supervised segmentation, even under highly constrained conditions.

4.5 Ablation Study on Hyper-Parameters and Training Details

Next, we present an ablation study on 20 labelled CityScapes dataset to understand the behaviour of ReCo concerning different hyper-parameters. We use our default experimental setting from Section 4.4, using ReCo with ClassMix. All results are averaged over three independent runs.

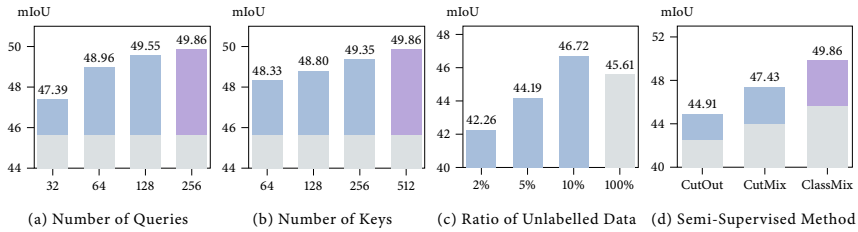


Figure 4.7. mean IoU validation performance on 20 labelled CityScapes dataset based on different choices of hyper-parameters. Grey: ClassMix (if not labelled otherwise) in our default setting. Light Blue: ReCo + ClassMix (if not labelled otherwise) in a different hyper-parameter setting. Purple: ReCo + ClassMix in our default setting.

Number of Queries and Keys. We first evaluate the performance by varying different number of queries and keys used in the ReCo framework, while fixing all other hyper-parameters in default. In Fig. 4.7a and 4.7b, we can observe that performance is better when sampling more queries and keys, but after a certain point, the improvements would become marginal. Notably, even in our smallest option having 32 queries per class in a mini-batch — consisting only less than 0.5% among all available pixel space, can still improve performance in a non-trivial margin. Compared to a concurrent work [ZTRR21] which requires 10k queries and 40k keys in each training iteration, ReCo can be optimised with $\times 50$ more efficiency in terms of memory footprint.

Ratio of Unlabelled Data. We examine how effectively ReCo can generalise across varying ratios of unlabelled data. As depicted in Fig. 4.7c, we demonstrate that ReCo outperforms the ClassMix baseline, even with only 10% of the original amount of unlabelled data. This observation underscores the remarkable capacity of ReCo not only to achieve impressive gains in label efficiency but also to excel in data efficiency.

Choice of Semi-Supervised Method. Finally, we demonstrate the robustness of ReCo across different semi-supervised methods. As illustrated in Fig. 4.7d, we observe that ReCo consistently achieves higher performance across a range of semi-supervised baselines, with similar relative improvements.

Effect of Active Sampling. Table 4.5 reveals that when queries and keys are randomly sampled without active sampling, the performance improvement is notably lower compared to the active sampling approach used in our default setting. Additionally, the active sampling strategy that focuses on hard queries has a pronounced impact on generalisation. In contrast, if we were to sample solely from easy queries, ReCo only yields marginal improvements over the baseline. This observation reinforces that the strategic sampling of queries and keys is a crucial component of the ReCo framework, demonstrating its pivotal role in achieving superior results in semi-supervised semantic segmentation.

Random Query Random Key	Active Query Random Key	Easy Query Active Key	Baseline	Our Setting
46.56	46.38	45.81	45.61	49.86

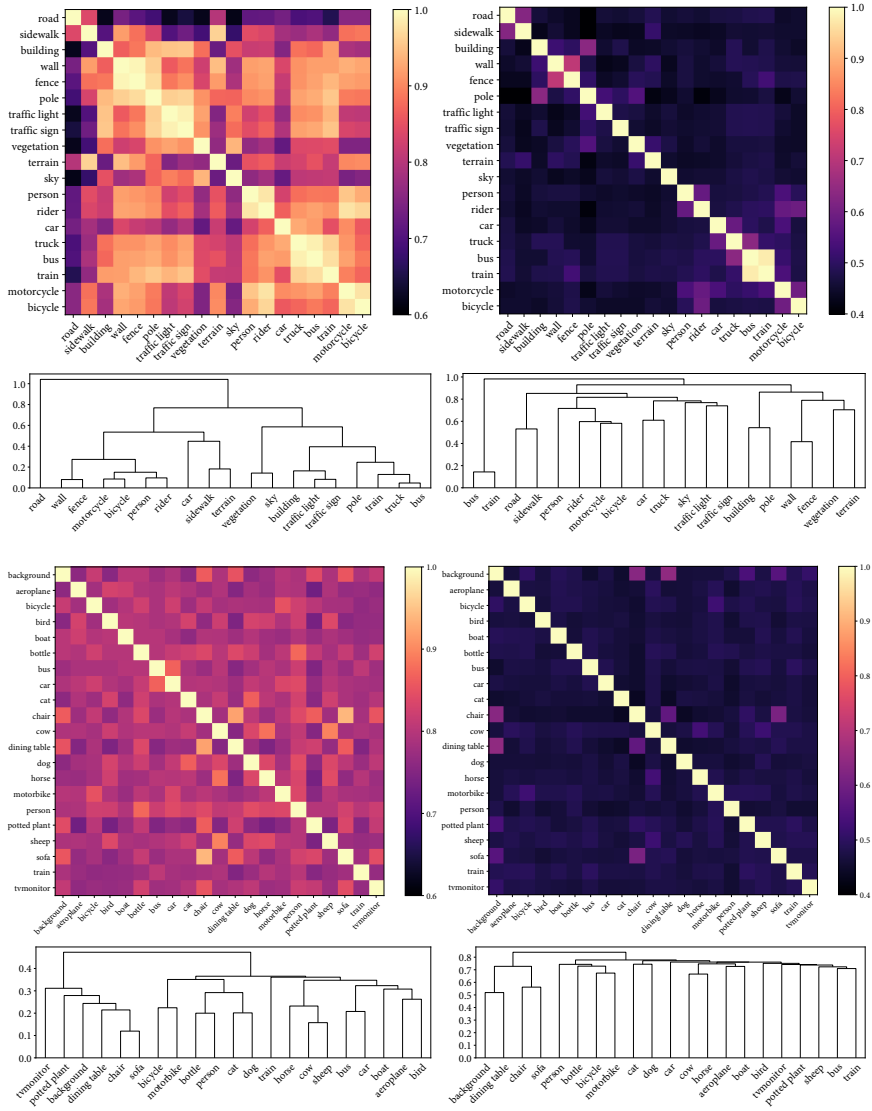
Table 4.5. mean IoU validation performance on 20 labelled CityScapes dataset based on different query and key sampling strategies. Active key and query sampling offer a significant improvement over random sampling.

Compared to Feature Bank Methods. We also test ReCo with a stored feature bank, which is similar to the design employed in concurrent works [ASF+21, WZY+21]. We found that replacing our batch-wise sampling with a feature bank sampling will achieve a similar performance (49.34 mIoU) compared to our original design (49.86 mIoU) on 20 labelled CityScapes, but with a slower training speed. This confirms that our batch-wise sampling accurately approximates class distribution across the dataset, making it an efficient choice.

4.6 Visualisations and Interpretability of Class Relationships

In this section, we present visualisations of the pair-wise semantic class relation graph defined in Eq. 4.4. We further enhance these visualisations with a semantic class dendrogram using the off-the-shelf hierarchical clustering algorithm available in SciPy [VGO+20]. These visualisations aim to provide a more comprehensive understanding of the semantic class relationships in the learned representations.

For both visualisations, we compute the features for each semantic class by averaging the pixel embeddings from the validation set. In all the visualisations, we compare the features



(a) Embedding $\mathcal{Z}$ (Supervised)

(b) Embedding $\mathcal{R}$ (ReCo + Supervised)

Figure 4.8. Visualisation of semantic class relation graph and its corresponding semantic class dendrogram on CityScapes (top) and PASCAL VOC (bottom) datasets. A brighter colour represents a closer (more confused) relationship. Best viewed in zoom.

learned with ReCo on top of supervised learning with a standard supervised learning method trained on all labelled data. This comparison helps us gain insights into how ReCo affects the semantic class relationships in the learned representations when compared to traditional supervised learning on the full dataset.

Utilising the same definitions as in Section 4.3, we present these visualisations for two types of embeddings: embedding $\mathcal{Z}$, which is the embedding predicted from the encoder network f_ϕ^e and used for pixel-level classification in supervised method, and embedding $\mathcal{R}$, which is the actual representation utilised for ReCo loss and active sampling.

In Fig. 4.8, we showcase the semantic class relationships and dendrograms for the CityScapes and PASCAL VOC dataset based on embeddings $\mathcal{Z}$ and $\mathcal{R}$, with and without ReCo loss. The visualisations reveal that ReCo significantly aids in disentangling features compared to standard supervised learning, where many pairs of semantic classes exhibit high similarity. Additionally, nearly all classes based on embedding $\mathcal{R}$ are perfectly disentangled, except for "bus" and "train", suggesting that the CityScapes dataset might lack sufficient examples of these two classes to learn distinctive representations for them.

The pair-wise relation graph and dendrogram visualisations offer valuable insights into the distribution of semantic classes within each dataset and help clarify the patterns of incorrect predictions made by the trained semantic network. Additionally, we provide a dendrogram based on embedding $\mathcal{R}$ for the SUN RGB-D dataset, which highlights ambiguous class pairs, such as "night stand" and "dresser", "table" and "desk", and "floor" and "floormat", aligning with the results presented in Fig. 4.5. These visualisations serve as a useful tool for understanding the relationships between semantic classes and the impact of ReCo on feature disentanglement and class separation.

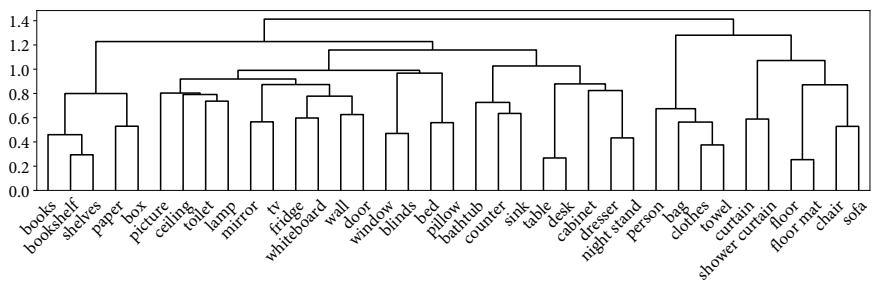


Figure 4.9. Visualisation of semantic class dendrogram based on embedding $\mathcal{R}$ on SUN RGB-D dataset using ReCo + Supervised method. Best viewed in zoom.

4.7 Conclusion, Limitations and Discussions

In this chapter, we have presented ReCo, a new pixel-level contrastive framework with active sampling, designed specifically for semantic segmentation. ReCo explores structured semantic class relationships, and can improve performance in supervised or semi-supervised semantic segmentation methods with minimal additional memory footprint. In particular, ReCo has shown its strongest effect in semi-supervised learning with very few labels, where we improved on the previous state-of-the-art by a large margin.

We now discuss some limitations of ReCo that we have noted during our implementations, and we discuss our thoughts on future directions with this work.

Constraive Learning for Video Object Segmentation We are convinced that ReCo holds potential for extension to other tasks beyond semantic segmentation, with a special focus on video object segmentation. To achieve this, we can implement ReCo in the temporal dimension, allowing the sampling of queries and keys from different sequences. We believe the exploration of contrastive learning frameworks can be a promising and interesting direction for improving data-efficient video object segmentation.

ReCo for Interactive Semantic Segmentation Additionally, we see the opportunity to apply ReCo to the task of interactive semantic segmentation. In this context, ReCo can be employed in collaboration with human experts to identify the most informative pixels for annotation, thereby alleviating the annotation workload for human annotators. We envision that this application of ReCo can offer significant benefits, especially in domains like medical imaging, where the annotation of medical images often proves to be time-consuming and resource-intensive.

Impact on Future Research The ReCo framework has demonstrated its efficacy and efficiency in improving semantic segmentation performance across diverse settings. It has been adapted and iteratively refined to enhance active learning [RKH22], probabilistic representation learning [XWZ+23], information transfer [WFL+24], and pseudo-label generation [WWS+22]. Moreover, its application has yielded performance enhancements in numerous domains, including remote sensing [YYD+23] and medical imaging [YDM+24].

5

Self-Supervised Generalisation with Meta Auxiliary Learning

In the previous chapter, we explored an auxiliary learning strategy of using the inherent structure of semantics to improve semantic segmentation tasks. In this chapter, we propose an alternative approach, wherein we directly generate a semantic structure for auxiliary labels, and leverage them to improve the performance of the primary task. We call this approach Meta Auxiliary Learning (MAXL). We demonstrate that MAXL can improve single-task learning on a range of image datasets, all while operating without the need for extra data. Furthermore, our results indicate that MAXL outperforms other baseline methods for generating auxiliary labels and even competes favourably with human-defined auxiliary labels. The self-supervised aspect of MAXL introduces a promising avenue for automated generalisation in machine learning.

5.1 Understanding Auxiliary Learning with Semantic Complexity

We first take a closer look at the generalisation of auxiliary learning, seeking to gain insights into how the generalisation performance over the primary task behaves when we train alongside different numbers and designs of auxiliary tasks. For simplicity, we evaluate and perform all experiments in image classification tasks in a single-domain setting.

3 Class	10 Class	20 Class	100 Class
animals	large animals	reptiles	crocodile, dinosaur, lizard, snake, turtle
		large carnivores	bear, leopard, lion, tiger, wolf
		large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
	medium animals	aquatic mammals	beaver, dolphin, otter, seal, whale
		medium-sized mammals	fox, porcupine, possum, raccoon, skunk
	small animals	small mammals	hamster, mouse, rabbit, shrew, squirrel
		fish	aquarium fish, flatfish, ray, shark, trout
	invertebrates	insects	bee, beetle, butterfly, caterpillar, cockroach
		non-insect invertebrates	crab, lobster, snail, spider, worm
	people	people	baby, boy, girl, man, woman
vegetations	vegetations	flowers	orchids, poppies, roses, sunflowers, tulips
		fruit and vegetables	apples, mushrooms, oranges, pears, peppers
		trees	maple, oak, palm, pine, willow
objects and scenes	household objects	food containers	bottles, bowls, cans, cups, plates
		household electrical devices	clock, keyboard, lamp, telephone, television
		household furniture	bed, chair, couch, table, wardrobe
	construction	large man-made outdoor things	bridge, castle, house, road, skyscraper
	natural scenes	large natural outdoor scenes	cloud, forest, mountain, plain, sea
	vechicles	vehicles 1	bicycle, bus, motorcycle, pickup truck, train
		vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

Table 5.1. A 4-level hierarchy for multi-label image classification task based on CIFAR-100 dataset. This extended dataset introduces additional complexity to the original CIFAR-100 by incorporating two new levels of coarser labels. We define a higher complexity level for labels associated with a larger number of classes, and a lower complexity level for labels associated with fewer classes.

Training Setup. Specifically, we employ a multi-task network to predict a set of training tasks, where each training task is designed as one level of hierarchical label from a pre-defined multi-label image classification dataset. In all training tasks, one task is considered as the primary task, and all the remaining tasks are considered as auxiliary tasks. Our goal is to understand how generalisation over the primary task is impacted by the presence of auxiliary tasks, which exhibit varying degrees of semantic complexity. In this context, we measure semantic complexity in each task directly by the number of classes defined in the label, *i.e.* the finer classification label which describes with more detailed information gives a higher complexity.

In the creation of this multi-label dataset, we construct a four-level hierarchy, building upon the original CIFAR-100 dataset [Kri09]. The structure of this hierarchy is thoughtfully defined and detailed in Table 5.1. To ensure the consistency and robustness of learning performance across various learning methods and network architectures, we conduct experiments with two well-known image classification architectures: VGG-16 [SZ15] and

PRI [3]			
No AUX	AUX [10]	AUX [20]	AUX [100]
92.92	93.94$_{+1.02}$	94.56$_{+1.64}$	94.16$_{+1.24}$

(a) with Higher Auxiliary Task Complexity

PRI [100]			
No AUX	AUX [3]	AUX [10]	AUX [20]
69.84	68.45$_{-1.39}$	69.28$_{-0.56}$	68.95$_{-0.89}$

(b) with Lower Auxiliary Task Complexity

PRI [10]		PRI [20]	
No AUX	AUX [100]	No AUX	AUX [100]
82.52	84.36$_{+1.84}$	79.28	80.37$_{+1.09}$

(c) with Single Auxiliary Task

PRI [3]		
AUX [10+20]	AUX [20+100]	AUX [10+20+100]
94.51$_{+1.59}$	94.54$_{+1.62}$	94.55$_{+1.63}$

(d) with Multiple Auxiliary Tasks

Table 5.2. Test performance of the primary task trained with various designs and numbers of auxiliary tasks. The primary and auxiliary tasks are denoted as PRI [·] and AUX [·], respectively, with the number within the square brackets indicating the number of classes within each label. Additionally, the subscript indicates the extent of performance enhancement observed in the primary task when paired with the corresponding auxiliary task.

ResNet-50 [HZRS16]. We train these networks both with and without regularisation, employing the vanilla hard parameter-sharing approach.

It's noteworthy that our observations remain consistent across these different settings. As a result, we present the test performance achieved with the VGG-16 architecture equipped with regularisation in Table 5.2. This setting provides a representative view of the generalisation behaviour and performance outcomes in our experiments.

Results. In our analysis, we have made several intriguing observations regarding the performance of primary and auxiliary tasks under different conditions. These observations are summarised as follows.

Firstly, we have noticed that when pairing primary tasks with auxiliary tasks containing 3/10/20 classes, there is a significant improvement in the performance of the primary task. Conversely, when the primary task has 100 classes, its performance experiences a certain degree of decline when paired with auxiliary tasks having 3/10/20 classes, compared to single-task training. This indicates that the performance improvement in the primary task *correlates positively with the complexity of the auxiliary class*. This observation is also aligned with the learning strategy we explored in Auto-λ, where we found the part segmentation task with higher complexity can improve the performance of the semantic segmentation task more effectively. However, this improvement rate eventually plateaus and decreases when the auxiliary class complexity becomes excessive. This suggests that there exists an *optimal balance in auxiliary class complexity* for maximising the benefit to the primary task.

Finally, our observation pertains to the influence of auxiliary tasks on the primary task. We have found that the performance of the primary task depends *solely* on the single auxiliary task that provides the best performance improvement. In other words, the primary task's performance is not influenced by the inclusion of multiple auxiliary tasks. This observation underscores the importance of selecting the most beneficial auxiliary task, as it has a more pronounced impact on the primary task's performance compared to the cumulative effect of multiple auxiliary tasks.

These observations shed light on the intricate dynamics of auxiliary learning and provide valuable insights into optimising the performance of primary tasks in multi-task training.

5.2 Related Work

Multi-task & Transfer Learning The aim of multi-task learning (MTL) is to achieve shared representations by simultaneously training a set of related learning tasks. In this case, the learned knowledge used to share across domains is encoded into the feature representations to improve performance of each individual task, since knowledge distilled from related tasks are interdependent. The success of deep neural networks has led to some recent methods advancing the multi-task architecture design, such as applying a linear combination of task-specific features [MSGH16, Kok17]. [LJD19] applied soft-attention modules as feature selectors, allowing learning of both task-shared and task-specific features in an end-to-end manner. Transfer learning is another common approach to improve generalisation, by incorporating knowledge learned from one or more related domains. Pre-training a model with a large-scale dataset such as ImageNet [DDS+09] has become a standard practise in many vision-based applications. Please refer to Sec. 2.2 and 2.3 for a detailed review.

Auxiliary Learning Whilst in multi-task learning the goal is high test accuracy across all tasks, auxiliary learning differs in that high test accuracy is only required for a single primary task, and the role of the auxiliary tasks is to assist in generalisation of this primary task. Applying related learning tasks is one straightforward approach to assist primary tasks. [TTLL17] applied auxiliary supervision with phoneme recognition at intermediate low-level representations to improve the performance of conversational speech recognition. [LK18] chose auxiliary tasks which can be obtained with low effort, such as global descriptions of a scene, to boost the performance for single scene depth estimation and semantic segmentation. By carefully choosing a pair of learning tasks, we may also perform auxiliary learning without ground truth labels, in an unsupervised manner. [JMC+17] introduced a method for improving agent learning in Atari games, by building unsupervised

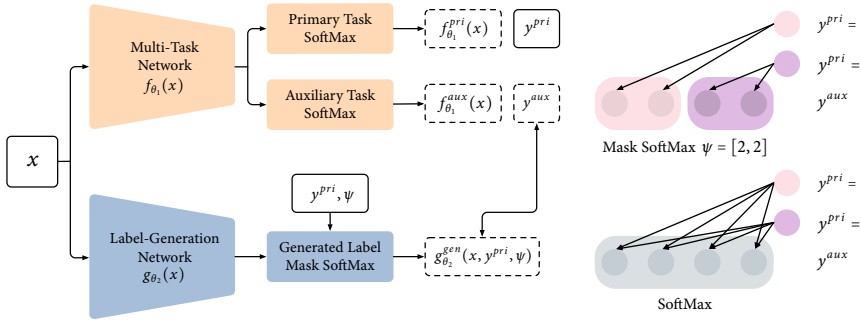
auxiliary tasks to predict the onset of immediate rewards from a short historical context. [FNPS16, ZBSL17] proposed image synthesis networks to perform unsupervised monocular depth estimation by predicting the relative pose of multiple cameras. [DCJ+18] proposed to use cosine similarity as an adaptive task weighting to determine when a defined auxiliary task is useful. Differing from these works which require prior knowledge to manually define suitable auxiliary tasks, our proposed method requires no additional task knowledge, since it generates useful auxiliary knowledge in a purely unsupervised fashion. The most similar work to ours is [ZTJ18], in which meta learning was used in auxiliary data selection. However, this still requires manually-labelled data from which these selections are made, whilst our method is able to generate auxiliary data from scratch.

Meta Learning Meta learning (or learning to learn) aims to induce the learning algorithm itself. Early works in meta learning explored automatically learning update rules for neural models [BBC90, BBCG92, Sch92]. Recent approaches have focussed on learning optimisers for deep networks based on LSTMs [RL16] or synthetic gradients [ADG+16, JCO+17]. Meta learning has also been studied for finding optimal hyperparameters [LZCL17] and a good initialisation for few-shot learning [FAL17]. [SBB+16] also investigated few shot learning via an external memory module. [VBL+16, SSZ17] realised few shot learning in the instance space via a differentiable nearest-neighbour approach. Related to meta learning, our framework is designed to learn to generate useful auxiliary labels, which themselves are used in another learning procedure.

5.3 MAXL: Self-Supervised Auxiliary Learning for Image Classification

We now introduce our method for automatically generating optimal labels for an auxiliary task, as a form of self-supervised auxiliary learning, which we call Meta Auxiliary Learning (MAXL). Based on these insights we have explored in the previous section, we only consider finding a single auxiliary task, although our method could be modified to include several auxiliary tasks. And we only focus on classification tasks for both the primary and auxiliary tasks, but the overall framework could also be extended to regression. As such, the auxiliary task is defined as a sub-class labelling problem, where each primary class is associated with some auxiliary classes, in a two-level hierarchy.

Problem Setup. The goal of MAXL is to generate labels for the auxiliary task which, when trained alongside a primary task, improve the performance of the primary task. To accomplish this, we train two networks: a *multi-task network*, which trains on the primary and



(a) two networks applied in MAXL (b) SoftMax v.s. Mask SoftMax

Figure 5.1. MAXL framework overview. (a) Illustration of the two networks which make up MAXL. Dashed white boxes represent data generated by neural networks, solid white boxes represent given data, and coloured boxes represent functions. The double arrow represents equivalence. (b) Illustration of vanilla SoftMax and Mask SoftMax with 2 primary classes. Vanilla SoftMax outputs over all 4 auxiliary classes, whereas Mask SoftMax outputs over a hierarchical structure $\psi = [2, 2]$.

auxiliary task in a standard multi-task learning setting, and a *label-generation network*, which generates the labels for the auxiliary task.

We denote the multi-task network as a function $f_{\theta_1}(x)$ with parameters θ_1 which takes an input x, and the label-generation network as a function $g_{\theta_2}(x)$ with parameters θ_2 which takes the same input x. Parameters θ_1 are updated by losses of both the primary and auxiliary tasks, as is standard in auxiliary learning. However, parameters θ_2 are updated only by the performance of the primary task.

In the multi-task network, we apply the vanilla hard parameter sharing approach, in which we predict both the primary and auxiliary classes using the shared features θ_1. At the final feature layer, $f_{\theta_1}(x)$, we then further apply task-specific layers to output the corresponding prediction for each task, using a SoftMax function. We denote the primary task predictions by $f^{pri}_{\theta_1}(x)$, and the auxiliary task predictions by $f^{aux}_{\theta_1}(x)$. And we denote the ground-truth primary task labels by y^{pri}, and the generated auxiliary task labels by y^{aux}.

We found during experiments that training benefited from assigning each primary class its own unique set of possible auxiliary classes, rather than sharing all auxiliary classes across all primary classes. In the label-generation network, we therefore define a hierarchical structure ψ, which determines the number of auxiliary classes for each primary class. At the output layer of the label-generation network, we then apply a masked SoftMax function to ensure that each output node represents an auxiliary class corresponding to

only one primary class, as described further in the later section. Given input data x, the label-generation network then takes in a fixed hierarchy ψ together with the ground-truth primary task label y^{pri}, and applies Mask SoftMax to predict the auxiliary labels, denoted by $y^{aux} = g^{gen}_{\theta_2}(x, y^{pri}, \psi)$. A visualisation of the overall MAXL framework is shown in Fig. 5.1. Note that we allow soft assignment for the generated auxiliary labels, rather than one-hot encoding, which we found during experiments enables greater flexibility to obtain optimal auxiliary labels.

Model Objectives. The multi-task network is trained alongside the label-generation network, with two stages per epoch. In the first stage, the multi-task network is trained using primary task ground-truth labels, and the auxiliary labels from the label-generation network. In the second stage, the label-generation network is updated by computing its gradients with respect to the multi-task network's prediction accuracy on the primary task. We train both networks in an iterative manner until convergence.

In the first stage of each epoch, given target auxiliary labels as determined by the label-generation network, the multi-task network is trained to predict these labels for the auxiliary task, alongside the ground-truth labels for the primary task. For both the primary and auxiliary tasks, we apply the focal loss [LGG+17] with a focusing parameter $\gamma = 2$:

$$\mathcal{L}(\hat{y}, y) = -y(1-\hat{y})^{\gamma}\log(\hat{y}), \tag{5.1}$$

where $\hat{y}$ is the predicted label and y is the target label. The focal loss helps to focus on the incorrectly predicted labels, which we found improved performance during our experimental evaluation compared with the regular cross-entropy log loss.

To update θ_1 of the multi-task network, we define the multi-task objective as follows:

$$\arg\min_{\theta_1} \left(\mathcal{L}(f^{pri}_{\theta_1}(x_{(i)}), y^{pri}_{(i)}) + \mathcal{L}(f^{aux}_{\theta_1}(x_{(i)}), y^{aux}_{(i)}) \right) \tag{5.2}$$

where (i) represents the i^{th} batch from the training data, and $y^{aux}_{(i)} = g^{gen}_{\theta_2}(x_{(i)}, y^{pri}_{(i)}, \psi)$ is generated by the label-generation network.

In the second stage of each epoch, the label-generation network is then updated by encouraging auxiliary labels to be chosen such that, if the multi-task network were to be trained using these auxiliary labels, the performance of the primary task would be maximised on this same training data. Leveraging the performance of the multi-task network to train the label-generation network can be considered as a form of meta learning. Therefore, to

update parameters θ_2 of the label-generation network, we define the meta objective as follows:

$$\arg\min_{\theta_2} \mathcal{L}\left(f^{pri}_{\theta_1^+}(x_{(i)}), y^{pri}_{(i)}\right). \tag{5.3}$$

Here, θ_1^+ represents the weights of the multi-task network after one gradient update using the multi-task loss defined in Equation 5.2:

$$\theta_1^+ = \theta_1 - \alpha \nabla_{\theta_1}\left(\mathcal{L}(f^{pri}_{\theta_1}(x_{(i)}), y^{pri}_{(i)}) + \mathcal{L}(f^{aux}_{\theta_1}(x_{(i)}), y^{aux}_{(i)})\right), \tag{5.4}$$

where α is the learning rate.

The trick in this meta objective is that we perform a derivative over a derivative (a Hessian matrix) to update θ_2, by using a retained computational graph of θ_1^+ in order to compute derivatives with respect to θ_2. This second derivative trick was also proposed in several other meta-learning frameworks such as [ME14] and [LSY19].[1]

However, we found that the generated auxiliary labels can easily collapse, such that the label-generation network always generates the same auxiliary label. This leaves parameters θ_2 in a local minimum without producing any extra useful knowledge. Therefore, to encourage the network to learn more complex and informative auxiliary tasks, we further apply an entropy loss $\mathcal{H}(y^{aux})$ as a regularisation term in the meta objective on all auxilairy classes. A detailed explanation of the entropy loss and the collapsing label problem is presented below. Finally, we update MAXL's label generation network by

$$\theta_2 \leftarrow \theta_2 - \beta \nabla_{\theta_2}\left(\mathcal{L}(f^{pri}_{\theta_1^+}(x_{(i)}), y^{pri}_{(i)}) + \lambda \cdot \mathcal{H}(y^{aux}_{(i)})\right). \tag{5.5}$$

Mask SoftMax for Hierarchical Predictions

As previously discussed, we include a hierarchy ψ which defines the number of auxiliary classes per primary class. To implement this, we design a modified SoftMax function, which we call Mask SoftMax, to predict auxiliary labels only for certain auxiliary classes. This takes ground-truth primary task label y, and the hierarchy ψ, and creates a binary mask $M = \mathcal{B}(y, \psi)$. The mask is zero everywhere, except for ones across the set of auxiliary classes associated with y. For example, consider a primary task with 2 classes $y = 0, 1$, and a hierarchy of $\psi = [2, 2]$ as in Figure 5.1b. In this case, the binary masks are $M = [1, 1, 0, 0]$ for $y = 0$, and $[0, 0, 1, 1]$ for $y = 1$.

[1] The finite approximation used in Auto-λ can also be applied here to speed up training.

Applying this mask element-wise to the standard SoftMax function then allows the label-prediction network to assign auxiliary labels only to relevant auxiliary classes:

$$\text{SoftMax:} \quad p(\hat{y}_i) = \frac{\exp \hat{y}_i}{\sum_i \exp \hat{y}_i}, \qquad \text{Mask SoftMax:} \quad p(\hat{y}_i) = \frac{\exp M \odot \hat{y}_i}{\sum_i \exp M \odot \hat{y}_i}, \tag{5.6}$$

where $p(\hat{y}_i)$ represents the probability of the generated auxiliary label $\hat{y}$ over class i, and $\odot$ represents element-wise multiplication. Note that no domain knowledge is required to define the hierarchy, and MAXL performs well across a range of values for ψ.

Finally, the complete MAXL framework is defined as follows:

Initialise: network parameters: θ_1, θ_2; hierarchical structure: ψ
Initialise: learning rate: α, β; entropy weighting: λ
while *not converged* **do**
  for *each training iteration i* **do**
    # fetch one batch of training data
    $(x_{(i)}, y^{pri}_{(i)}) \in (x, y)$
    # auxiliary-training step
    Update: $\theta_1 \leftarrow \theta_1 - \alpha \nabla_{\theta_1} \left(\mathcal{L}(f^{pri}_{\theta_1}(x_{(i)}), y^{pri}_{(i)}) + \mathcal{L}(f^{aux}_{\theta_1}(x_{(i)}), g_{\theta_2}(x_{(i)}, y^{pri}_{(i)}, \psi)) \right)$
  end
  for *each training iteration i* **do**
    # fetch one batch of training data
    $(x_{(i)}, y^{pri}_{(i)}) \in (x, y)$
    # retain training computational graph
    Compute: $\theta_1^+ = \theta_1 - \alpha \nabla_{\theta_1} \left(\mathcal{L}(f^{pri}_{\theta_1}(x_{(i)}), y^{pri}_{(i)}) + \mathcal{L}(f^{aux}_{\theta_1}(x_{(i)}), g_{\theta_2}(x_{(i)}, y^{pri}_{(i)}, \psi)) \right)$
    # meta-training step (second derivative trick)
    Update: $\theta_2 \leftarrow \theta_2 - \beta \nabla_{\theta_2} \left(\mathcal{L}(f^{pri}_{\theta_1^+}(x_{(i)}), y^{pri}_{(i)}) + \lambda \mathcal{H}(y^{aux}_{(i)}) \right)$
  end
end

The Collapsing Class Problem

As previously discussed, we introduce an additional regularisation loss, which we call the entropy loss $\mathcal{H}(\hat{y}_{(i)})$. This encourages high entropy across the auxiliary class prediction space, which in turn encourages the label-prediction network to fully utilise all auxiliary classes. The entropy loss calculates the KL divergence between the predicted auxiliary label space $\hat{y}_{(i)}$, and a uniform distribution, for each i^{th} batch. This is equivalent to calculating the entropy of the predicted label space, and is defined as:

$$\mathcal{H}(\hat{y}_{(i)}) = \sum_{k=1}^{K} \hat{y}^k_{(i)} \log \hat{y}^k_{(i)}, \quad \hat{y}^k_{(i)} = \frac{1}{N} \sum_{n=1}^{N} \hat{y}^k_{(i)}[n]. \tag{5.7}$$

where K is the total number of auxiliary classes, and N is the training batch size.

5.4 Experiments

In this section, we present experimental results to evaluate MAXL with respect to several baselines and datasets on image classification.

Experimental Setup

Datasets. We evaluate on six different datasets, with varying sizes and complexities. One of these, CIFAR-100 [Kri09], being expanded into a 4-level hierarchy was used in Section 5.1. This hierarchy is then used for ground-truth auxiliary labels for the *Human* baseline (see below). For the other five datasets: MNIST [LBBH98], SVHN [SCL12], CIFAR-10 [Kri09], ImageNet [DDS+09] and UCF-101 [SZS12], a hierarchy is either not available or difficult to access, and so no ground-truth auxiliary labels exist. All larger datasets were rescaled to resolution $[32 \times 32]$ to accelerate training.

Baselines. We compare MAXL to a number of baselines. First, we compare with *Single Task*, which trains only with the primary class label and does not employ auxiliary learning. This comparison was done to determine whether MAXL could improve classification performance without needing any extra labelled data. Then, we compare to three baselines for generating auxiliary labels: *Random*, *K-Means*, and *Human*, to evaluate the effectiveness of MAXL's meta-learning for label generation. *Random* assigns each training image to random auxiliary classes in a randomly generated (well-balanced) hierarchy. *K-Means* determines auxiliary labels via unsupervised clustering using K-Means [HW79], performed on the latent representation of an auto-encoder, with clustering updated after every training iteration. *Human* uses the human-defined hierarchy of CIFAR-100, where the auxiliary classes are at a lower (finer-grained) level hierarchy than the primary classes. Note that in order to compare these baselines to *Human*, they were only evaluated on CIFAR-100 which is the only dataset containing a human-defined hierarchy (and hence ground-truth auxiliary labels).

Compared to Single Task Learning

First, we compare MAXL to a single-task learning baseline, to determine whether MAXL can improve recognition accuracy without needing access to any additional data. To test the robustness of MAXL, we evaluate it on 3 different networks: a simple 4-layer ConvNet, VGG-16 [SZ15], and ResNet-32 [HZRS16]. We use hyper-parameter search for all networks and apply regularisation methods in order to achieve optimal performance. Since the power of MAXL lies in its ability to work without domain knowledge, we test MAXL across

a range of hierarchies ψ, to study if it is effective without needing to tune this hierarchy for each dataset. Here, the hierarchies are well balanced such that $\psi[i]$ (representing the number of auxilary classes for i^{th} primary class) is the same for all primary classes.

Table 5.3 shows the test accuracy of MAXL and single-task learning, with each accuracy averaged over three individual runs. We see that MAXL consistently outperforms single-task learning across all five datasets, despite both methods using exactly the same training data. We also see that MAXL outperforms single-task learning across all tested values of ψ, showing the robustness of our method without requiring domain knowledge or a manually-defined hierarchy.

Datasets	Backbone	Single	MAXL, $\psi[i]$ =			
			2	3	5	10
MNIST	4-layer ConvNet	99.57 ± 0.02	99.56 ± 0.04	**99.71 ± 0.02**	99.59 ± 0.03	99.57 ± 0.02
SVHN	4-layer ConvNet	94.05 ± 0.07	94.39 ± 0.08	94.38 ± 0.07	**94.59 ± 0.12**	94.41 ± 0.09
CIFAR-10	VGG-16	92.77 ± 0.13	93.27 ± 0.09	93.47 ± 0.08	**93.49 ± 0.05**	93.10 ± 0.08
ImageNet	VGG-16	46.67 ± 0.12	46.82 ± 0.14	46.97 ± 0.10	**47.02 ± 0.11**	46.85 ± 0.11
UCF-101	ResNet-32	53.15 ± 0.12	54.19 ± 0.18	**55.39 ± 0.16**	54.70 ± 0.12	54.32 ± 0.18

Table 5.3. Comparison of MAXL with single-task learning, across a range of hierarchies. We report results with the range of three individual runs, and the best performance for each dataset is marked with bold. All larger datasets were rescaled to a resolution of $[32 \times 32]$.

Compared to Auxiliary Label Generation Baselines

Next, we compare MAXL to a number of baseline methods for generating auxiliary labels, on CIFAR-100. Here, all the baselines are trained without any regularisation to test the full generalisation ability purely from auxiliary tasks. This dataset has a manually-defined hierarchy, which is used in *Human* for ground-truth auxiliary labels. However, MAXL, *Random*, and *K-Means* do not require any human knowledge or manually-defined hierarchy to generate auxiliary labels. Therefore, as in Section 5.4, a hierarchy ψ is defined, assigning each primary class a set of auxiliary classes. We create well-balanced hierarchies by assigning an equal number of auxiliary classes per primary class, and for cases when the hierarchy is unbalanced by one auxiliary class, we randomly choose which primary classes are assigned each number of auxiliary classes in ψ. We run each experiment three times and average the results.

Test accuracies are presented in Table 5.4, using all possible combinations of the numbers of primary classes and total auxiliary classes in CIFAR-100 (where the auxiliary classes are at a lower level to the primary classes). We observe that MAXL outperforms *Single*

	PRI [3] AUX [10]	PRI [3] AUX [20]	PRI [3] AUX [100]	PRI [10] AUX [20]	PRI [10] AUX [100]	PRI [20] AUX [10]
Single	87.49	87.49	87.49	75.15	75.15	70.71
Random	89.86	89.15	87.81	77.26	75.88	71.11
K-Means	90.16	90.57	90.43	77.68	78.63	73.35
Human	**90.78**	**90.78**	**91.23**	77.97	78.18	73.11
MAXL	90.59	90.68	90.61	**78.64**	**78.43**	**74.28**

Table 5.4. Test accuracy for the multi-level CIFAR-100 dataset, comparing MAXL with baseline methods for generating auxiliary labels. Our version of CIFAR-100 has a four-level hierarchy of 3, 10, 20, 100 classes per level, and we use this to create the hierarchy ψ for auxiliary learning.

Task, *Random*, and *K-Means*. Note that *K-Means* requires significantly longer training time than MAXL due to the need to run clustering after each iteration. Also, note that the superior performance of MAXL over these three occurs despite all four methods using exactly the same data. Finally, we observe that MAXL performs similarly and in some cases better than *Human*, despite this baseline requiring manually-defined auxiliary labels for the entire training dataset. With the performance of MAXL similar to that of a system using human-defined auxiliary labels, we see strong evidence that MAXL is able to learn to generalise effectively in a self-supervised manner.

Analysis on the Collapsing Class Problem

In Table 5.5, we show results on CIFAR-100 trained with and without entropy loss, for all 6 combinations of primary and auxiliary tasks evaluated in Table 5.4. In each of these settings, we provide the test accuracy as well as the percentage of auxiliary labels that are effectively utilised, as determined by the label-generation network.

Notably, we observe that training MAXL with entropy loss effectively leverages the entire auxiliary label space in all tasks, to encourage a more comprehensive exploration when generating the auxiliary task. This utilisation of the auxiliary label space results in performance improvements compared to where entropy loss is not applied. This observation confirms the beneficial impact of entropy loss within the MAXL framework and its role in promoting the effective utilisation of the auxiliary task.

Understanding the Utility of Auxiliary Labels

In Fig. 5.2, we show the cosine similarity measurements of gradients in the shared layers of the multi-task network, trained on all 6 pairs of hierarchies in Table 5.4. We observe that baseline methods *Human* and *Random*, with fixed auxiliary labels, reach their maximal

PRI	AUX	with entropy loss		without entropy loss	
		Label %	Test Acc.	Label %	Test Acc.
3	10	100	90.50	100	90.26
3	20	100	90.65	65	90.39
3	100	100	90.66	35	90.22
10	20	100	78.40	100	77.73
10	100	100	78.46	57	78.20
20	100	100	74.27	61	73.97

Table 5.5. Comparison of test accuracies of 4-level CIFAR-100 dataset trained with and without entropy loss. The results highlight the value of incorporating entropy loss within the MAXL framework, as it not only ensures efficient utilisation of the auxiliary label space but also contributes to enhanced performance across all primary and auxiliary task combinations.

similarity at an early stage during training, which then drops significantly afterwards. *K-Means* produces smooth auxiliary gradients throughout training, but its similarity depends on the number of auxiliary classes. In comparison, MAXL produces auxiliary gradients with high similarity throughout the entire training period, and consistently so across the number of auxiliary classes. While we cannot say what the optimal cosine similarity should be, it is clear that MAXL's auxiliary labels affect primary task performance in a very different way from the other baselines.

Due to MAXL's cosine similarity measurements being greater than zero across the entire training stage, a standard gradient update rule for shared feature space is then guaranteed to converge to a local minima given a small learning rate [DCJ+18].

5.5 Visualisations of Generated Auxiliary Knowledge

In Fig. 5.3, we visualise 2D embeddings of examples from the CIFAR-100 test dataset, on two different hierarchies. The visualisations are computed using t-SNE [VdMH08] on the final feature layer of the multi-task network, and compared across three methods: our MAXL method, the *Human* baseline, and the *Single Task* baseline.

This visualisation shows the separability of primary classes after being trained with the multi-task network. Qualitatively, we see that both MAXL and *Human* show a better separation of the primary classes than with *Single Task*, owing to the generalisation effect of the auxiliary learning. This again shows the effectiveness of MAXL while requiring no additional human knowledge.

We also show examples of images assigned to the same auxiliary class through MAXL's

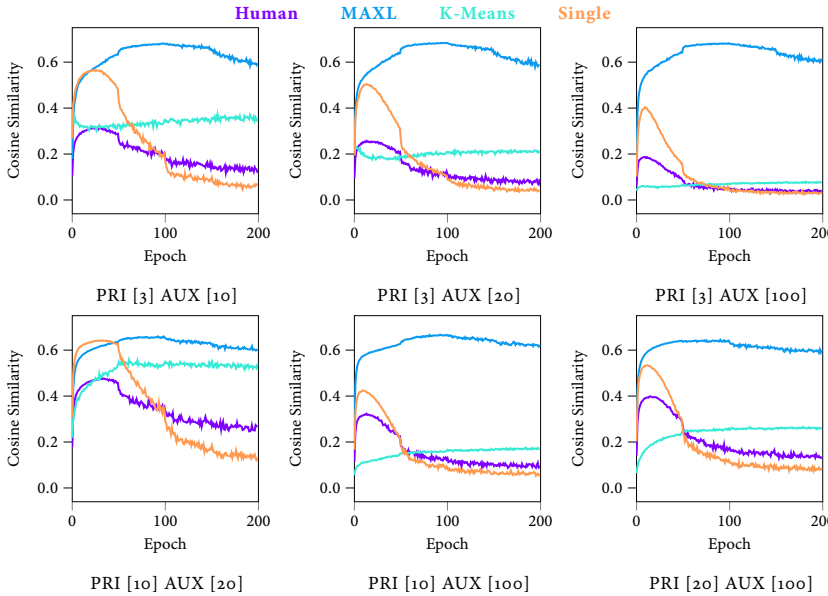
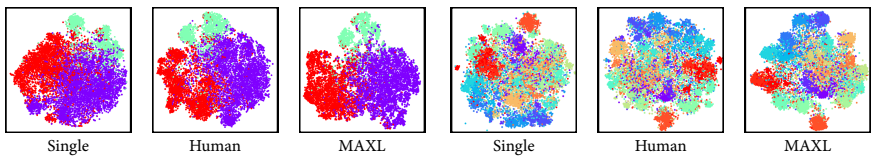


Figure 5.2. Cosine similarity measurement between the auxiliary loss gradient and primary loss gradient, on the shared representation in the multi-task network. We consistently observe high similarity between the auxiliary loss gradients and the primary loss gradients throughout the entire training duration. This consistency underlines the efficacy and reliability of MAXL in maintaining a strong alignment between the learning objectives of auxiliary and primary tasks, regardless of the complexity introduced by varying numbers of auxiliary classes.



(a) PRI [3] AUX [10] (b) PRI [20] AUX [100]

Figure 5.3. t-SNE visualisation of the learned final layer of the multi-task network, trained on CIFAR-100 with two different hierarchies. Both the MAXL and human-defined auxiliary tasks contribute to a more distinct separation of learned feature representations compared to single-task learning. Primary classes are represented by different colours.

label-generation network. Fig. 5.4 shows example images with the highest prediction probabilities for three random auxiliary classes from CIFAR-100, using the hierarchy of 20 primary classes and 100 total auxiliary classes (5 auxiliary classes per primary class), which

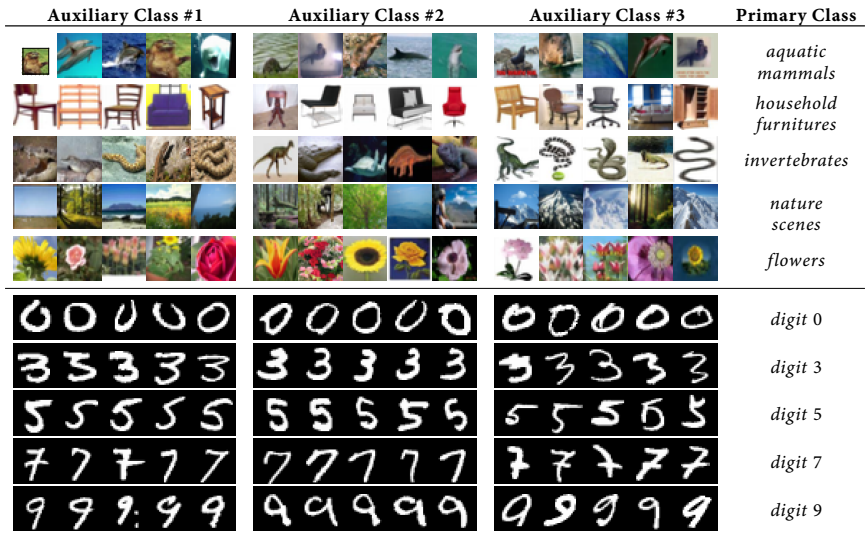


Figure 5.4. Visualisation of 5 test examples with the highest prediction probability, for each of 3 randomly selected auxiliary classes, for different primary classes. We present the visualisation for CIFAR-100 (top) when trained with 20 primary classes and 5 auxiliary classes per primary class, and for MNIST (bottom) when trained with 10 primary classes and 3 auxiliary classes per primary class.

show the best performance of MAXL in Table 5.4. In addition, we also present examples of MNIST, in which 3 auxiliary classes were used for each of the 10 primary classes.

To our initial surprise, only part of the generated auxiliary labels visualised in both datasets show human-understandable knowledge. For example, we can observe that the auxiliary classes #1 and #2 of digit nine are clustered by the direction of the "tail"; auxiliary classes #2 and #3 of digit seven are clustered by the distinction of the "horizontal line". But in most cases, there are no obvious similarities within each auxiliary class in terms of shape, colour, style, structure or semantic meaning, similar to the findings we explored in Auto-λ. However, this makes more sense when we re-consider the role of the label-generation network, which is to assign auxiliary labels that assist the primary task, rather than grouping images in terms of semantic or visual similarity. The label-generation network would therefore be more effective if it were to group images in terms of a shared aspect of reasoning which the primary task is currently struggling to learn.

Further, different from the consistent task relationships found in Auto-λ, we discover that the generated auxiliary knowledge here by MAXL is not deterministic, since the top predicted candidates are different when we re-train the network from scratch. We, therefore,

speculate that using a human-defined hierarchy is just one out of a potentially infinite number of local optima, and on each run of training the label-generation network produces another of these local optima.

5.6 Conclusions, Limitations and Discussions

In this paper, we have presented Meta Auxiliary Learning (MAXL) for generating optimal auxiliary labels which, when trained alongside a primary task in a multi-task setup, improve the performance of the primary task. Rather than employing domain knowledge and human-defined auxiliary tasks as is typically required, MAXL is self-supervised and, combined with its general nature, has the potential to automate the process of generalisation to new levels.

Our evaluation on multiple datasets has shown the performance of MAXL in an image classification setup, where the auxiliary task is to predict sub-class, hierarchical labels for an image. We have shown that MAXL significantly outperforms other baselines for generating auxiliary labels, and is competitive even when human-defined knowledge is used to manually construct the auxiliary labels.

We now discuss some limitations of MAXL that we have noted during our implementations, and we discuss our thoughts on future directions with this work.

MAXL with Multiple Primary Tasks Our current implementation of MAXL primarily focuses on enhancing the performance of a single primary task within a multi-task learning setup. One promising avenue for future research involves extending MAXL to cater to the needs of multiple primary tasks. This extension could involve generating auxiliary labels specific to each primary task, thereby potentially offering an alternative approach to unifying multi-task learning and auxiliary learning to Auto-λ.

Generality of Auxiliary Tasks MAXL's intrinsic flexibility raises intriguing questions about the broader applicability of self-supervised auxiliary learning beyond sub-class image classification. In our exploration, we conducted preliminary experiments aimed at predicting arbitrary vectors, transforming the generated auxiliary task into a regression problem. While the results from these preliminary experiments have yet to yield conclusive findings, they underscore the exciting potential of MAXL in learning versatile auxiliary tasks. The ability of MAXL to adapt and tune these auxiliary tasks automatically for the primary task opens up a promising direction for automated generalisation across a diverse range of more complex tasks.

Impact on Future Research MAXL has introduced a novel paradigm by bridging meta learning and auxiliary learning, thereby exerting a notable influence on a range of machine learning investigations focused on improving representation learning with auxiliary tasks [SNG+23, NAM+21, SLO20, CWG+22, LQZ+22]. Furthermore, MAXL has found broad applicability in facilitating fast test-time online adaptation by leveraging self-supervised or readily accessible auxiliary tasks to improve primary tasks performance across various domains. These domains include a wide array of applications, including human pose estimation [CSL+23], dynamic image deblurring [CWYT21], image denoising [GNP22], future depth prediction [LCY+23], 3D object detection [LXW21], language understanding [GFDZ22], and recommendation systems [LML+23].

6

Vision-Language Reasoning with Multi-Task Experts

We have examined different designs in multi-task and auxiliary learning methods to achieve structured representations in computer vision tasks. In this chapter, we shift our focus to explore how we can leverage multi-task knowledge to improve training efficiency on open-ended vision-language reasoning.

We introduce Prismer, a data- and parameter-efficient vision-language model that leverages an ensemble of task-specific experts. Prismer only requires training of a small number of components, with the majority of network weights inherited from multiple readily-available, pre-trained experts, and kept frozen during training. Unlike other vision-language models that require training huge models on massive datasets, Prismer is a more scalable alternative that can efficiently pool expert knowledge and adapt it to various vision-language reasoning tasks. In our experiments, we show that Prismer achieves fine-tuned and few-shot learning performance which is competitive with current state-of-the-arts, while requiring up to two orders of magnitude less training data.

6.1 Breaking Down Vision-Language Reasoning

Large pre-trained models have demonstrated exceptional generalisation capabilities across a wide range of tasks. However, these capabilities come at a hefty cost in terms of

computational resources required for training and inference, as well as the need for large amounts of training data. In the language domain, models with hundreds of billions of learnable parameters typically require a compute budget on the yottaFLOP scale [CND+22, BMR+20, BBH+22, RBC+21].

The problems in vision-language learning are arguably more challenging. This domain is a strict super-set of language processing, while also requiring extra skills unique to visual and multi-modal reasoning. For example, many image captioning and VQA problems require the model to be capable of fine-grained object recognition, detection, counting, and 3D perception [AAL+15, CFL+15]. A typical solution is to use a massive amount of image-text data to train one giant, monolithic model that learns to develop these task-specific skills from scratch, simultaneously, and within the same generic architecture.

Instead, we investigate an alternative approach to learning these skills and domain knowledge via *distinct and separate sub-networks*, referred to as "experts". As such, each expert can be optimised independently for a specific task, allowing for the use of domain-specific data and architectures that would not be feasible with a single large network. This leads to improved training efficiency, as the model can focus on *integrating* specialised skills and structured domain knowledge, rather than trying to learn everything at once, making it an effective way to *scale down* multi-modal learning.

6.2 Related Work

Vision-Language Models (VLMs) Inspired by the breakthrough of transformers in the language domain [VSP+17, DCLT19], early works aimed to model the vision-language relationship using a shared network based on transformers in a *single-stream* design [li220, CLY+20, LYL+20, SZC+20b]. These works usually leverage a pre-trained object detector, encoding images as sequences of *visual words*, parameterised by object- or region-level features. Prismer takes a different approach by including pre-trained model predictions as auxiliary signals, whilst still relying on the original images to encode visual features.

Another line of works encodes vision and language features in separate networks in a *dual-stream* design, where the vision-only and language-only features are aligned through contrastive learning [RKH+21, ZWM+22, JYX+21, LSG+21]. These works typically focus on close-ended multi-modal alignment tasks such as image-text classification and retrieval. In contrast, Prismer's vision encoder also aligns its vision features with the language embedding through pre-training with contrastive learning, but with a greater emphasis on multi-modal generation tasks.

Both single- and dual-steam VLMs in the past years have often been pre-trained with a combination of multiple objectives, such as masked language modelling, masked region modelling, word-region alignment, visual grounding and more [li220, CLTB21, LLXH22, LSG+21, LBPL19]. These multiple objectives can make the training process more complex and require careful balancing of the different losses. Prismer adopts a different approach, aligning with recent developments in VLMs that focus on language generation, and only require a single autoregressive training objective [WYH+22, WYY+21, HGW+22]. Despite the reduced complexity, training these large-scale VLMs is data intensive and computationally demanding, often requiring billions of training data. To overcome these challenges, Prismer leverages powerful pre-trained task-specific expert models for data-efficient training. Unlike another set of works that prioritise in-context capability by conditioning on a large frozen language model with no task-specific fine-tuning [EBW+21, TMC+21, ADL+22], Prismer focuses on fine-tuned performance with an emphasis on parameter efficiency, using smaller but diverse pre-trained models.

Multi-task and Auxiliary Learning Multi-task learning and auxiliary learning aim to train models to predict multiple outputs (such as semantic segmentation, object detection, and depth estimation) from a single input, thereby improving the performance across one or multiple tasks. This is often achieved through the design of effective multi-task networks that balance task-shared and task-specific features [LJD19, MSGH16, SPFS20, XOWS18], or through the explicit modelling of task relationships [LDJ19, LJDJ22, NAM+21, ZSS+18, FAZ+21]. Recently, multi-task learning has been further generalised to unify vision-only, language-only, and vision-language tasks by considering them within a sequence-to-sequence framework [WYM+22, LCZ+22a, ZZL+22]. Prismer also employs multiple tasks, specifically in the vision domain, similar to these methods, but uniquely uses them solely as input, serving as auxiliary knowledge. Prismer is more related to works such as [BMAZ22, GZC+21], which utilise pre-trained experts to create pseudo labels for multi-task self-training. However, whilst those methods focus on learning task-agnostic features through multi-task supervision, Prismer focuses purely on multi-modal reasoning with a single-task objective. Please refer to Sec. 2.2 and 2.3 for a detailed review.

Unifying Pre-trained Experts The utilisation of diverse pre-trained domain experts for multi-modal reasoning has been investigated in previous studies. Socratic models [ZWW+22] use language as a one-way communication interface to connect different pre-trained experts. ViperGPT [SMV23] and Visual Programming [GK23] harness the in-context learning capabilities of large language models, breaking down complex multi-modal reasoning into modular programs, which are then solved sequentially by leveraging

pre-trained vision experts through APIs. The aforementioned methods excel at modular problem decomposition and establishing connections among pre-trained experts, thereby being limited to zero-shot multi-modal reasoning within the domains on which the experts were pre-trained, and errors predicted by previous experts can be carried forward to future experts. However, Prismer stands out with a distinct objective by aiming to better bridge these pre-trained experts through a unified architecture design. As such, Prismer aims to create a more seamless collaboration between these experts, optimising multi-modal reasoning in a more integrated manner, and more robust to non-optimal experts.

Finally, we would like to highlight the distinction between the concept of "experts" defined in "Mixture of Experts (MoE)" [RPM+21, NC18, ME14] and in Prismer. In MoE, the "experts" are sub-modules in a single network, interconnected through their corresponding gating networks, encoding *implicit knowledge* guided by a shared training objective. On the other hand, in Prismer, the "experts" are independently pre-trained models, encoding *explicit knowledge* based on their pre-trained tasks or domains.

6.3 Prismer: Unifying Multi-Task Experts for Vision-Language Reasoning

To achieve this, we propose Prismer[1], a type of vision-language generative model that takes multi-task signals as input, and outputs free-form text.

Model Overview

The design of the Prismer model is illustrated in Fig. 6.2. Prismer is an encoder-decoder transformer model [VSP+17] that leverages a library of existing pre-trained experts. It consists of a vision encoder and an auto-regressive language decoder. The vision encoder takes an RGB image and its multi-task labels as input (*e.g.* depth, surface normal, segmentation labels, predicted from the frozen pre-trained experts), and outputs a sequence of RGB and multi-task features. The language decoder is then conditioned on these multi-task features via cross attention, and produces a sequence of text tokens.

One of the key advantages of the Prismer model is its exceptional data efficiency during training. This is achieved by leveraging *a combined power of strong domain-specific experts*, resulting in a significant reduction in the number of GPU hours required to achieve comparable performance to other state-of-the-art vision-language models. Prismer is built on top of existing pre-trained vision-only and language-only backbone models — this allows

[1] The model name "Prismer" draws from the analogy to an optical prism which breaks a white light into a spectrum of colours, and here we break down a single reasoning task into diverse domain-specific reasoning.

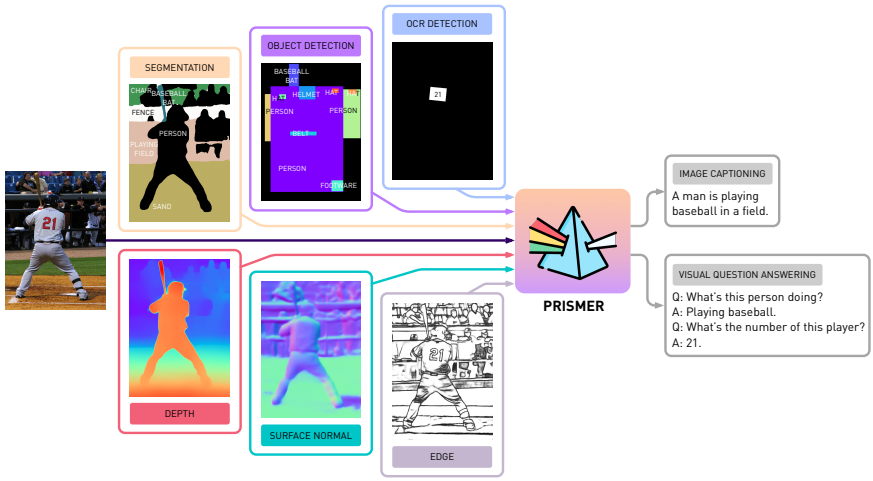


Figure 6.1. Prismer model overview. Prismer is a data-efficient vision-language model that leverages diverse pre-trained experts through its predicted multi-task signals. It can perform vision-language reasoning tasks such as image captioning and visual question answering. The analogy is with an optical prism: Prismer splits a single reasoning task into structured domain-specific reasoning.

us to tap into the vast amount of *web-scale knowledge* already stored in these pre-trained parameters. Additionally, we also extend the vision encoder to accept multi-task signals — this enables it to better capture semantics and information about the input image through the help of the *multi-task auxiliary knowledge*. For example, we expect "text-reading" problems can be easily solved by leveraging an OCR detection expert; and "object-recognition" problems can be easily solved by leveraging an object detection expert. A visualisation of all expert labels we included in Prismer is shown in Fig. 6.1.

Prismer is designed to leverage pre-trained experts while keeping the number of trainable parameters to a minimum. To do this, the network weights of the pre-trained experts are frozen to maintain the *integrity of their learned knowledge* and prevent catastrophic forgetting [KMA+18, KPR+17]. To link the multi-task knowledge as well as the vision and language parts of Prismer, we insert two parameter-efficient trainable components: *Experts Resampler* and *Adaptor*. The Experts Resampler is used in the vision encoder to map a variable length of multi-task signals to a sequence of multi-task features with a *fixed length*. The Adaptors are inserted in each transformer layer of the vision and language parts of the model to better adapt the pre-trained experts to new tasks and modalities.

Prismer is a *generative* model, and we re-formulate all vision-language reasoning tasks as a *language modelling or prefix language modelling* problem. For example, given the

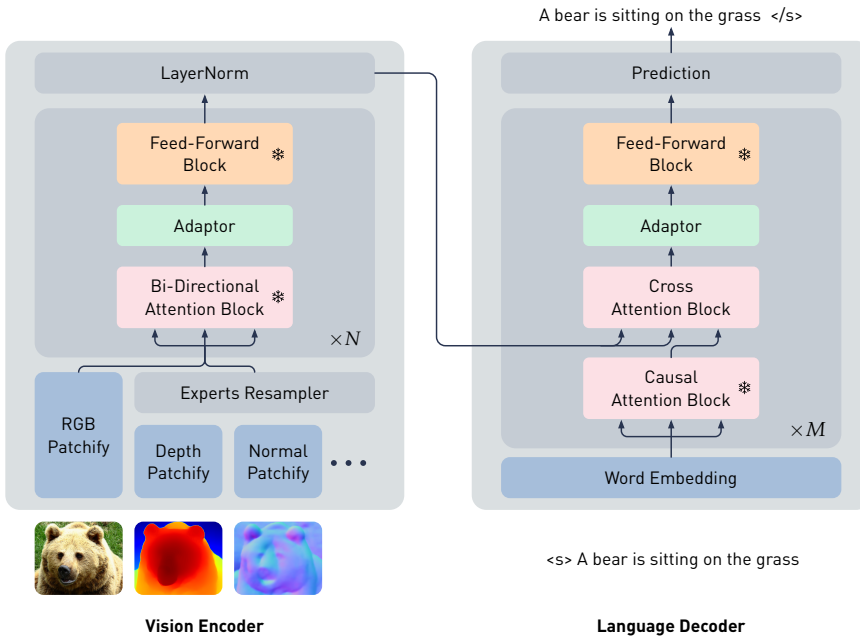


Figure 6.2. Prismer architecture design overview. Prismer has two main trainable components: the Experts Resampler which converts variable multi-task signals to a fixed number of outputs, and the Adaptor which enhances the model's expressivity for vision-language reasoning. To ensure that the model takes advantage of the rich domain-specific knowledge encoded in the pre-trained experts, the majority of network weights are frozen during training, as represented by ❄.

input image along with its multi-task tokens (predicted with the multi-task experts) and a question as the prefix, the model generates the answer for the visual question answering task; given the input image along with its multi-task tokens, the model generates its caption for the image captioning task. Once we have a prefix prompt, we may either sample the output text in an autoregressive manner, as in an *open-ended* setting; or we may rank the log-likelihood from a fixed set of completions, as in a *closed-ended* setting.

Pre-trained Experts

In Prismer, we include two types of pre-trained experts:

Backbone Experts. The vision-only and language-only pre-trained models, which are responsible for encoding images and texts into a meaningful sequence of tokens. Both models are required to be based on the transformer architecture [VSP+17], so we can easily

connect them with a few trainable components of similar designs. To preserve their rich domain-specific knowledge encoded in the network parameters, the majority of the weights are frozen during pre-training.

Task Experts. The models that can produce multiple task-specific labels depending on their training datasets. These task experts are treated as *black-box predictors*, can be designed either as a single multi-task expert or an ensemble of multiple task-specific experts, and their predicted labels are used as input for the Prismer model. As a result, all network weights of the task experts are frozen, and they can have *any design*. In Prismer, we include up to 6 task-specific experts all in the vision domain, encoding three *low-level* vision signals: depth, surface normals, and edge; and three *high-level* vision signals: object labels, segmentation labels, and OCR labels.

We apply task-specific post-processing on these predicted labels, transforming them to a $\mathbb{R}^{H\times W\times C}$ tensor (here H, W, C represent image height, width and channels respectively. *e.g.* $C = 1$ for depth and edge labels, and $C = 3$ for surface normals label). For all expert labels encoding high-level signals, we tile each pixel with its corresponding text embedding from a pre-trained CLIP text model [RKH+21], and then we apply PCA to down-sample the dimensionality to $C = 64$ for efficient training. The detailed descriptions of all task experts, including their pre-trained datasets and the architecture design, are listed in Table 6.1.

Task	Dataset	Model	Params.	Post-Processing
Semantic Segmentation	COCO-Stuff [CUF18]	Mask2Former [CMS+22]	215M	Tile each pixel with its corresponding label parametrised by CLIP text embedding.
Object Detection	COCO [LMB+14] + Objects365 [SLZ+19] + OpenImages [KRA+20] + Mapillary [NORBK17]	UniDet [ZKK22]	120M	Tile each pixel with its corresponding label parametrised by CLIP text embedding. The labels for the overlapping pixels are further determined by the depth expert.
Text Detection	ICDAR 2015 [KGBN+15]	CharNet [LCW18]	89M	Tile each pixel with its corresponding text parametrised by CLIP text embedding.
Depth Estimation	MIX-6 [RBK21]	DPT [RBK21]	123M	Re-normalised to $[-1, 1]$.
Surface Normal	ScanNet [DCS+17]	NLL-AngMF [BBC21]	72M	Re-normalised to $[-1, 1]$.
Edge Detection	BIPED [PRS20]	DexiNed [PRS20]	35M	Re-normalised to $[-1, 1]$.

Table 6.1. The detailed description of modality experts. We provide a detailed description of each modality expert including its pre-trained dataset, parameter size, model name and type and post-processing strategy.

Key Architectural Components

Task-Specific Convolutional Stem. All expert labels are first processed with randomly initialised convolution layers to map them to the same dimensionality. Specifically, we

apply 5 convolutional layers and each is composed of a small $[3 \times 3]$ kernel, which is shown to perform better than a single convolutional layer but with a larger kernel in the original Vision Transformer design [DBK+20], consistent with the finding in [XSM+21]. The convolutional stem is designed to be task-specific, which we have found to yield superior performance in comparison to a shared design in a multi-task learning setting [LJD19, MSGH16].

For high-level semantic labels such as those in object detection, semantic segmentation, and OCR detection, we down-sample the resolution by a factor of 4 to conserve running memory. Furthermore, for each object instance, we add a trainable and randomly sampled embedding to distinguish among different object instances. The size of this instance embedding is set to 128, which corresponds to the maximum possible number of object instances to be present in a single image. For RGB images, we simply process with the pre-trained convolutional stem defined by the original vision backbone. All task expert embeddings, including RGB, are then added with a pre-trained positional embedding before being further processed by transformer layers.

Experts Resampler The computational complexity of self-attention is *quadratically proportional* to the number of input patches. And therefore, the vision encoder can easily require tremendous memory when including a large number of modality experts. To address this issue, we propose *Experts Resampler*, which takes a *variable* number of expert labels as input and outputs a *fixed* number of embeddings, illustrated in Fig. 6.3 Left. Such design produces a *constant* memory for the self-attention computation in the vision encoder, as well as the vision-text cross attention in the language decoder (shown in Fig. 6.2), independent of the inclusion of a different number of experts. Inspired by the design in the Perceiver [JGB+21] and the Flamingo model [ADL+22], the Experts Resampler learns a pre-defined number of latent input queries, to cross-attend a flattened embedding concatenated from all multi-task features. The Resampler then compresses the multi-task features into a much smaller number of tokens equal to the number of learned latent queries, as a form of *auxiliary knowledge distillation*. We design keys and values to be a concatenation for both multi-task features and the learned latent queries, which is shown to be more effective, and consistent with the design in the Flamingo model [ADL+22].

Lightweight Adaptor We insert one lightweight adaptor into each transformer layer of both vision and language backbones to improve Prismer's expressivity and conditioning on multi-task features, illustrated in Fig. 6.3 Right. The adaptor has an encoder-decoder design, which has proven to be successful for efficient transfer learning in the NLP domain [HGJ+19,

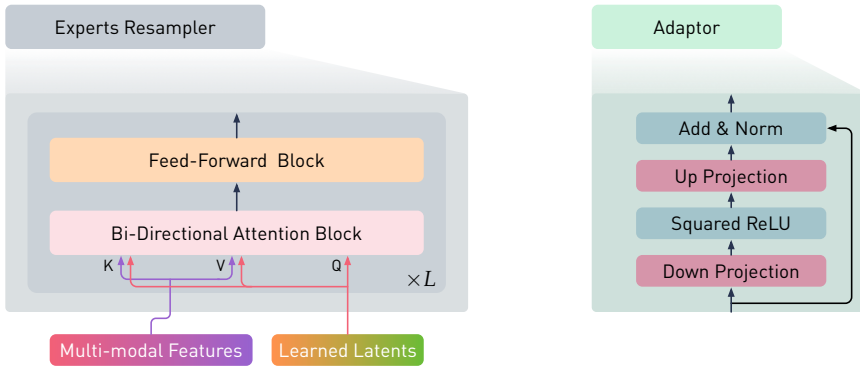


Figure 6.3. Design details in Experts Resampler and Adaptor. Left: The Experts Resampler takes multi-task features with variable length as input, and outputs a fixed number of tokens via cross attention. Right: The Adaptor has a residual connection to the input and two fully-connected layers, that down-project the input features to a smaller bottleneck dimension and then up-project back to the original dimension.

PRP+20]. It first down-projects the input features into a smaller dimension, applies a non-linearity, and then up-projects the features back to the original input dimension. We choose the non-linearity function to be squared ReLU [SML+21] — a simple and parameter-free function that delivers strong training stability. With the residual connection, we initialise all adaptors with near-zero weights to approximate the identity function. Combined with a standard cross attention block in the language decoder, the model is able to smoothly transition from the domain-specific vision-only and language-only backbones to a vision-language model during pre-training with paired image-text data.

Training Objective

For simplicity, we train Prismer with a *single* objective — to predict the next text token autoregressively. Following the standard encoder-decoder architecture, the vision encoder predicts the multi-task features z, and the language decoder learns to maximise the conditional likelihood of the paired text caption y under the forward autoregressive factorisation: $L = -\sum_{t=1}^{T} \log p(y_t | y_{<t}, z)$.

In practice, our *one-time* pre-processing step of collecting multi-task expert labels is computationally cheap and fast with data parallelism. The single generative objective then only requires one forward pass to compute gradients, which is significantly more efficient and streamlined than many other VLMs that may require a multi-stage and/or multi-step pre-training [LLXH22, LSG+21, WYM+22, DXG+22, CLY+20], with multiple objectives and

data sources. However, because our model only focuses on multi-modal language generation, it is less suitable for multi-modal discriminative tasks such as image-text retrieval and visual entailment, which are the focus of other types of VLMs [GCL+20, CLY+20, JYX+21].

6.4 Experiments

Prismer Model Variants

In addition to Prismer, we also introduce a model variant named PrismerZ, which solely relies on the power of strong backbone experts and is trained with *zero* task experts. PrismerZ has the same architectural design as the original Prismer but without the Experts Resampler. PrismerZ simplifies the data inference process as it only requires RGB images, making it more efficient and applicable to a wider range of applications. Prismer is less efficient in data inference due to the need for data processing on expert labels, but as we will show, it has better predictive performance.

Both Prismer and PrismerZ utilise ViT [DBK+20] pre-trained by CLIP [RKH+21] as the frozen vision encoder, and RoBERTa [LOG+19] as the frozen language decoder. We have alternatively tried using two other popular open-sourced decoder-only autoregressive language models: OPT [ZRG+22] and BLOOM [SFA+22], but early experiments showed that they did not perform as well.

We experiment with two model sizes, `BASE` and `LARGE`. The `BASE` model is built on top of `ViT-B/16` and $\texttt{RoBERTa}_{\texttt{BASE}}$, and the `LARGE` model is built on top of `ViT-L/14` and $\texttt{RoBERTa}_{\texttt{LARGE}}$. In Prismer, we apply the same Experts Resampler with roughly 50M parameters in both model sizes. The detailed architecture details are summarised in Table 6.2.

	Resampler		Vision Encoder			Language Decoder			Trainable Params.	Total Params.
	Layers	Width	Backbone	Layers	Width	Backbone	Layers	Width		
$\text{Prismer}_{\texttt{BASE}}$	4	768	`ViT-B/16`	12	768	$\texttt{RoBERTa}_{\texttt{BASE}}$	12	768	160M	980M
$\text{Prismer}_{\texttt{LARGE}}$	4	1024	`ViT-L/14`	24	1024	$\texttt{RoBERTa}_{\texttt{LARGE}}$	24	1024	360M	1.6B
$\text{PrismerZ}_{\texttt{BASE}}$	-	-	`ViT-B/16`	12	768	$\texttt{RoBERTa}_{\texttt{BASE}}$	12	768	105M	275M
$\text{PrismerZ}_{\texttt{LARGE}}$	-	-	`ViT-L/14`	24	1024	$\texttt{RoBERTa}_{\texttt{LARGE}}$	24	1024	270M	870M

Table 6.2. Prismer and PrismerZ architecture details. We report the backbone we choose for each architecture size, along with its corresponding number of layers and width. We also report the number of trainable parameters and total parameters for each architecture. We count the total parameters required for data inference, which include the additional 6 task experts with a combined parameter size of 654M parameters in our Prismer model.

Training and Evaluation Details

Pre-training Datasets. We construct our pre-training data from the following datasets: two in-domain datasets: COCO [LMB+14] and Visual Genome [KZG+17]; and three web datasets: Conceptual Captions [SDGS18], SBU captions [OKB11], and a much noisier Conceptual 12M [CSDS21]. The web datasets are pre-filtered and re-captioned by a pre-trained image captioner [LLXH22]. The pre-training datasets include 11M unique images or 12.7M image/alt-text pairs.[2] All datasets are available publicly and have been widely used for pre-training many VLMs [LSG+21, LLXH22, CLY+20].

Optimisation and Implementation. All models are trained with AdamW optimiser [LH19] with a weight decay of 0.05. Since only a small proportion of the model parameters are trainable, model sharding is only applied during fine-tuning on large-resolution images. Specifically, we employ ZeRO Stage 2 technique [RRRH20], which enables the sharding of optimiser states and parameter gradients across all GPU instances. Additionally, we apply Automatic Mixed Precision (AMP) with `fp16` precision to further reduce training time.

Evaluation Setting. We evaluate the performance of our models through *generative language modelling*, which is a more challenging task than discriminative learning (particularly in VQA tasks), and aligns with that used in other vision-language generative models [LLXH22, ADL+22, WYH+22, CWC+23]. For example, the model must accurately generate all text tokens for a question (which is on average 2.2 tokens per question in the VQAv2 dataset [AAL+15] as reported in [WYH+22]), rather than just one correct prediction as required in discriminative models.

Specifically, we evaluate image captioning tasks in an open-ended generative setting, and we apply beam search with a beam size of 3 for text generation. A prefix prompt of "A picture of" is added to the input text for fined-tuned image captioning tasks, similar to previous studies in [WYY+21, LLXH22, RKH+21], which have shown to improve the quality of image captions. We evaluate both VQA and image classification tasks in a close-ended generative setting, by ranking the per-token log-likelihood from a pre-defined answer list.

Training Cost. Prismer is highly efficient in terms of the training cost. The largest model variant, $\text{Prismer}_{\texttt{LARGE}}$, only requires 8 days of training on 32 NVIDIA V100 GPUs. This is significantly more efficient than previous state-of-the-art VLMs such as SimVLM [WYY+21] which requires 5 days of training on 2048 TPUv3, GIT-2 [WYH+22] which requires 1.5

[2] This is slightly less than the theoretical number which should be 14M unique images. This is because some image URLs in the web datasets were not valid during the time we downloaded the datasets.

months of training on 1500 NVIDIA A100s, and Flamingo [ADL+22] which requires 2 weeks of training on 1536 TPUv4. A detailed breakdown of the pre-training cost can be found in Table 6.3.

	Model Params.	Pre-training Data (# Image-Text Pairs)	Pre-training Cost (# PFlops Days)
$\text{BLIP}_{\texttt{LARGE}}$	583M	129M	22.2‡
$\text{SimVLM}_{\texttt{HUGE}}$	1.4B	1.8B	66.9‡
GIT	681M	0.8B	45.8‡
PaLI	17B	2.3B	450
Flamingo	80B	2.3B	1.4K†
GIT-2	5.1B	12.9B	5.5K†
$\text{Prismer}_{\texttt{BASE}}$	980M	12.7M	0.66
$\text{Prismer}_{\texttt{LARGE}}$	1.6B	12.7M	1.9

Table 6.3. Training cost of vision-language models. We compare the training cost of Prismer with several other vision-language models using the approximation method in [BMR+20]. † represents the training cost estimated by [CWC+23], and ‡ represents the training cost estimated by us.

Results on Vision-Language Benchmarks

Fine-tuned Performance on COCO Caption, NoCaps and VQAv2. We fine-tune our models on COCO Caption dataset [CFL+15] on a widely adopted Karpathy split [KFF15], with the standard cross-entropy loss, and without metric-specific optimisation [VLZP15]. We evaluate the fine-tuned models on the COCO Caption Karpathy test split and NoCaps [ADW+19] validation set. We also evaluate our models on the VQAv2 dataset [AAL+15], with additional training samples from Visual Genome [KZG+17] following [LLXH22]. We compare our models with prior state-of-the-art VLMs that are mostly pre-trained on image-text data for a fair comparison. We sort all VLMs by their model sizes and report the results in Table 6.4.

The results show that both Prismer and PrismerZ achieve superior performance considering their model sizes, which suggests that the strong backbone experts are primarily responsible for good generalisation. However, the task experts provide an additional boost in performance, particularly in image captioning tasks (such as a 6 CIDEr score increase in the NoCaps out-of-domain set in the `BASE` model) and in the `LARGE` model variant (such as a 1 VQAv2 accuracy increase in the `LARGE` model). Both $\text{Prismer}_{\texttt{BASE}}$ and $\text{Prismer}_{\texttt{LARGE}}$ achieve comparable image captioning performance to BLIP [LLXH22] and LEMON [HGW+22], despite being trained on 10 and 20 times less data, respectively. Additionally, the $\text{Prismer}_{\texttt{LARGE}}$ model has achieved VQAv2 accuracy comparable to GIT [WYH+22], despite being trained on 60 times less data. while we acknowledge a noticeable performance gap between Prismer and the current state-of-the-art VLMs (such as CoCa

	Pre-train (# Pairs)	COCO Caption				NoCaps				VQAv2	
		B @ 4	M	C	S	In	Near	Out	Overall	test-dev	test-std
OSCAR$_{\text{BASE}}$ [LYL$^+$20]	6.5M	36.5	30.3	123.7	23.1	83.4	81.6	77.6	81.1	73.2	73.4
VinVL$_{\text{BASE}}$ [ZLH$^+$21]	8.9M	38.2	30.3	129.3	23.6	103.7	95.6	83.8	94.3	76.0	76.1
GIT$_{\text{BASE}}$ [WYH$^+$22]†	10M	**40.4**	30.0	131.4	23.0	100.7	97.7	89.6	96.6	72.7	-
BLIP$_{\text{BASE}}$ [LLXH22]†	129M	39.7	-	133.3	-	**111.8**	**108.6**	111.5	**109.6**	**78.3**	**78.3**
LEMON$_{\text{BASE}}$ [HGW$^+$22]	200M	40.3	30.2	133.3	23.3	107.7	106.2	107.9	106.8	-	-
PrismerZ$_{\text{BASE}}$†	12.7M	39.7	31.1	133.7	24.1	108.7	107.8	105.8	107.5	76.6	-
Prismer$_{\text{BASE}}$†	12.7M	40.1	**31.1**	**135.1**	**24.1**	108.8	108.3	**111.7**	109.1	76.8	77.0
OSCAR$_{\text{LARGE}}$ [LYL$^+$20]	6.5M	37.4	30.7	127.8	23.5	85.4	84.0	80.3	83.4	73.4	73.8
VinVL$_{\text{LARGE}}$ [ZLH$^+$21]	8.9M	38.5	30.4	130.8	23.4	-	-	-	-	76.5	76.6
GIT$_{\text{LARGE}}$ [WYH$^+$22]†	20M	**42.0**	30.8	**138.5**	23.8	107.7	107.8	102.5	106.9	75.5	-
BLIP$_{\text{LARGE}}$ [LLXH22]†	129M	40.4	-	136.7	-	114.9	112.1	**115.3**	113.2	-	-
LEMON$_{\text{LARGE}}$ [HGW$^+$22]	200M	40.6	30.4	135.7	23.5	**116.9**	**113.3**	111.3	**113.4**	-	-
PrismerZ$_{\text{LARGE}}$†	12.7M	40.0	31.2	135.7	24.2	112.3	111.2	112.8	111.8	77.5	-
Prismer$_{\text{LARGE}}$†	12.7M	40.4	**31.4**	136.5	**24.4**	114.2	112.5	113.5	112.9	**78.4**	**78.5**
LEMON$_{\text{HUGE}}$ [HGW$^+$22]	200M	41.5	30.8	139.1	24.1	118.0	116.3	120.2	117.3	-	-
SimVLM$_{\text{HUGE}}$ [WYY$^+$21]	1.8B	40.6	33.7	143.3	25.4	113.7	110.9	115.2	112.2	80.0	80.3
GIT [WYH$^+$22]†	0.8B	44.1	31.5	144.8	24.7	129.8	124.1	127.1	125.5	78.6	78.8
GIT-2 [WYH$^+$22]†	12.9B	44.1	31.4	145.0	24.8	126.9	125.8	130.6	126.9	81.7	81.9
CoCa [YWV$^+$22]	4.8B	40.9	33.9	143.6	24.7	-	-	-	122.4	82.3	82.3
PaLI [CWC$^+$23]†	1.6B	-	-	149.1	-	-	-	-	127.0	84.3	84.3

Table 6.4. Fine-tuned performance on COCO Caption (Karpathy split), NoCaps (validation set) and VQAv2. Both Prismer and PrismerZ achieve superior performance in all three datasets compared to other VLMs with similar model sizes. Prismer can achieve competitive performance on par with VLMs that are trained with orders of magnitude more data. {B@4, M, C, S} refer to BLEU@4, METEOR, CIDEr, SPICE respectively. {In, Near, Out} refer to in-domain, near-domain and out-of-domain respectively. † evaluates the VQAv2 dataset in a generative setting, while all other models evaluate the VQAv2 dataset in a closed-ended discriminative setting.

[YWV$^+$22], GIT-2 [WYH$^+$22] and PaLI [CWC$^+$23]), these models require substantially higher training costs and access to large-scale private training data.

Zero-shot Performance on Image Captioning. Our generative pre-training approach allows for zero-shot generalisation, where the models can be directly applied to image captioning tasks without additional fine-tuning. In Fig. 6.4 Left, we show that Prismer achieves state-of-the-art performance on the NoCaps dataset similar to SimVLM [WYY$^+$21] and BLIP-2 [LLSH23], while using significantly smaller network parameter size and trained with 140 times and 10 times less data respectively. Additionally, we notice that the zero-shot performance of Prismer models even surpasses the fine-tuned performance of certain VLMs such as OSCAR [LYL$^+$20] and VinVL [ZLH$^+$21], as shown in Table 6.4.

We present a list of example captions generated by Prismer in Table 6.5. The results show that both Prismer$_{\text{BASE}}$ and Prismer$_{\text{LARGE}}$ are capable of generating captions that are semantically

	NoCaps	
	C	S
FewVLM [JCS+22]	47.7	9.1
MetaLM [HSD+22]	58.7	8.6
VLKD [DHS+22]	63.6	12.8
SimVLM$_{\text{HUGE}}$ [WYY+21]	101.4	-
BLIP-2 (Vicuna-7B) [LLSH23]	107.5	-
BLIP-2 (Vicuna-13B) [LLSH23]	103.9	-
Prismer$_{\text{BASE}}$	87.5	13.0
Prismer$_{\text{LARGE}}$	107.9	14.8

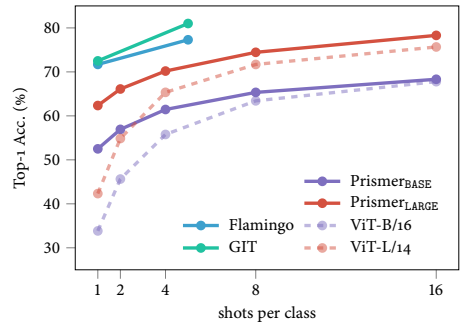


Figure 6.4. Results on zero-shot image captioning and few-shot ImageNet classification. Left: Prismer achieves state-of-the-art zero-shot image-captioning results on NoCaps (validation set), with similar performance to SimVLM and BLIP-2, despite being trained on 140 times and 10 times less data respectively. Right: Prismer significantly improves few-shot performance compared to its corresponding vision backbone. However, Prismer still underperforms GIT and Flamingo which are trained on significantly more data.

coherent and aligned with the visual content of the images. Notably, Prismer$_{\text{LARGE}}$ generates captions of higher quality compared to Prismer$_{\text{BASE}}$, exhibiting a deep understanding of fine-grained object semantics such as brand recognition (*e.g.* Mercedes, CK One), and cultural concepts (*e.g.* vintage drawing, tango), indistinguishable to human-written captions.

Few-shot Performance on ImageNet Classification. Finally, we fine-tune and evaluate Prismer on ImageNet [DDS+09] in a few-shot setting. Following the approach outlined in [RKH+21], we convert the classification task into a language modelling problem by mapping each category to a template caption: "A photo of a [CLASS NAME]", and we then score captions using the log-likelihood estimated by our model. Unlike Flamingo [ADL+22] which performs few-shot classification via in-context examples without gradient updates, we perform few-shot classification via lightweight fine-tuning following [WYH+22]. This is more similar to the standard linear probe setting, by considering the entire language decoder as an image classifier. Accordingly, we also compare with the few-shot linear probe performance of Prismer's original vision backbones ViT-B/16 and ViT-L/14 [DBK+20], as reported in [SBV+22, RKH+21].

From the results shown in Fig. 6.4 Right, we observe that Prismer underperforms GIT [WYH+22] and Flamingo [ADL+22], which both have stronger vision backbones and are pre-trained on significantly more data. However, Prismer still outperforms its original

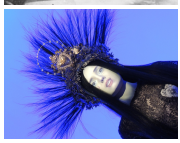
	Ground-Truth	Prismer$_{\texttt{BASE}}$	Prismer$_{\texttt{LARGE}}$
	*1. A clear bottle of CK cologne is full of liquid.* *2. The bottle of perfume is made by Calvin Klein.*	*A bottle of alcohol sitting next to a computer keyboard.*	*A bottle of ck one next to a computer keyboard.*
	*1. A young child stands in front of a house.* *2. A little boy is standing in his diaper with a white shirt on.*	*An old photo of a little girl standing on a step.*	*An old black and white photo of a baby standing in front of a house.*
	*1. A statue has a large purple headdress on it.* *2. A woman decorated in fashioned clothing and relics.*	*The woman is wearing a black dress.*	*A mannequin dressed in a black dress with feathers on her head.*
	*1. A new white car with the door open is in a showroom full of people.* *2. A shiny white mercedes car is on display.*	*A white car on display at a car show.*	*A white mercedes car on display at an auto show.*
	*1. Large piece of meat with slices of pineapple with cherries being held on with toothpicks on blue and white plate.* *2. A cake has several slices of pineapple and cheries in them.*	*Pineapples on a plate.*	*Pineapple upside down cake on a blue and white plate.*
	*1. A man and woman is dancing as a crowd watches them in the distance.* *2. A woman in a red dress dancing with a bald man wearing black.*	*A couple of people that are standing in the dirt.*	*A couple dancing tango in front of a crowd.*
	*1. Man in skydiving gear giving two thumbs up with skydivers in the sky behind him.* *2. Person giving double thumbs up sign while others are parachuting in the background.*	*Man wearing a blue and purple jacket.*	*A man wearing a helmet and goggles with parachutes in the background.*

Table 6.5. Visualisation of zero-shot image captioning on NoCaps. Prismer$_{\texttt{LARGE}}$ produces more detailed and semantically coherent captions than Prismer$_{\texttt{BASE}}$, showing an understanding of fine-grained object recognition and abstractions.

vision backbones ViT-B and ViT-L by a large margin, especially in a very few-shot setting. This suggests that Prismer's generalisation abilities are enhanced by the multi-modal training data and expert labels, and its performance can likely be improved further by using an even stronger vision backbone.

6.5 Learning Strategy and Utility Analysis of Multi-Task Experts

We now include a comprehensive evaluation of Prismer, characterised by a meticulous and fine-grained analysis of its learning strategy. We delve into various aspects of Prismer's performance, examining its behaviour with different types of multi-task experts (as discussed in Sec.6.5). Additionally, we explore the individual utility of each expert in addressing domain-specific reasoning tasks, allowing us to gain insights into the specific strengths and contributions of each expert (as discussed in Sec.6.5).

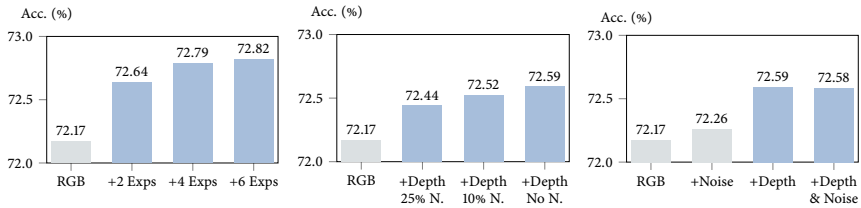
Intriguing Learning Strategy of Prismer

To speed up training, all experiments are conducted with the `BASE` model on a combined dataset of the Conceptual Captions and SBU, consisting of a total of 3M data. All experiments are evaluated on the VQAv2 `test-dev` split in a smaller $[224 \times 224]$ resolution.

More Experts, Better Performance. We observe that the performance of Prismer improves with more task experts, as shown in Fig. 6.5a. This is intuitive because more experts provide a greater diversity of domain knowledge to the model. However, we also note that the performance of the model eventually plateaus, which suggests that additional task experts beyond a certain number do not provide any extra gains.

Better Experts, Better Performance. To evaluate the impact of expert quality on Prismer's performance, we construct a *corrupted* depth expert by replacing a certain number of predicted depth labels with random noise sampled from a Uniform Distribution. As shown in Fig. 6.5b, Prismer's performance improves as the quality of the depth expert improves. This is intuitive as better experts provide more accurate domain knowledge, allowing the model to perceive more accurately.

Robustness to Noisy Experts. Our results also demonstrate that Prismer maintains performance even when including experts that predict noise, as shown in Fig. 6.5c. Interestingly, adding noise can even result in a non-trivial improvement compared to training on RGB images alone, which can be considered as a form of implicit regularisation. This property allows the model to safely include many experts *without degrading the performance*, even



(a) with More Experts (b) with Better Experts (c) with Noisy Experts

Figure 6.5. Prismer's VQAv2 accuracy with different types and the number of experts. Prismer has shown that its performance improves with an increase in the number and quality of task experts. Additionally, Prismer also demonstrates its strong robustness to noisy experts, making it a practical and effective multi-modal learning strategy.

when the expert is *not necessarily informative*. Therefore, Prismer presents a more effective learning strategy than the standard multi-task or auxiliary learning methods, which either require exploring task relationships [LJDJ22, FAZ+21, ZSS+18] or designing more advanced optimisation procedures [LDJ19, NAM+21].

Utility of Task Experts

In this experiment, we conduct a comprehensive evaluation to assess the utility of each task expert within Prismer concerning different types of reasoning tasks. To accomplish this, we employ Prismer$_{\texttt{LARGE}}$, which was trained on the VQAv2 dataset, and evaluate its zero-shot performance in combination with each individual task expert on two specific domain-specific reasoning tasks: i) Visual Spatial Reasoning (VSR) [LEC23]: This task evaluates a VLM's spatial reasoning ability. It involves classifying image-caption pairs as either true or false, indicating whether the caption correctly describes the spatial relation in an image. ii) Text-VQA [SNS+19]: This task assesses a VLM's ability to understand and reason about text within an image. It involves comprehending and answering questions related to text in an image.

The results presented in Table 6.6 demonstrate that Prismer consistently outperforms several competitive baselines, such as VisualBERT [li220], LXMERT [TB19], and ViLT [KSK21] in the VSR dataset, all without requiring dataset-specific fine-tuning as required by these methods. Prismer also surpasses BLIP-2 [OPT 2.7B] [LLSH23] and OFA$_{\texttt{HUGE}}$ [WYM+22], despite employing a smaller backbone network and significantly less pre-training data respectively.

Furthermore, Prismer's utility analysis offers valuable insights into the contributions of in-

Baselines (fine-tuned)			Prismer (zero-shot)							
VisualBERT	LXMERT	ViLT	+Depth	+Normal	+Edge	+Seg.	+OCR Det.	+Obj. Det.	No Experts	+6 Experts
51.0	61.2	63.0	68.4	68.3	67.8	68.4	67.2	68.3	65.6	68.7

(a) VSR

Baselines (zero-shot)			Prismer (zero-shot)							
OFA	BLIP-2	Flamingo	+Depth	+Normal	+Edge	+Seg.	+OCR Det.	+Obj. Det.	No Experts	+6 Experts
18.3	15.7	35.0	27.4	28.0	28.2	27.8	28.4	28.4	22.6	28.8

(b) Text-VQA

Table 6.6. Zero-shot accuracies in VSR (zero-shot split) and Text-VQA (validation split) datasets, considering various types of experts. These results shed light on the valuable contributions of individual experts for domain-specific reasoning tasks, offering insights into the versatility and adaptability of Prismer across different domains and problem-solving scenarios. The colour green represents the most helpful experts, while the colour red represents the least helpful experts.

dividual experts in addressing specific reasoning tasks. For example, the "object detection" expert is identified as crucial in both the VSR and Text-VQA tasks, highlighting the significance of object recognition capability in general visual reasoning problems. Additionally, the "depth" and "OCR detection" experts are recognised as key contributors to Prismer's performance in spatial reasoning and text reasoning, respectively, aligning with human intuition — depth information enhances 3D spatial understanding, whilst OCR detection directly improves text reading capability.

Finally, the substantial performance improvement observed (compared to standard reasoning tasks in Table 6.4) when comparing Prismer to PrismerZ (with no experts) underscores the pivotal role played by the experts in domain-specific reasoning tasks. This highlights the tangible benefits of incorporating experts within the Prismer architecture, particularly when tackling tasks that require specialised knowledge and reasoning capabilities.

6.6 Ablation Study on Architecture Design and Training Details

To perform the ablation studies, we use the Prismer$_{\texttt{BASE}}$ model and train it on the Conceptual Captions and SBU with a total of 3M training data. The results of the ablation studies are presented in Table 6.7

Adaptor Design: Single and Simple In our ablation study of adaptor designs, as shown in row (i) and (ii) of Table 6.7, we find that the simplest adaptor design, which consisted of a standard residual connection and an encoder-decoder structure, performs the best. We have experimented with more complex designs, such as adding an additional adaptor at

Ablated Component	Our Setting	Changed Setting	Params. (Rel.)	Step Time (Rel.)	VQAv2 (Acc.)
Prismer$_{\text{BASE}}$ (our setting with reduced training)			**1.00**	**1.00**	**72.79**
(i) Adapter Design	Residual MLP	Residual MLP ×2	1.04	1.02	72.36
		Gated Residual MLP	1.03	1.03	70.54
(ii) Adapter Bottleneck Dim.	1	1/2	0.95	0.96	72.52
		1/4	0.93	0.93	71.66
(iii) Resampler Design	Experts Perceiver	Random Sampling	0.91	0.96	72.24
		Full Perceiver	1.00	0.90	65.05
		Dual Perceiver	1.08	1.02	71.56
(iv) Resampler Layers	4	1	0.94	0.93	70.61
		2	0.96	0.96	72.39
		6	1.04	1.01	72.78
(v) Resampler Latents	64	32	1.00	0.95	72.44
		128	1.00	1.01	70.28
		256	1.00	1.06	68.07
(vi) Pre-training	Freeze Vision and Lang.	Freeze Vision Only	1.00	1.07	70.49
		Freeze Lang. Only	1.00	1.05	67.77
		All Parameters	1.00	1.15	68.13
(vii) Fine-tuning	Freeze Vision	Freeze Vision and Lang.	1.00	1.00	71.36
		Freeze Lang. Only	1.00	1.00	70.37
		All Parameters	1.00	1.00	68.69

Table 6.7. Ablation studies for architecture components and training strategies. We perform ablation studies to evaluate the impact of different architectural components and training strategies on the VQAv2 `test-dev` performance. We compare the performance of our default setting to other design and training options. The number of parameters and pre-training step time of the changed setting relative to the default setting are reported. To ensure a fair comparison, all experiments are evaluated using a reduced amount of training data and 3 task experts: depth, normal and segmentation.

the end of each transformer layer or incorporating a learnable gating mechanism similar to that in [LDSL21], but both have achieved worse performance. We also observe that having a larger bottleneck hidden size for the single adaptor improves performance.

Resampler Design: Auxiliary Knowledge Distillation In our ablation study of Experts Resampler designs and different sampling strategies for encoding multi-task signals, as shown in row (iii) - (v) of Table 6.7, we find that keeping the number of resampler layers and latents lightweight is essential for a stable training process. We also experiment with replacing the resampler with a non-learnable random sampling approach, which results in slightly worse performance compared to using the resampler. We attempt to make the resampler more efficient by receiving full signals, including the RGB, before self-attention, but this has resulted in significantly degraded performance. Additionally, we have tried adding an additional resampler at the end of the vision encoder, but this design also results in worse performance.

Frozen Backbone to Preserve Web-Scale Knowledge In our experiments on pre-training and fine-tuning whilst freezing different parts of the model, as shown in row (vi) and (vii) of Table 6.7, we find that freezing pre-trained parameters is essential for achieving strong performance and avoiding over-fitting and catastrophic forgetting of the learned web-scale knowledge.[3] Freezing these parameters has also saved a significant amount of GPU memory. Even when fine-tuning on different downstream tasks, we find that freezing the vision encoder is beneficial (whilst keeping the resampler and adaptors trainable). This observation is consistent with the findings in [ZWM+22], which shows that only fine-tuning the language model with a frozen vision model for vision-language contrastive learning can achieve much stronger zero-shot performance.

6.7 Conclusions, Limitations and Discussion

In this chapter, we have introduced Prismer, a vision-language model designed for reasoning tasks. Prismer is parameter-efficient and utilises a small number of trainable components to connect an ensemble of diverse, pre-trained experts. By leveraging these experts, Prismer achieves competitive performance in image captioning, VQA, and image classification benchmarks, comparable to models trained on up to two orders of magnitude more data.

For full transparency, we now discuss some limitations of Prismer during our implementation and explore potential future directions for this work.

Multi-modal In-context Learning Zero-shot in-context generalisation is an emergent property that only exists in very large language models [BMR+20, WTB+22]. In this work, we build Prismer on top of a small-scale language model with the main focus on parameter-efficient learning. Therefore, it does not have the ability to perform few-shot in-context prompting by design.

Zero-shot Adaptation on New Experts We experiment with inference on a pre-trained Prismer with a different segmentation expert pre-trained on a different dataset. Although we apply the same language model to encode semantic labels, Prismer shows limited adaptability to a different expert with a different set of semantic information, which leads to a notable performance drop.

Free-form Inference on Partial Experts Similarly, we discover that Prismer entangles its multi-task features from all experts we include during pre-training. Therefore, only having

[3] We assume the size of our pre-training data is significantly smaller than the original pre-training data used to train these backbone models.

a partial number of experts during inference will lead to a notable performance drop. We attempt to use a different training objective such as masked auto-encoding [BMAZ22], to design Prismer to reason on an arbitrary number of experts, but it eventually leads to a degraded fine-tuned performance.

Representation of Expert Knowledge In our current design of Prismer, we convert all multi-task expert labels into an image-like 3-dimensional tensor via task-specific post-processing for simplicity. There are potentially other efficient methods to represent expert knowledge, such as converting object detection into a sequence of text tokens [CSL+21, CSL+22]. This may lead to stronger reasoning performance and a more stable training landscape in future works.

Impact on Future Research Prismer has emerged as a competitive baseline for parameter-efficient vision-language learning [WLY+23, BDPAT23, RBB+23] and has served as inspiration for recently proposed multi-modal neural architecture designs, with particular attention to expert and modality ensembling [BGM+23, SFH+23, YYB24].

7

Conclusions and Future Works

In this thesis, we have introduced several multi-task learning techniques aimed at enhancing generalisation and interpretability within the field of computer vision. This concluding chapter provides an overview of the contributions made in each preceding chapter. And finally, we discuss potential directions for future research in this area.

7.1 Summary of Contributions

In Chapter 3, we have introduced an automated weighting framework known as Auto-λ, which is designed to streamline the process of uncovering multi-task relationships. In contrast to typical task-grouping methods that assume fixed task relationships, Auto-λ explores dynamic task relationships by employing task-specific weightings. Its adaptability allows for the optimisation of any combination of tasks, rendering it a versatile solution for a wide range of multi-task and auxiliary learning challenges within the domains of computer vision and robotics. Our experimental findings indicate that Auto-λ achieves state-of-the-art performance, even when compared to optimisation strategies tailored for these specific problems and data domains. Furthermore, we have observed that Auto-λ has the capacity to unveil intriguing learning patterns, thus contributing novel insights into understanding transferred task knowledge and the relationships between tasks.

In Chapter 4, we have introduced an auxiliary learning framework for semantic segmentation known as ReCo, which has proven to be a powerful framework for improving semantic

segmentation models. By leveraging regional-level contrastive learning and focusing on challenging pixels guided by semantic class relationships, we have successfully achieved significant performance gains in both semi-supervised and supervised learning settings. ReCo's ability to facilitate high-quality segmentation models with minimal labelled data can significantly alleviate the burden on human annotators by reducing the number of labelled examples required for effective model training. This has promising implications for enhancing the efficiency and accuracy of the human labelling process, thereby advancing the field of semantic segmentation.

In Chapter 5, we have introduced and validated the effectiveness of the Meta Auxiliary Learning (MAXL) framework, which involves the training of two neural networks: a label-generation network and a multi-task network. The label-generation network is responsible for generating auxiliary labels, while the multi-task network trains the primary tasks alongside the generated auxiliary tasks. MAXL's label-generation network aims to create well-structured auxiliary tasks that significantly improve the generalisation of primary tasks, achieved through their simultaneous training in a standard multi-task learning setting. MAXL demonstrates its capacity to significantly boost single-task learning across diverse image datasets, all without the need for additional data. Importantly, our results highlight that MAXL not only outperforms baseline methods for generating auxiliary labels but also competes favourably with human-defined auxiliary labels, making it a promising solution for self-supervised generalisation in machine learning.

In Chapter 6, we have introduced Prismer, a parameter-efficient vision-language model that harnesses an ensemble of task-specific experts, a majority of whose network weights are pre-trained and remain fixed during training. This approach offers a more scalable alternative compared to conventional vision-language models that demand extensive training on massive datasets, drastically reducing the need for data and training resources. Prismer exhibits competitive fine-tuned and few-shot learning performance in image captioning, VQA, and image classification benchmarks, while requiring up to two orders of magnitude less training data. Moreover, it breaks away from the conventional monolithic models by utilising separate sub-networks or "experts" that can be individually optimised for specific tasks, thereby improving training efficiency and allowing for domain-specific data and architectures. This approach facilitates the integration of specialised skills and structured domain knowledge, presenting an effective way of scaling down multi-modal learning.

We can also draw more general conclusions and broader insights that extend beyond their individual contributions. First, they underscore that the benefits and applications of multi-task learning extend well beyond its original purpose of improving generalisation.

Multi-task learning can play a pivotal role in improving model interpretability (shown in Auto-λ), reducing data requirements (shown in ReCo and MAXL), and scaling down model size (shown in Prismer). This versatility highlights the adaptability and potential of multi-task learning in addressing a variety of challenges in the fields of computer vision and machine learning. Second, these works highlight the significance of incorporating human guidance and prior knowledge into the design of effective optimisation strategies. While it's possible to employ end-to-end multi-task learning without imposing structure, it is evident that the involvement of human insight and guidance can significantly benefit the development of optimisation strategies by reducing the optimisation search space and leading to more efficient and effective solutions (such as the hierarchical structure design in MAXL and the selection of expert models in Prismer). This emphasises the importance of a balanced and informed approach to multi-task learning, where human expertise can play a crucial role in shaping the learning process, designing effective auxiliary signals, and achieving superior performance.

7.2 Future Works

Undoubtedly, the scope of this thesis does not include all aspects of multi-task learning. In this section, we explore some potential directions for future research in this area.

Vision Foundation Model for Multi-Task Perception

The remarkable zero-shot multi-task learning capabilities of large language foundation models in language generation tasks [Ope23, CND+22] have yet to find an equivalent in the domain of computer vision for multi-task perception tasks. Up to this point, the best-performing models still rely on domain-specific designs and training strategies, making them less adaptable to other tasks. For example, the Segment Anything Model [KMR+23] for object detection relies on a mask decoder, and the NLL-AngMF model [BBC21] for surface normal estimation relies on uncertainty-guided sampling, both of which are specific to their respective tasks, hindering their broader application.

Recent efforts like Unified-IO [LCZ+22b] and BEiT-3 [WBD+22] aim to unify vision and language tasks by converting visual data into tokens for processing by language models, as a standard sequence-to-sequence learning problem. However, these models still require task-specific decoders and fine-tuning to achieve strong performance. Consequently, the design of a vision foundation model that can excel in a diverse array of multi-task perception tasks without the need for task-specific modifications and fine-tuning remains an ongoing challenge and an area of uncertainty.

Multi-Modal Representation Learning

Humans possess the remarkable ability to perceive the world through multiple modalities. For instance, we can form mental images of scenes by reading descriptions or listening to stories, and we can discern a person's emotions by observing their facial expressions or listening to their voice. Therefore, the pursuit of representation learning techniques that are universally applicable across different modalities is crucial. Currently, representation learning strategies in various data domains remain distinct. In language, self-supervised representation learning commonly relies on auto-regressive generation, where the goal is to predict the next tokens based on previous tokens (such as in GPT-3 [RW19]). In vision, self-supervised representation learning primarily employs contrastive learning objectives, aiming to maximise the similarity between different views of the same image (as seen in methods like SimCLR [CKNH20] and MoCo [HFW+20]), or masked auto-encoding to predict missing pixels in an image (as seen in methods like MAE [HCX+21]).

CLIP [RKH+21], a multi-modal vision-language model, has effectively demonstrated that the contrastive representation learning technique can be applied to align vision and language, leading to successful methods for building vision-language models like Prismer (as presented in Chapter 6). Nevertheless, there is an underlying importance in discovering general representation learning techniques that can be applied to more modalities. Such innovation holds the potential to construct more efficient and powerful multi-modal learning models, enabling broader applications across many domains.

Open-Ended Exploration and Lifelong Learning

Finally, multi-task learning holds significant potential in advancing open-ended exploration and lifelong learning, both critical for the pursuit of human-level AI. Open-ended exploration [WLCS19, TSM+21] involves the continuous acquisition of skills and knowledge over time without pre-defined goals or limitations, making it a highly desirable trait for AI systems. Nevertheless, most machine learning models are tailored for specific tasks, being trained on fixed datasets with pre-defined labels, which can lead to "catastrophic forgetting" [Fre99] when exposed to new data, causing them to forget previously acquired knowledge. To design AI models capable of lifelong learning and skill improvement, we must innovate novel learning strategies that can effectively and continually integrate new knowledge, laying the foundation for more adaptive and lifelong learning AI systems.

Bibliography

[AAL+15] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[ADG+16] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[ADL+22] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[ADW+19] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. Nocaps: Novel object captioning at scale. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019.

[ASF+21] Iñigo Alonso, Alberto Sabater, David Ferstl, Luis Montesano, and Ana C Murillo. Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[BBC90] Yoshua Bengio, Samy Bengio, and Jocelyn Cloutier. *Learning a synaptic learning rule*. Université de Montréal, Département d'informatique et de recherche opérationnelle, 1990.

[BBC21] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Estimating and exploiting the aleatoric uncertainty in surface normal estimation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[BBCG92] Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. On the optimization of a synaptic learning rule. In *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, 1992.

[BBH+22] Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*, 2022.

[BCC+20] Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[BCG+19] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[BDPAT23] Daniela Ben-David, Tzuf Paz-Argaman, and Reut Tsarfaty. Apollo: Zero-shot multimodal reasoning with multiple experts. *arXiv preprint arXiv:2310.18369*, 2023.

[BGM+23] Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. *arXiv preprint arXiv:2312.00785*, 2023.

[BLCW09] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[BMAZ22] Roman Bachmann, David Mizrahi, Andrei Atanov, and Amir Zamir. MultiMAE: Multi-modal multi-task masked autoencoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[BMR+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. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[BV16] Hakan Bilen and Andrea Vedaldi. Integrated perception with recurrent multi-task neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[Car97] Rich Caruana. Multitask learning. *Machine learning*, 1997.

[CBLR18] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[CFL+15] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[CKNH20] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[CLB+17] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[CLTB21] Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. Unifying vision-and-language tasks via text generation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[CLY+20] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[CMS+22] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[CND+22] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[CNH+20] Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[COR+16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[CPK+17] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017.

[CSDS21] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[CSL+21] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[CSL+22] Ting Chen, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J. Fleet, and Geoffrey Hinton. A unified sequence interface for vision tasks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[CSL+23] Qiongjie Cui, Huaijiang Sun, Jianfeng Lu, Bin Li, and Weiqing Li. Meta-auxiliary learning for adaptive human pose prediction. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2023.

[CUF18] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[CWC+23] 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. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[CWG+22] Hong Chen, Xin Wang, Chaoyu Guan, Yue Liu, and Wenwu Zhu. Auxiliary learning with joint task and data scheduling. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

[CWYT21] Zhixiang Chi, Yang Wang, Yuanhao Yu, and Jingshan Tang. Test-time fast adaptation for dynamic scene deblurring via meta-auxiliary learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[CZP+18] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[DBK+20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[DCBC15] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2015.

[DCJ+18] Yunshu Du, Wojciech M Czarnecki, Siddhant M Jayakumar, Razvan Pascanu, and Balaji Lakshminarayanan. Adapting auxiliary losses using gradient similarity. *arXiv preprint arXiv:1812.02224*, 2018.

[DCLT19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Long and Short Papers)*, 2019.

[DCS+17] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[DFL23] Yanqi Dai, Nanyi Fei, and Zhiwu Lu. Improvable gap balancing for multi-task learning. In *Uncertainty in Artificial Intelligence*, 2023.

[dGML+21] Daan de Geus, Panagiotis Meletis, Chenyang Lu, Xiaoxiao Wen, and Gijs Dubbelman. Part-aware panoptic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[DHS16] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[DHS+22] Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. Enabling multimodal generation on clip via vision-language knowledge distillation. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2022.

[DLLT21] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[DR19] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[DXG+22] Zi-Yi Dou, Yichong Xu, Zhe Gan, Jianfeng Wang, Shuohang Wang, Lijuan Wang, Chenguang Zhu, Pengchuan Zhang, Lu Yuan, Nanyun Peng, et al. An empirical study of training end-to-end vision-and-language transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[EBW+21] Constantin Eichenberg, Sidney Black, Samuel Weinbach, Letitia Parcalabescu, and Anette Frank. Magma–multimodal augmentation of generative models through adapter-based finetuning. *arXiv preprint arXiv:2112.05253*, 2021.

[EEVG+15] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 2015.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[FAL+20] Geoff French, Timo Aila, Samuli Laine, Michal Mackiewicz, and Graham Finlayson. Semi-supervised semantic segmentation needs strong, high-dimensional perturbations. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.

[FAZ+21] Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[FDFP17] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[FFS+18] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[FNPS16] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[Fre99] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 1999.

[GBJ+20] Yuan Gao, Haoping Bai, Zequn Jie, Jiayi Ma, Kui Jia, and Wei Liu. Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[GCG+22] Pierre-Louis Guhur, Shizhe Chen, Ricardo Garcia, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. Instruction-driven history-aware policies for robotic manipulations. In *Conference on Robot Learning (CoRL)*, 2022.

[GCL+20] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[GFDZ22] Yingying Gao, Junlan Feng, Chaorui Deng, and Shilei Zhang. Meta auxiliary learning for low-resource spoken language understanding. In *Interspeech*, 2022.

[GHH+18] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[GK23] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[GLU20] Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to branch for multi-task learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[GMZ+19] Yuan Gao, Jiayi Ma, Mingbo Zhao, Wei Liu, and Alan L Yuille. Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[GNP22] Agus Gunawan, Muhammad Adi Nugroho, and Se Jin Park. Test-time adaptation for real image denoising via meta-transfer learning. *ArXiv*, 2022.

[GXGF23] Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[GZC+21] Golnaz Ghiasi, Barret Zoph, Ekin D Cubuk, Quoc V Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[HAMS20] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[HCX+21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

[HDO+98] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 1998.

[HFW+20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[HGJ+19] 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 *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[HGW+22] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[HLVDMW17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[HMBS21] Falk Heuer, Sven Mantowsky, Saqib Bukhari, and Georg Schneider. Multitask-centernet (mcn): Efficient and diverse multitask learning using an anchor free approach. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[HSD+22] Yaru Hao, Haoyu Song, Li Dong, Shaohan Huang, Zewen Chi, Wenhui Wang, Shuming Ma, and Furu Wei. Language models are general-purpose interfaces. *arXiv preprint arXiv:2206.06336*, 2022.

[HSS18] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 1989.

[HTL+19] Wei Chih Hung, Yi Hsuan Tsai, Yan Ting Liou, Yen Yu Lin, and Ming Hsuan Yang. Adversarial learning for semi-supervised semantic segmentation. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2019.

[HW79] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 1979.

[HZG20] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing*, 2020.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[JCO+17] Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled

neural interfaces using synthetic gradients. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[JCP+24] Junguang Jiang, Baixu Chen, Junwei Pan, Ximei Wang, Dapeng Liu, Jie Jiang, and Mingsheng Long. Forkmerge: Mitigating negative transfer in auxiliary-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[JCS+22] Woojeong Jin, Yu Cheng, Yelong Shen, Weizhu Chen, and Xiang Ren. A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2022.

[JD21] Stephen James and Andrew J Davison. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 2021.

[JDJ17] Stephen James, Andrew J Davison, and Edward Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. *Conference on Robot Learning (CoRL)*, 2017.

[JGB+21] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[JMAD20] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[JMC+17] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[JV22] Adrián Javaloy and Isabel Valera. Rotograd: Dynamic gradient homogenization for multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[JYX+21] 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 *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KDPK+22] Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and M Pawan Kumar. In defense of the unitary scalarization for deep multi-task learning. *arXiv preprint arXiv:2201.04122*, 2022.

[KFF15] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[KGBN+15] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

[KGC18] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[KGHD19] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[KMA+18] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2018.

[KMHK20] Chia-Wen Kuo, Chih-Yao Ma, Jia-Bin Huang, and Zsolt Kira. Featmatch: Feature-based augmentation for semi-supervised learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[KMR+23] 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. 2023.

[Kok17] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[KPR+17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 2017.

[KQL+20] Zhanghan Ke, Di Qiu, Kaican Li, Qiong Yan, and Rynson W.H. Lau. Guided collaborative training for pixel-wise semi-supervised learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[KRA+20] 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. *International Journal of Computer Vision (IJCV)*, 2020.

[Kri09] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[KS+20] Jean Kaddour, Steindór Sæmundsson, et al. Probabilistic active meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[KSK21] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[KTW+20] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[KWHG20] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[KZG+17] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision (IJCV)*, 2017.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[LBKH19] Xingyu Lin, Harjatin Baweja, George Kantor, and David Held. Adaptive auxiliary task weighting for reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[LBPL19] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[LCW18] Wei Liu, Chaofeng Chen, and Kwan-Yee Wong. Char-net: A character-aware neural network for distorted scene text recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2018.

[LCY+23] Huan Liu, Zhixiang Chi, Yuanhao Yu, Yang Wang, Jun Chen, and Jingshan Tang. Meta-auxiliary learning for future depth prediction in videos. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[LCZ+22a] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[LCZ+22b] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[LDJ19] Shikun Liu, Andrew J Davison, and Edward Johns. Self-supervised generalisation with meta auxiliary learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[LDSL21] Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[LEC23] Fangyu Liu, Guy Edward Toh Emerson, and Nigel Collier. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 2023.

[LFDA16] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 2016.

[LFJ+24] Shikun Liu, Linxi Fan, Edward Johns, Zhiding Yu, Chaowei Xiao, and Anima Anandkumar. Prismer: A vision-language model with multi-task experts. *Transactions on Machine Laerning Research (TMLR)*, 2024.

[LFSL24] Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[LGG+17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[LGRN09] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[LH19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[li220] What does BERT with vision look at?", author = "li, liunian harold and yatskar, mark and yin, da and hsieh, cho-jui and chang, kai-wei. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2020.

[LJD19] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[LJDJ22] Shikun Liu, Stephen James, Andrew J Davison, and Edward Johns. Auto-lambda: Disentangling dynamic task relationships. *Transactions on Machine Laerning Research (TMLR)*, 2022.

[LK18] Lukas Liebel and Marco Körner. Auxiliary tasks in multi-task learning. *arXiv preprint arXiv:1805.06334*, 2018.

[LLJ+21] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multitask learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[LLK+21] Liyang Liu, Yi Li, Zhanghui Kuang, J Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[LLSH23] 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 *Proceedings of the International Conference on Machine Learning (ICML)*, 2023.

[LLWT15] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[LLXH22] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

[LMB+14] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[LML+23] Ximing Li, Chen Ma, Guozheng Li, Peng Xu, Chi Harold Liu, Ye Yuan, and Guoren Wang. Meta auxiliary learning for top-k recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[LOG+19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[LQZ+22] Jiangmeng Li, Wenwen Qiang, Changwen Zheng, Bing Su, and Hui Xiong. Metaug: Contrastive learning via meta feature augmentation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[LSG+21] 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. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[LSY19] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[LWS+20] Chenghao Liu, Zhihao Wang, Doyen Sahoo, Yuan Fang, Kun Zhang, and Steven C. H. Hoi. Adaptive task sampling for meta-learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[LXW21] Xianpeng Liu, Nan Xue, and Tianfu Wu. Learning auxiliary monocular contexts helps monocular 3d object detection. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2021.

[LYH16] Giwoong Lee, Eunho Yang, and Sung Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

[LYH18] Hae Beom Lee, Eunho Yang, and Sung Ju Hwang. Deep asymmetric multi-task feature learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[LYL+20] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[LYZT22] Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor W. Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. 2022.

[LZCL17] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[LZJD21] Shikun Liu, Shuaifeng Zhi, Edward Johns, and Andrew J Davison. Bootstrapping semantic segmentation with regional contrast. *arXiv preprint arXiv:2104.04465*, 2021.

[LZL+19] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[MDA15] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

[ME14] Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *The Artificial Intelligence Review*, 2014.

[MHK14] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 2014.

[MMFY21] Yue Ming, Xuyang Meng, Chunxiao Fan, and Hui Yu. Deep learning for monocular depth estimation: A review. *Neurocomputing*, 2021.

[MMKI18] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2018.

[MRK19] Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[MRY21] Paul Michel, Sebastian Ruder, and Dani Yogatama. Balancing average and worst-case accuracy in multitask learning. *arXiv preprint arXiv:2110.05838*, 2021.

[MSGH16] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[MTB19] Sudhanshu Mittal, Maxim Tatarchenko, and Thomas Brox. Semi-supervised semantic segmentation with high-and low-level consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2019.

[MVS22] Aakarsh Malhotra, Mayank Vatsa, and Richa Singh. Dropped scheduled task: Mitigating negative transfer in multi-task learning using dynamic task dropping. *Transactions on Machine Laerning Research (TMLR)*, 2022.

[NAM+21] Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary learning by implicit differentiation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[NAS18] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[NC18] Hien D Nguyen and Faicel Chamroukhi. Practical and theoretical aspects of mixture-of-experts modeling: An overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2018.

[NORBK17] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[NS18] Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2018.

[NSA+22] Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *ICML*, 2022.

[NSF12] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[NSFC20] Aviv Navon, Aviv Shamsian, Ethan Fetaya, and Gal Chechik. Learning the pareto front with hypernetworks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[OHT20] Yassine Ouali, Céline Hudelot, and Myriam Tami. Semi-supervised semantic segmentation with cross-consistency training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[OKB11] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[OPAB+20] Pedro O O. Pinheiro, Amjad Almahairi, Ryan Benmalek, Florian Golemo, and Aaron C Courville. Unsupervised learning of dense visual representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[Ope23] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[OTPS21] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[PRP+20] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[PRS20] Xavier Soria Poma, Edgar Riba, and Angel Sappa. Dense extreme inception network: Towards a robust cnn model for edge detection. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[Ray23] Partha Pratim Ray. Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 2023.

[RBB+23] Noam Rotstein, David Bensaid, Shaked Brody, Roy Ganz, and Ron Kimmel. Fusecap: Leveraging large language models to fuse visual data into enriched image captions. *arXiv preprint arXiv:2305.17718*, 2023.

[RBC+21] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[RBK21] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015.

[RKH+21] 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 *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[RKH22] Aneesh Rangnekar, Christopher Kanan, and Matthew Hoffman. Semantic segmentation with active semi-supervised representation learning. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2022.

[RKR18] Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[RL16] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[RPM+21] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[RRRH20] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20:*

International Conference for High Performance Computing, Networking, Storage and Analysis, 2020.

[RVY14] Veselin Raychev, Martin Vechev, and Eran Yahav. Code completion with statistical language models. In *Proceedings of the 35th ACM SIGPLAN conference on programming language design and implementation*, 2014.

[RW19] Alec Radford and Jeffrey Wu. Rewon child, david luan, dario amodei, and ilya sutskever. 2019. *Language models are unsupervised multitask learners. OpenAI blog*, 2019.

[SBB+16] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

[SBBD23] Samuel L Smith, Andrew Brock, Leonard Berrada, and Soham De. Convnets match vision transformers at scale. *arXiv preprint arXiv:2310.16764*, 2023.

[SBC+20] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[SBV+22] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[Sch92] Jürgen Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 1992.

[SCL12] Pierre Sermanet, Soumith Chintala, and Yann LeCun. Convolutional neural networks applied to house numbers digit classification. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2012.

[SDGS18] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2018.

[SFA+22] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

[SFH+23] Mohammadreza Salehi, Mehrdad Farajtabar, Maxwell Horton, Fartash Faghri, Hadi Pouransari, Raviteja Vemulapalli, Oncel Tuzel, Ali Farhadi, Mohammad Rastegari, and Sachin Mehta. Clip meets model zoo experts: Pseudo-supervision for visual enhancement. *arXiv preprint arXiv:2310.14108*, 2023.

[SK18] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[SLO20] Siyuan Shan, Yang Li, and Junier B Oliva. Meta-neighborhoods. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[SLX15] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[SLZ+19] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019.

[SMF23] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[SML+21] David So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V Le. Searching for efficient transformers for language modeling. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[SMV23] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[SNG+23] Aviv Shamsian, Aviv Navon, Neta Glazer, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Auxiliary learning as an asymmetric bargaining game. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2023.

[SNS+19] Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[SP97] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *Proceedings of Signal Processing: Image Communication (SPIC)*, 1997.

[SPFS20] Ximeng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. Adashare: Learning what to share for efficient deep multi-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[SPKK23] Dmitry Senushkin, Nikolay Patakin, Arseny Kuznetsov, and Anton Konushin. Independent component alignment for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[SRZP23] Haosen Shi, Shen Ren, Tianwei Zhang, and Sinno Jialin Pan. Deep multi-task learning with progressive parameter sharing. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[SSZ17] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[Sta20] Felix Stahlberg. Neural machine translation: A review. *Journal of Artificial Intelligence Research*, 2020.

[SWR+22] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma,

Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. Multitask prompted training enables zero-shot task generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[SZC+20a] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[SZC+20b] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[SZS12] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[TB19] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[Tes22] Tesla, Oct 2022.

[THK+21] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[TMC+21] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language

models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[TSM+21] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michael Mathieu, et al. Open-ended learning leads to generally capable agents. *arXiv preprint arXiv:2107.12808*, 2021.

[TTLL17] Shubham Toshniwal, Hao Tang, Liang Lu, and Karen Livescu. Multitask learning with low-level auxiliary tasks for encoder-decoder based speech recognition. *Proc. Interspeech 2017*, 2017.

[TV17] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[VBL+16] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[VD02] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 2002.

[VdMH08] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, 2008.

[VGDBVG20] Simon Vandenhende, Stamatios Georgoulis, Bert De Brabandere, and Luc Van Gool. Branched multi-task networks: deciding what layers to share. *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.

[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 2020.

[VGVG20] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[VGVG+21] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021.

[VLZP15] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[WBD+22] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022.

[WFL+24] Jiawei Wu, Haoyi Fan, Zuoyong Li, Guanghai Liu, and Shouying Lin. Information transfer in semi-supervised semantic segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[WLCS19] Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O Stanley. Poet: open-ended coevolution of environments and their optimized solutions. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2019.

[WLY+23] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2311.03079*, 2023.

[WTB+22] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Laerning Research (TMLR)*, 2022.

[WWS+22] Yuchao Wang, Haochen Wang, Yujun Shen, Jingjing Fei, Wei Li, Guoqiang Jin, Liwei Wu, Rui Zhao, and Xinyi Le. Semi-supervised semantic

segmentation using unreliable pseudo-labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[WXC+21] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[WYH+22] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *Transactions on Machine Laerning Research (TMLR)*, 2022.

[WYM+22] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

[WYY+21] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[WZL21] Haoxiang Wang, Han Zhao, and Bo Li. Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[WZS+21] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[WZY+21] Wenguan Wang, Tianfei Zhou, Fisher Yu, Jifeng Dai, Ender Konukoglu, and Luc Van Gool. Exploring cross-image pixel contrast for semantic segmentation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[XOWS18] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth

estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[XSM+21] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[XWZ+23] Haoyu Xie, Changqi Wang, Mingkai Zheng, Minjing Dong, Shan You, Chong Fu, and Chang Xu. Boosting semi-supervised semantic segmentation with probabilistic representations. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2023.

[YCK92] Roni Yagel, Daniel Cohen, and Arie Kaufman. Normal estimation in 3d discrete space. *The visual computer*, 1992.

[YDM+24] Chenyu You, Weicheng Dai, Yifei Min, Fenglin Liu, David Clifton, S Kevin Zhou, Lawrence Staib, and James Duncan. Rethinking semi-supervised medical image segmentation: A variance-reduction perspective. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[YGL+21] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[YH17] Yongxin Yang and Timothy Hospedales. Trace norm regularised deep multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[YKG+20] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[YKZ21] Teresa Yeo, Oğuzhan Fatih Kar, and Amir Zamir. Robustness via cross-domain ensembles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[YLY+21] Feiyang Ye, Baijiong Lin, Zhixiong Yue, Pengxin Guo, Qiao Xiao, and Yu Zhang. Multi-objective meta learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[YWV+22] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *Transactions on Machine Laerning Research (TMLR)*, 2022.

[YX22a] Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[YX22b] Hanrong Ye and Dan Xu. Taskprompter: Spatial-channel multi-task prompting for dense scene understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[YYB24] Shoubin Yu, Jaehong Yoon, and Mohit Bansal. Crema: Multimodal compositional video reasoning via efficient modular adaptation and fusion. *arXiv preprint arXiv:2402.05889*, 2024.

[YYD+23] Zhujun Yang, Zhiyuan Yan, Wenhui Diao, Qian Zhang, Yuzhuo Kang, Junxi Li, Xinming Li, and Xian Sun. Label propagation and contrastive regularization for semisupervised semantic segmentation of remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[ZBSL17] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1851–1858, 2017.

[ZKK22] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple multi-dataset detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[ZLG22] Lijun Zhang, Xiao Liu, and Hui Guan. Automtl: A programming framework for automating efficient multi-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[ZLH+21] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[ZQD+20] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 2020.

[ZRG+22] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[ZRH+19] Philipp Zech, Erwan Renaudo, Simon Haller, Xiang Zhang, and Justus Piater. Action representations in robotics: A taxonomy and systematic classification. *The International Journal of Robotics Research*, 2019.

[ZSC+20] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[ZSS+18] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[ZTJ18] Yabin Zhang, Hui Tang, and Kui Jia. Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[ZTRR21] Feihu Zhang, Philip Torr, Rene Ranftl, and Stephan R Richter. Looking beyond single images for contrastive semantic segmentation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[ZVM+21] Xiangyun Zhao, Raviteja Vemulapalli, Philip Andrew Mansfield, Boqing Gong, Bradley Green, Lior Shapira, and Ying Wu. Contrastive learning for label efficient semantic segmentation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[ZWM+22] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[ZWP22] Yuhan Zheng, Fuping Wu, and Bartfblomiej W Papież. An ensemble method to automatically grade diabetic retinopathy with optical coherence tomography angiography images. In *MICCAI Challenge on Mitosis Domain Generalization*. 2022.

[ZWW+22] Andy Zeng, Adrian Wong, Stefan Welker, Krzysztof Choromanski, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[ZY18] Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 2018.

[ZYKW18] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[ZYL+19] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019.

[ZZL+22] Xizhou Zhu, Jinguo Zhu, Hao Li, Xiaoshi Wu, Hongsheng Li, Xiaohua Wang, and Jifeng Dai. Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[ZZW+21] Yi Zhu, Zhongyue Zhang, Chongruo Wu, Zhi Zhang, Tong He, Hang Zhang, R Manmatha, Mu Li, and Alexander Smola. Improving semantic segmentation via self-training. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021.

[ZZW+23] Weixia Zhang, Guangtao Zhai, Ying Wei, Xiaokang Yang, and Kede Ma. Blind image quality assessment via vision-language correspondence: A multitask learning perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[ZZXW19] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 2019.

[ZZZ+21] Yuliang Zou, Zizhao Zhang, Han Zhang, Chun-Liang Li, Xiao Bian, Jia-Bin Huang, and Tomas Pfister. Pseudoseg: Designing pseudo labels for semantic segmentation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.