UNIVERSITÀ DI TRENTO
Department of Information Engineering and Computer Science

Doctoral School in Information and Communication Technology

Efficient Knowledge Transfer and Adaptation for Speech and Beyond

Umberto Cappellazzo

Advisor
Daniele Falavigna
Fondazione Bruno Kessler

Co-Advisor
Alessio Brutti
Fondazione Bruno Kessler

November 2024

Abstract

This thesis advances the field of efficient knowledge transfer and adaptation in the realm of speech processing. It is structured to address the limitations of transfer learning in dynamically evolving audio and speech processing contexts, particularly through novel approaches for class-incremental learning, parameter-efficient adaptation, and multimodal modeling. First, we provide a comprehensive framework for class-incremental spoken language understanding, allowing models to incrementally learn new intents and entities while retaining previously acquired knowledge. Using knowledge distillation and rehearsal-based strategies, we enhance robustness against catastrophic forgetting, a key limitation in continual learning. We also introduce a unique approach to class-incremental audio classification, utilizing mutual information optimization. Second, given the prohibitive computational costs of traditional model adaptation (i.e., full fine-tuning), this thesis introduces a comprehensive framework for parameter-efficient fine-tuning of audio and speech foundation models. Furthermore, we propose new adapter designs to achieve effective transfer learning with minimal computational overhead. Finally, extending beyond unimodal settings, we propose `Llama-AVSR`, a new multimodal large language model with strong audio-visual speech recognition (AVSR) abilities. `Llama-AVSR` leverages pre-trained audio and video encoders along with a large language model to achieve state-of-the-art accuracy on the major AVSR benchmark. Notably, this model adapts pre-trained language models to the multimodal AVSR domain while keeping the core model parameters frozen, ensuring computational efficiency. Overall, this thesis provides a comprehensive framework and empirical findings that advance the application of continual learning, efficient fine-tuning, and multimodal models for speech processing tasks. These contributions pave the way for adaptive, resource-efficient speech understanding systems.

Keywords

[Transfer Learning, Class-Incremental Continual Learning, Spoken Language Understanding, Parameter-Efficient Transfer Learning, MultiModal Large Language Models, Audio-Visual Speech Recognition]

Contents

List of Tables

List of Figures

List of Abbreviations

ASC Acoustic Scene Classification.

ASR Automatic Speech Recognition.

AST Audio Spectrogram Transformer.

AVSR Audio-Visual Speech Recognition.

CIL Class-Incremental Learning.

E2E end-to-end.

FSC Fluent Speech Commands.

KD Knowledge Distillation.

LLM Large Language Model.

MI Mutual Information.

MLLM Multimodal Large Language Model.

MoA Mixture of Adapters.

MoE Mixture of Experts.

NAS Neural Architectural Search.

PETL Parameter-Efficient Transfer Learning.

seq2seq sequence-to-sequence.

SLU Spoken Language Understanding.

SLURP Spoken Language Understanding Resource Package.

VSR Video Speech Recognition.

Chapter 1

Introduction

1.1 The Transfer Learning Paradigm

Transfer Learning [267] has emerged as a powerful technique in machine learning, particularly in domains like natural language processing [23, 186], computer vision [52, 60], speech processing [222], and reinforcement learning [183]. This approach involves training a model on a large, diverse dataset to learn general representations. Then, this pre-trained model can be adapted to new, specific tasks using a relatively small amount of task-specific data. This process, known as *fine-tuning*, leverages the knowledge gained from the pre-training phase to improve the model's performance on the target task(s). Usually, the model's parameters are entirely updated (i.e., *full fine-tuning*, see Figure 1.1).

The transfer learning paradigm comes with several benefits. One of the primary advantages is its ability to reduce the amount of data necessary to train a new model. By using a pre-trained network as a foundation, the model already possesses a fundamental understanding of the underlying patterns within the data. This enables the model to learn more rapidly and efficiently with less data, making it feasible to apply neural networks to smaller datasets. Another advantage of transfer learning is its ability to mitigate the risk of overfitting. Overfitting occurs when a model becomes overly specialized to the training data, leading to poor performance on un-

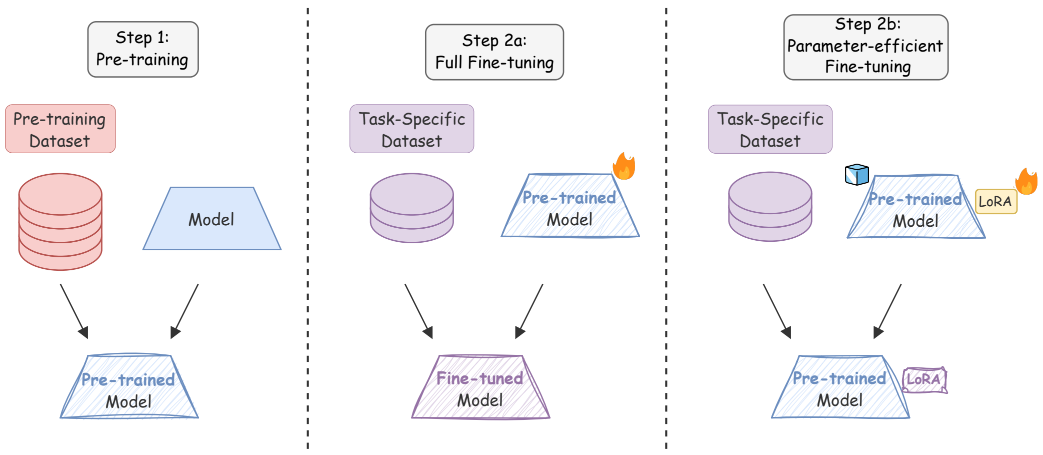


Figure 1.1: During stage 1, the model is pre-trained on a massive dataset, often unlabeled. Subsequently, traditional (full) *fine-tuning* involves updating all the pre-trained model's parameters using a smaller, task-specific dataset. This process results in a specialized model but can be computationally expensive and prone to overfitting. In contrast, *parameter-efficient fine-tuning* freezes the pre-trained model and introduces lightweight modules (LoRA in this example [101]) that are trained specifically for each downstream task. This thesis delves into this efficient approach.

seen data. By utilizing a pre-trained network, the model has already been exposed to a vast dataset, allowing it to develop a strong ability to generalize to diverse data. Additionally, transfer learning can accelerate the training process and diminish the computational cost. The pre-trained network offers a solid starting point that can guide the model to converge toward a good solution more rapidly, thereby lessening the time and resources necessary to train a new model.

Despite its successes, this paradigm for transfer learning faces several limitations in various contexts. Firstly, in multi-task fine-tuning, the learning signals from different tasks can negatively interfere with each other [159]. Similarly, in continual learning, the model tends to forget previously learned knowledge when adapting to new examples [69, 204]. Secondly, when the training and evaluation data distributions differ, the model may struggle to generalize effectively [103, 122]. This makes the model fragile and less accurate, hindering its deployment in real-world scenarios where distribution shifts are common. Moreover, updating all the pre-trained model's parameters can be very expensive and unfeasible, mainly when dealing with huge pre-trained models and many downstream tasks, requiring a copy of the model to be stored in the memory for each task.

Given these limitations, simply fine-tuning the entire model is impractical and often ineffective in multiple scenarios. Therefore, targeted approaches are necessary to address these specific challenges. In this direction, this thesis aims to develop efficient methods for conveying the knowledge and information encapsulated in pre-trained models to audio and speech downstream tasks. As current research in this area is limited, there is a clear need for further investigation. Specifically, this thesis will explore the following research questions:

(Q1) *In a continual learning scenario, fine-tuning a pre-trained model continuously results in complete catastrophic forgetting of its past knowledge. How can we develop effective continual learning strategies to prevent catastrophic forgetting in pre-trained models, allowing them to learn new tasks without compromising their existing knowledge?*

(Q2) *When dealing with numerous downstream tasks, adapting the pre-trained model for each downstream task is unfeasible and computationally expensive. How can we efficiently adapt pre-trained audio and video foundation models to various downstream tasks using parameter-efficient fine-tuning techniques, minimizing computational costs and preserving model performance?*

(Q3) *Given the exceptional multimodal understanding and reasoning capabilities of pre-trained Large Language Models, how can we efficiently leverage them to enhance generative tasks like audio, visual, and audio-visual speech recognition?*

This thesis aims to systematically investigate and address these research questions, providing a comprehensive study on efficiently transferring and adapting knowledge from pre-trained models to challenging, real-world scenarios.

1.2 Contributions

This thesis tackles the problem of how to transfer the knowledge of pre-trained models to multiple tasks in an efficient and catastrophic forgetting-free way for audio and speech processing. This thesis bridges gaps in important areas of audio and speech processing that received minimal or no attention compared to other fields. We start by studying spoken language understanding and acoustic scene classification through the lens of class-incremental learning, where different tasks are defined by partitioning the dataset classes into disjoint subsets. In this way, we try to adapt a single model to incorporate new concepts over time whilst retaining the past knowledge, avoiding the catastrophic forgetting issue. We provide ad-hoc strategies based on knowledge distillation and rehearsal paradigms to endow the models with incremental-learning capabilities. On top of this, we frame the problem of spoken language understanding as a sequence-to-sequence problem where the model generates the outputs in an auto-regressive fashion. This formulation allows us to investigate different methods to mitigate forgetting at both the multimodal encoders level and within the ASR decoder.

We then move our attention to the problem of parameter-efficient transfer learning of pre-trained audio and speech models, which aims to dispense with the expensive paradigm of updating all the parameters of the pre-trained model in favor of more a resource-efficient one. Whereas much attention has been devoted to this line of research in the natural language processing and vision domains, the speech and audio domains have seen slower progress. To address this gap, we provide a framework whereby we study several parameter-efficient transfer learning methods for the efficient fine-tuning of audio and speech foundation models. Furthermore, we propose: 1) a new adapter design that builds on the Conformer architecture to enhance performance, and 2) the use of the mixture of experts paradigm to tailor different adapter modules to specific input data.

Finally, given the strong multimodal capabilities of large language models, we study how to leverage powerful LLMs (e.g., Llama 3.1) to carry out the tasks of audio, visual, and audio-visual speech recognition. This leads to the development of Llama-AVSR, a novel multimodal large language model with strong audio-visual speech recognition capabilities. Llama-AVSR leverages pre-trained audio and video encoders, as well as a large language model to carry out multiple tasks, establishing new state-of-the-art results on the largest public audio-visual speech recognition benchmark.

Overall, our main contributions can be summarized as follows:

1. **Class-Incremental Learning for Spoken Language Understanding and Audio Classification**. We study the *classification* tasks of spoken language understanding (SLU) (i.e., intent classification) and audio classification from the class-incremental learning perspective. We present a detailed analysis of the optimal combination of rehearsal and knowledge distillation for the task of class-incremental spoken language understanding. Besides, for audio classification, we propose a novel approach that leverages mutual information optimization to enhance both the training process and memory selection.

2. **Class-Incremental Sequence-to-Sequence Spoken Language Understanding**. We treat the task of SLU as a sequence-to-sequence task where the intent labels are generated along with the ASR transcriptions in an auto-regressive way. We study how to mitigate catastrophic forgetting on the ASR decoder side via a sequence-level knowledge distillation loss, Seq-KD [32], as well as on the encoders space through multimodal contrastive distillation objectives, COCONUT [30]. We also show that employing both Seq-KD and COCONUT methods leads to the best results.

3. **Parameter-Efficient Transfer Learning of Audio and Speech Foundation Models**. We provide a framework whereby we study several parameter-efficient transfer learning methods for the efficient fine-tuning of audio and speech foundation models. Furthermore, we

propose: 1) a new adapter design that builds on the Conformer architecture to boost performance, and 2) the use of the mixture of experts paradigm to specialize different adapter modules to specific input data.

4. **A New State-of-the-art Multimodal Large Language Model for Audio-Visual Speech Recognition**. We propose `Llama-AVSR`, a new multimodal large language model with strong audio-visual speech recognition capabilities. It leverages pre-trained audio and video encoders, as well as a large language model (i.e., LLama 3.1 8B [64]) to perform automatic speech recognition, visual speech recognition, and audio-visual speech recognition. We establish new state-of-the-art results on the largest public audio-visual speech recognition benchmark.

1.3 A Note on the Notion of "Task"

In this thesis, the term "**task**" can have a different meaning based on the context it is used in. In class-incremental learning, with task we refer to a specific training stage in which the model is learning a new set of classes. The number of tasks depends on how we partition the dataset's classes. For example, if a dataset has 30 classes, we could assign 3 classes to each task, resulting in a sequence of 10 tasks. The model is trained sequentially on each of these tasks and must be able to integrate the knowledge of the new classes included in the current task whilst retaining the knowledge from the previous tasks. This reasoning applies to Chapter 2 of this thesis.

In Chapters 3 and 4, instead, "**task**" denotes the usual connotation of downstream task, on which the pre-trained model is tested to assess its generalization abilities. Usually, unlike in class-incremental learning, the model is not required to perform well on the pre-training tasks it has been trained on. To minimize the confusion throughout this thesis, we will specify "downstream task" in Chapters 3 and 4.

1.4 Publications

This doctoral thesis is supported by a series of publications that showcase the research contributions made in the fields of audio, speech, and audio-visual speech processing, with a particular emphasis on continual learning for spoken language understanding and speech recognition, parameter-efficient transfer learning of audio and speech foundation models, and multimodal large language models for audio-visual speech recognition. Below is a list of these publications in chronological order.

1. *An Investigation of the Combination of Rehearsal and Knowledge Distillation in Continual Learning for Spoken Language Understanding*, **U. Cappellazzo**, D. Falavigna, A. Brutti. Proceedings of Interspeech, 2023 [27]. The code is available here.

2. *Sequence-level Knowledge Distillation for Class-incremental End-to-end Spoken Language Understanding*, **U. Cappellazzo**, M. Yang, D. Falavigna, A. Brutti. Proceedings of Interspeech, 2023 [32].

3. *Improving Continual Learning of Acoustic Scene Classification via Mutual Information Optimization*, M. Yang, **U. Cappellazzo**, X. Li, B. Raj. Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2024 [239].

4. *Training Early-Exit Architectures for Automatic Speech Recognition: Fine-Tuning Pre-Trained Models or Training from Scratch*, G. A. Wright, **U. Cappellazzo**, S. Zaiem, D. Raj, L. Ondel Yang, D. Falavigna, M. Nabih, A. Brutti. Proceedings of the ICASSP Self-supervision in Audio, Speech and Beyond (SASB) workshop, 2024 [230].

5. *Continual Contrastive Spoken Language Understanding*, **U. Cappellazzo**, E. Fini, M. Yang, D. Falavigna, A. Brutti, B. Raj. Proceedings of ACL (Findings), 2024 [30].

6. *Parameter-Efficient Transfer Learning of Audio Spectrogram Transformers*, **U. Cappellazzo**, D. Falavigna, A. Brutti, M. Ravanelli. Pro-

ceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 2024 [29]. The code is available here.

7. *Efficient Fine-tuning of Audio Spectrogram Transformers via Soft Mixture of Adapters*, **U. Cappellazzo**, D. Falavigna, A. Brutti. Proceedings of Interspeech, 2024 [28]. The code is available here.

8. *Evaluating and Improving Continual Learning in Spoken Language Understanding*, M. Yang, X. Li, **U. Cappellazzo**, S. Watanabe, B. Raj. Proceedings of Interspeech, 2024 [241].

9. *Large Language Models Are Strong Audio-Visual Speech Recognition Learners*, **U. Cappellazzo**, M. Kim, H. Chen, P. Ma, S. Petridis, D. Falavigna, A. Brutti, M. Pantic. To appear at ICASSP 2025 [31].

Chapter 2

Class-Incremental Spoken Language Understanding

This chapter investigates the problem of spoken language understanding (SLU) from a class-incremental learning perspective. Our objective is to endow a single model with lifelong learning capabilities such that it can deal with a stream of emerging concepts or classes (in our case *intents* or *sounds*) over a (in)finite time horizon. This paradigm goes beyond the unrealistic and brittle assumption that the data distribution the model will face after deployment aligns with what it encountered during the training phase. The model must be able to incorporate fresh new information while consolidating and transferring forward the knowledge acquired in the previous tasks. We first formulate the problem of SLU as a standard classification problem and then as a more complex sequence-to-sequence problem where the model generates the outputs in an auto-regressive manner, and we present multiple techniques to attenuate the issue of catastrophic forgetting (i.e., the natural inclination of neural networks to erase the past knowledge in favor of new information). In addition to this, we propose a novel class-incremental learning method for the problem of audio classification by improving both the modeling and memory selection mechanism via mutual information optimization.

We organize this chapter as follows.

- We first discuss the problem of continual learning and its multiple settings.

- We then formulate mathematically the class-incremental learning setting and provide a taxonomy of recent state-of-the-art methods that tackle it.
- We describe in details three methods to mitigate catastrophic forgetting for SLU and one for audio classification, which all of them have been accepted for publications in top-notch conferences (e.g., *Interspeech* x2, *ICASSP*, *ACL*).

2.1 Continual Learning

Intelligent systems rely on learning as a foundation to adapt to dynamic environments. Throughout evolution, humans and other organisms have developed the remarkable ability to continually acquire, update, and utilize knowledge in response to external changes [120, 169]. Inspired by the human capacity for adaptation, AI systems must also be able to deal with real-world dynamics. This is the core focus of ***continual learning***, where systems learn a sequence of tasks incrementally, but perform as if they had learned all tasks simultaneously. These tasks might include new skills, updated versions of existing skills, or adjustments to different environments and contexts, all while addressing realistic problems [82, 213, 218]. Given this lifelong learning process, continual learning is often referred to as incremental or lifelong learning in the literature, with the terms used interchangeably in most cases [46].

Conventional machine learning models are typically designed to work with static data distributions, but continual learning involves adapting to ever-changing distributions. A key issue in this process is **catastrophic forgetting** [159], where learning a new distribution significantly impairs the system's ability to retain knowledge of previous ones. This challenge highlights the trade-off between *plasticity* (the ability to learn new information) and *stability* (the ability to preserve past knowledge) [78]: too much emphasis on one tends to weaken the other. However, beyond just finding a balance between these two, an effective continual learning approach should also focus

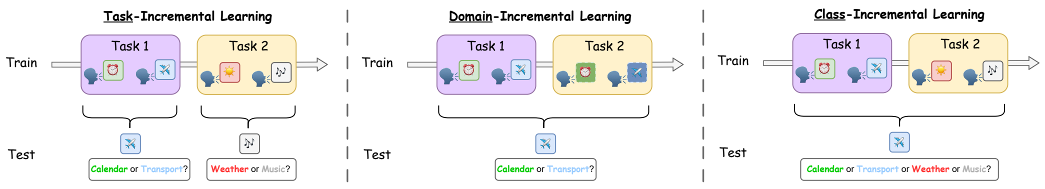


Figure 2.1: The setting of Task-Incremental Learning (TIL), Domain-Incremental Learning (DIL), and Class-Incremental Learning (CIL). CIL and TIL share the same training protocol, yet TIL is much easier during inference as it is required to only classify among the corresponding label spaces. DIL refers to the data stream with distribution change, where new tasks contain the same classes from different domains.

on achieving strong generalization—enabling the model to handle variations both within tasks and across different tasks.

Continual learning encompasses multiple scenarios based on how we define the tasks and the availability of task labels. The most common ones are depicted in Figure 2.1 in increasing complexity order. In a **class-incremental learning** (CIL) scenario (Figure 2.1, right), training data emerge sequentially over time. In each timestamp, the model learns from a new task, which comprises a subset of new classes from the original training dataset. For instance, during task one the model has access only to the intent classes "calendar" and "transport", and during the second task to the intent classes "weather" and "music", and so on. Note that each intent class can be included in one and only task, so the tasks correspond to non-overlapping subsets of classes. At test time, the model is tested on all seen classes to determine if it can still accurately distinguish between them. A good model is expected to balance learning the characteristics of new classes while maintaining knowledge of previously learned ones (i.e., plasticity-stability dilemma [78]). In addition to CIL, Task-Incremental Learning (TIL) and Domain-Incremental Learning (DIL) are two other popular settings to study the lifelong capabilities of a model [213] (Figure 2.1, left and middle). TIL is the same as CIL during the training phase, where new classes appear at each new task. However, there is a key difference at test time: whereas CIL requires the model to differentiate among all the classes seen so far, TIL relaxes this constraint by providing the task labels of each test data, thus the model needs to classify only among the classes present in that specific task. Therefore, the model is required to perform inter-task class discrimination in CIL, whereas for TIL it performs intra-task class discrimination, thus making TIL an easier problem than CIL since usually it is possible to train models with task-specific components. Finally, the DIL protocol concentrates on the scenario with concept drift or distribution change, where new tasks comprise data from different domains but with the same label space.

We finally note that other continual learning protocols have been proposed and studied in the literature, such as *online continual learning* and *continual pre-training*, showcasing the multiple facets that characterize this paradigm. However, given its higher complexity, the focus of this chapter and this thesis is on class-incremental learning. We refer the reader to [218] for an exhaustive description of CL protocols.

2.2 Class-Incremental Learning: Problem Formulation and Taxonomy

2.2.1 Problem Formulation

Class-Incremental Learning involves learning from a continuously-evolving stream of new classes. We suppose there is a sequence of K training tasks $\{\mathcal{D}_1, \dots, \mathcal{D}_K\}$, where each task introduces a new dataset $\mathcal{D}_k$, comprising n_k instances $\{(x_k^i, y_k^i)\}_{i=1}^{n_k}$. Here, x_k^i is an instance of class $y_k^i \in Y_k$ (e.g., speech or audio signals). The label spaces for different tasks are **disjoint**, i.e., $Y_k \cap Y_{k'} = \emptyset$ for $k \neq k'$. At any given time, the model can only access data from the current task $\mathcal{D}_k$. The objective of CIL is to develop a model that not only learns from the current task $\mathcal{D}_k$ but also retains knowledge from all previously learned tasks. At the end of each task, the model is evaluated on the union of all classes seen so far $\mathcal{Y} = Y_1 \cup \dots \cup Y_K$. The goal is to minimize the expected risk over all tasks, fitting a model $f(x) : \mathcal{X} \to \mathcal{Y}$ that can accurately classify instances from both new and old classes.

2.2.2 Taxonomy on CIL Strategies

Over the last few years, a vast array of continual learning methods has emerged. Consequently, multiple survey papers have tried to classify and elaborate on many CIL methods [54,124,158,169,213,218,261]. It is common practice to categorize CL methods into three macro groups: *regularization*-based, *replay*-based, and *architecture*-based. More fine-grained classifications have been proposed, but since our proposed CIL methods rely on knowledge-distillation and replay concepts, we mainly focus on regularization and replay CIL strategies.

Regularization-based methods introduce ad-hoc regularization terms to contrast catastrophic forgetting. Some methods regularize the weights of the network [2, 7, 117, 171, 189, 217], whereas others penalize changes to the model's intermediate or final outputs, usually by means of the *knowledge distillation* (KD) concept [94]. By denoting the model trained in the previous task as the *teacher* model and that trained in the current task as the *student* model, KD fosters the transfer of the knowledge accrued in the teacher model onto the student. Given the large number of proposed methods that exploit the KD concept, we can identify three sub-categories of KD-based methods. *Logit* distillation methods, the first to be proposed and the most popular ones, align the probability distributions over the classes of the student and teacher models [132, 182, 232]. In this way, they constrain the old and new models to share the same semantic relationship. *Feature* distillation approaches, instead, operate on the intermediate feature embeddings produced by the feature encoders such that those produced by the new models resemble the ones obtained with the teacher model [61,96,109,110]. Finally, we point out that logit and feature distillations can be combined together to provide a double-level distillation regularization and improve performance, as we will show in the next section in one of our proposed methods [27].

Replay-based methods, or **rehearsal**-based (we will use rehearsal and replay terms interchangeably throughout this dissertation), interleave the new data with cherry-picked samples from the previous tasks such that the model can revisit former data. This approach, despite being simple and intuitive, represents a strong baseline and sparked a lot of interest in how to select the best candidates to be retained in the replay buffer. For example, [182] suggests selecting the samples for each class closest to their moving barycenter. In this way, the rehearsal samples are a good representative of each class distribution. On the contrary, some methods propose to choose "hard" examples because they are more informative. For instance, [37] propose to sample exemplars with large prediction entropy and near the decision boundary. [17] estimate data uncertainty through data augmentation and keep the samples with the highest prediction uncertainty. Another line of research incorporates regularization terms with this additional data to steer the optimization process and prevent catastrophic forgetting [38,219,240]. Finally, it is also worth mentioning that some works propose to model the distribution and generate samples rather than storing them in the replay buffer [70,108,198,234]. This is motivated by the fact that storing past data is not always viable due to privacy issues.

Architecture-based methods introduce task-specific parameters, either by expanding the network itself [167,237,262] or keeping the network frozen and learning a small number of additional parameters (i.e., prompts) [162, 199,224,225]. These methods, despite achieving state-of-the-art results, often require expandable memory budgets, which is unfeasible for incremental learning on edge devices and when the number of tasks is big. This is the main reason our proposed methods leverage replay and KD principles rather than architectural methods.

2.3 Class-Incremental Spoken Language Understanding

With the rapid advancement of intelligent voice-enabled personal assistants, Spoken Language Understanding (SLU) has gained significant recognition in recent years [11,179]. Its primary function is to extract key information from spoken utterances, enabling the system to take appropriate actions to meet the user's requests. SLU involves two main tasks [179,211]: 1) *Intent Clas-*

sification, which involves mapping the spoken sentence to its corresponding intent, and 2) *Entity Classification*, also known as Slot Filling, where specific fields of predefined semantic structures are populated with relevant content.

Traditional SLU models follow a cascaded approach, where an automatic speech recognition (ASR) system is first employed, followed by a natural language understanding (NLU) module [95, 164]. In this pipeline, the ASR converts spoken input into text, and the NLU then identifies the target intent from the text. While this method can take advantage of large amounts of ASR and NLU data, it is prone to error propagation from the ASR system. In contrast, end-to-end (E2E) SLU models [5,147,173,188,192] have gained more attention recently, as they use a single trainable model to map speech directly to intent labels, eliminating the need for text transcription and reducing both latency and error propagation.

Most prior research on SLU has concentrated on the conventional independent, identically distributed (i.i.d.) setting, where the entire dataset is available to the model all at once. However, this approach contrasts sharply with real-world situations, where models face significant shifts in data distribution or must adapt to new domains without undergoing full retraining. In such cases, as we have discussed in detail in the previous section, deep models often overwrite previously learned knowledge in favor of the new task, resulting in catastrophic forgetting [159]. While a few studies have explored domain-incremental learning for SLU [165, 195], the more challenging class-incremental learning setting, which is the focus of this chapter and thesis, remains relatively unexplored.

Motivated by the absence of prior research on class-incremental learning for SLU and its potential to address real-world scenarios where new concepts must be incorporated over time, in the next sections we present three key contributions to advance this field.

2.4 An Investigation of the Combination of Rehearsal and Knowledge Distillation in Continual Learning for Spoken Language Understanding [27]

2.4.1 Overview

In this work [27], we consider the problem of intent classification for SLU applied to a class-incremental learning (CIL) scenario, whereby the intents are distributed into several tasks, and the model has to correctly learn them sequentially. To the best of our knowledge, no prior works have studied such a scenario for SLU. To mitigate the issue of catastrophic forgetting, we propose to combine knowledge distillation techniques with replay-based ones. This is buttressed by two main motivations: **1)** their mutual interaction is scarcely investigated for speech as well as for other modalities (e.g., vision), and **2)** the additional use of a distillation loss, which alone falls through in a CIL scenario, is not always beneficial to the model, as pointed out in [20, 158], who contend that it can even lead to a deterioration in the performance.

Therefore, we explore the interplay between applying knowledge distillation (KD) at different levels of the model, specifically at the prediction level and in the feature space, and the rehearsal method. Our findings reveal that combining KD at both the prediction and feature levels yields the best outcomes.

Our contributions can be summarized as follows:

- We define a CIL scenario for SLU over the Fluent Speech Command dataset [147];
- We provide a thorough analysis of the combination of rehearsal and KDs for 4 CL strategies, and we prove its efficacy in our scenario;
- We show that a careful design of the KD weights is crucial for obtaining optimal results:

- We provide an ablation study on the size of the rehearsal buffer, and we conclude that our approach attains larger gains for smaller sizes, thus making it appealing for low-resource devices.

2.4.2 Method

In our CIL setting, a classification model, which comprises a multilayered feature extractor ENC_θ and a classifier FC_ϕ (parameterized by θ and ϕ, respectively), is trained over a sequence of K distinct training phases, that is $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_K\}$. The dataset $\mathcal{D}_k$ related to the k^{th} training phase is interpreted as a task defined by audio signals $\mathcal{X}_k$ and associated intent class labels $\mathcal{Y}_k$, i.e. $\mathcal{D}_k = (\mathcal{X}_k, \mathcal{Y}_k)$. In CIL scenarios all task label sets are mutually exclusive, i.e. $\mathcal{Y}_k \cap \mathcal{Y}_{k'} = \emptyset, k \neq k'$.

At the end of task $k-1$ we select a set of data $\mathcal{R}_{k-1} \subset \mathcal{D}_{k-1}$ for the rehearsal memory. Then, all the rehearsal data, from task 0 to task $k-1$, $\mathcal{R}_0^{k-1} = \{\mathcal{R}_0, \dots, \mathcal{R}_{k-1}\}$ are joined with the training data $\mathcal{D}_k$ in order to train the model for the k^{th} task. A typical rehearsal CL strategy optimizes the CE loss computed over $\mathcal{D}_k \cup \mathcal{R}_0^{k-1}$:

$$\mathcal{L}_{CE}^k = - \sum_{(x,y) \in \mathcal{D}_k \cup \mathcal{R}_0^{k-1}} \log(p[y|x; (\theta_k, \phi_k)]), \tag{2.1}$$

where $p[y|x; (\theta_k, \phi_k)]$ is the output probability distribution of the model given the parameters θ_k and ϕ_k at task k.

We propose to further regularize the model adaptation through a KD loss. We experiment with two different distillation terms in combination with the CE loss. The first one is the Kullback Leibler Divergence (KLD) between the output probability distribution from the current model at task k (usually referred to as the "*student*" model) and the distribution predicted with the model trained at task $k-1$ (the "*teacher*"), so we are operating on the prediction space:

$$\mathcal{L}_{KLD}^k = \sum_{(x,y) \in \mathcal{I}_k} p[y|x; (\theta_{k-1}, \phi_{k-1})] \log(p[y|x; (\theta_k, \phi_k)]). \tag{2.2}$$

In the equation above, $\mathcal{I}_k$ represents the training set for task k, consisting of only the rehearsal data ($\mathcal{I}_k = \mathcal{R}_0^{k-1}$), or the union of the rehearsal and current data of task k ($\mathcal{I}_k = \mathcal{D}_k \cup \mathcal{R}_0^{k-1}$).

The second regularization term is given by the mean squared error (MSE) loss between the output of the model encoder at tasks $k-1$ and k, so we are operating on the feature space:

$$\mathcal{L}_{MSE}^k = \sum_{x \in \mathcal{I}_k} \|\mathrm{ENC}_{\theta_{k-1}}(x) - \mathrm{ENC}_{\theta_k}(x)\|^2. \tag{2.3}$$

Also in this case, we experiment with both $\mathcal{I}_k = \mathcal{R}_0^{k-1}$ and $\mathcal{I}_k = \mathcal{D}_k \cup \mathcal{R}_0^{k-1}$.

The total loss to optimize in each task k is therefore a linear combination of the CE loss in eq. 2.1 and the regularization losses in eqs. 2.2 and 2.3:

$$\mathcal{L}^k = \lambda_{CE}\mathcal{L}_{CE}^k + \lambda_{KD}\mathcal{L}_{MSE}^k + \lambda_{KD}\mathcal{L}_{KLD}^k. \tag{2.4}$$

Figure 2.2 shows a schematic illustration of the proposed CL approach.

2.4.3 Experiments and Discussion

CIL Definition and Rehearsal Memory. We evaluate our proposed approach on the Fluent Speech Commands (FSC) dataset [147]. FSC includes 30,043 English utterances, recorded at 16 kHz. The dataset provides 248 different utterances mapped in 31 different intents. There is only one intent per utterance. To give an example, the intent *increase_heat_kitchen* is associated with the utterance "*turn up the temperature in the kitchen*". We split the dataset into train, validation, and test sets with a ratio of 80:10:10 as proposed in [147].

To define the CIL scenario, we partition the FSC dataset into 10 disjoint tasks, where the first task comprises 4 unique intents and the subsequent 9 tasks contain 3 intents. The order of the intents is random, and we have not observed significant variations by considering different random orders.

Concerning the rehearsal memory, its entire capacity is not exploited from the very beginning, but each class has a pre-allocated space that is used when

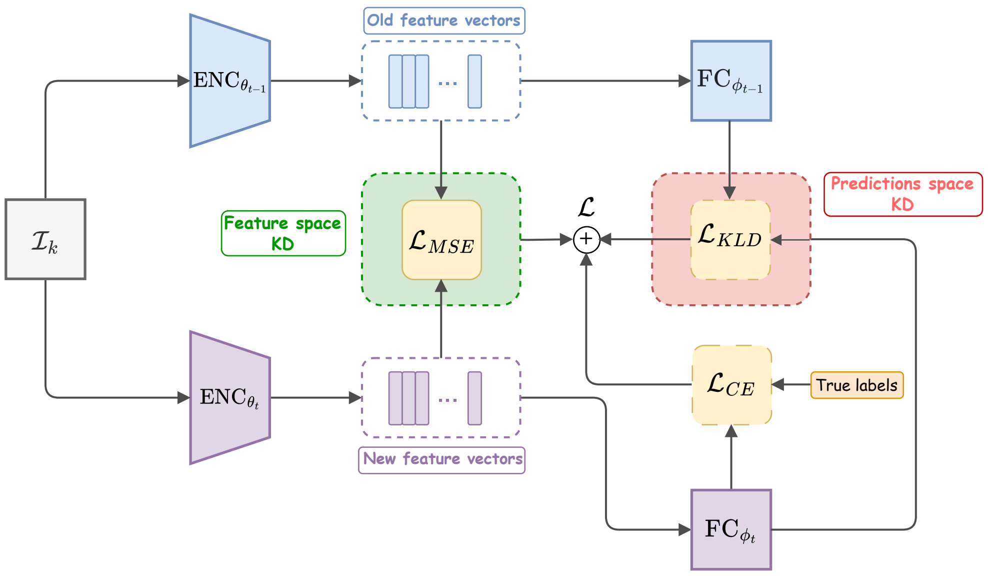


Figure 2.2: Overview of our proposed approach. Rehearsal samples are sampled by the rehearsal buffer $\mathcal{I}_k$ and interleaved with the data from the current task to compute the cross-entropy loss λ_{CE}. In addition to this, we add two KD losses, λ_{MSE} and λ_{KLD}, that foster the knowledge transfer from the teacher to the student model on the feature space and prediction space, respectively. For both losses, we experiment by computing them over only the rehearsal data or over rehearsal + new data.

that class is seen for the first time. In this way, we avoid a possible imbalance among the classes between the first and the last tasks.

Model Architecture. For our experiments, we use the temporal convolutional network (TCN) [148]. We observe that the KD strategies we propose are architecture-agnostic, so they do not rely on the underlying architecture. It would be possible to substitute the TCN network with any other deep architecture (e.g., transformer-based encoder). The network takes as input 40 Mel-spaced log filter-banks, computed using a sliding window of length 25 ms, with 10 ms stride. Then, it applies a global layer normalization (gLN) and a bottleneck layer (1x1 conv block) that maps the input features into 64 channels. The input layer is followed by 2 repetitions of 5 consecutive 1-D dilated convolutional residual blocks (Figure 2.3). Each residual block is formed by two symmetrical pipelines surrounding a depth-wise separable convolutional layer that maps the 64 bottleneck features into 128 channels.

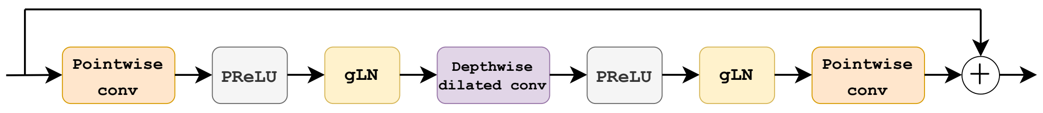


Figure 2.3: Structure of a 1-D convolutional residual block.

A residual branch connects the original input to the output. Mean pooling is applied to the output of the last block, followed by gLN and a linear layer. A softmax activation layer gives the final class scores.

Training Details. We train the TCN model for 50 epochs per task with Adam optimizer [115], with a learning rate equal to $5e^{-4}$. The CIL scenario is implemented with the Continuum library [62]. The code to reproduce all our experiments is available here.

On the Choice of the Distillation Weights. The selection of the KD weights deserves special attention. A common choice for the λ_{KD} weight is $\frac{n}{n+m}$, where n is the number of old (seen) classes and m is the number of new classes [232, 257]. This choice was originally proposed for works that used only KD as CL strategy, and it gives more and more importance to λ_{KD} over time because the past model retains the knowledge from more and more past classes. When we use both KD and rehearsal approaches applied to rehearsal and current data ($\mathcal{I}_k = \mathcal{D}_k \cup \mathcal{R}_0^{k-1}$), the importance of the past model is damped by the fact that the current model sees the rehearsal data, so we still would like λ_{KD} to increase, but at a slower pace, and this can be accomplished by using a log function. When we use the KD applied only to the rehearsal data ($\mathcal{I}_k = \mathcal{R}_0^{k-1}$), we give it a weight proportional to the fraction of rehearsal data in the mini-batch. Since this number is too small during the first tasks, we apply the square root operation to enlarge it. Ultimately, we set λ_{KD} as follows:

$$\lambda_{KD} = \begin{cases} \log(1 + \frac{n}{n+m}) & \text{if } \mathcal{I}_k = \mathcal{D}_k \cup \mathcal{R}_0^{k-1} \\ \sqrt{\frac{b_{rehe}}{b_{all}}} & \text{if } \mathcal{I}_k = \mathcal{R}_0^{k-1} \end{cases} \tag{2.5}$$

where b_{rehe} counts the number of rehearsal data in the current mini-batch, and b_{all} is the current mini-batch size. We found empirically that using λ_{KD}

Table 2.1: Intent classification accuracy with 930 samples in the rehearsal memory, using different distillation strategies (Feature and Prediction Space KDs, and their combination) and rehearsal-based strategies. The **best** accuracies overall are reported in bold.

Baselines		**last acc**	**avg acc**						
Offline		0.98	-						
Finetuning		0.07	0.27						
Pred. KD (no rehe)		0.08	0.27						
Feat-KD	**Pred-KD**	Random		Mean		iCaRL [182]		GEM [145]	
data	**data**	**Last Acc**	**Avg Acc**	**Last Acc**	**Avg Acc**	**Last Acc**	**Avg Acc**	**Last Acc**	**Avg Acc**
-	-	0.66	0.72	0.65	0.69	0.68	0.74	0.57	0.71
Feature Space KD									
$\mathcal{R}$	-	0.74	0.78	0.73	0.74	0.79	0.80	0.77	0.79
$\mathcal{D} \cup \mathcal{R}$	-	0.59	0.64	0.56	0.61	0.60	0.64	0.71	0.71
Prediction Space KD									
-	$\mathcal{R}$	0.68	0.74	0.63	0.69	0.66	0.73	0.62	0.73
-	$\mathcal{D} \cup \mathcal{R}$	0.76	0.76	0.69	0.72	0.78	0.79	0.60	0.71
Double KDs									
$\mathcal{R}$	$\mathcal{R}$	0.75	0.77	0.73	0.74	0.79	0.79	0.76	0.80
$\mathcal{R}$	$\mathcal{D} \cup \mathcal{R}$	0.77	0.80	0.73	0.74	**0.81**	**0.81**	0.75	0.80

as defined in Eq. 2.5 brings a 1% to 2% improvement in accuracy. This study suggests that a careful choice of the KD weights is essential.

Eq. 2.4 changes depending on the considered experiment. When we do not apply any KD loss, the weights boil down to $\lambda_{KD} = 0$, $\lambda_{CE} = 1$ (in practice, only the CE loss is used). When we use the KD in the feature space only, the KLD loss is not present, λ_{KD} follows eq. 2.5, and $\lambda_{CE} = 1 - \lambda_{KD}$. If we use the KD in the predictions space, the same as before applies with the KLD loss and the MSE loss inverted. Lastly, when both the KLD loss and the MSE loss are employed, their coefficient λ_{KD} follows eq. 2.5, and λ_{CE} is set to 1.

Main Results. Table 2.1 reports the intent classification accuracy for different KD strategies in combination with 4 CL approaches, i.e. a rehearsal approach with 3 different sample selection strategies (random, iCaRL [182], and "closest_to_mean", where the samples which are closest to their class mean in the feature space are chosen), and GEM [145]. The rehearsal memory size is 930 (around 4% of the dataset size). We consider 2 random class orders, and for each, we run 4 experiments and take the average. We use 2 metrics to test the efficacy of each strategy: the average accuracy (**avg acc**), which is defined as the average of the accuracies after each task, and the accuracy after the last task (**last acc**).

In the upper part of Table 2.1 we include the results for: *i)* the offline upper bound (i.e., no incremental learning), which is in line with the current state-of-the-art methods on the FSC dataset; *ii)* the results obtained with the naive fine-tuning method, and *iii)* the results we achieve applying solely the KD in the predictions space without rehearsal. We note that the latter two methods incur severe catastrophic forgetting, and this confirms previous works stating that the KD alone is not sufficient to achieve good performance results in a CIL scenario [125].

The lower part of the table shows the accuracy results when rehearsal data are employed. We report results when the distillation is performed at the feature level, predictions level, and both levels, respectively. For each configuration, the table also reports the performance when distillation is applied to either rehearsal data alone (denoted with $\mathcal{R}$ in the table) or to the union of rehearsal and actual task data (denoted with $\mathcal{D} \cup \mathcal{R}$).

When we endow the model with the KD in the feature space, the reliance on only the rehearsal data improves both the average accuracy and the last accuracy. On the contrary, the joint use of $\mathcal{D} \cup \mathcal{R}$ deteriorates the performance. This can be explained by observing that if we use $\mathcal{D} \cup \mathcal{R}$, we are forcing the current model, θ_k, to produce feature representations that are similar to the ones obtained with the previous model, θ_{k-1}. Whereas this is desirable for the rehearsal data (the previous model has been trained on them), this is not the case for the new data, since we want our model to learn in the actual task new clusters which should be far apart from the past clusters.

Considering the KD in the predictions space, instead, we witness a trend inversion. The use of $\mathcal{D} \cup \mathcal{R}$ achieves better results than just using the

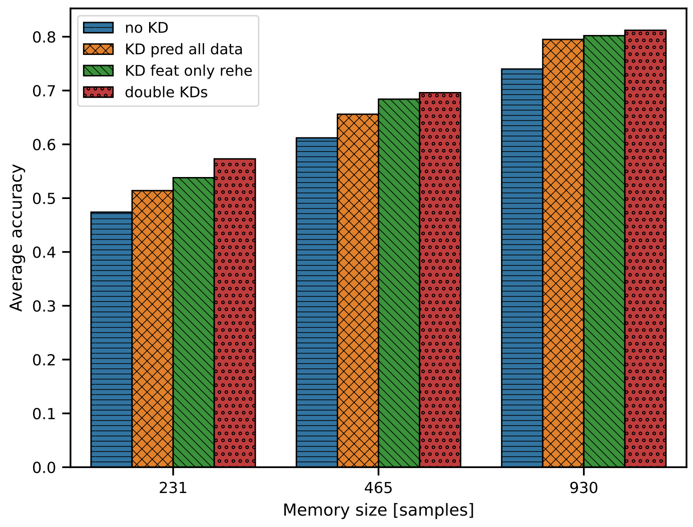


Figure 2.4: Average accuracies for different values of the memory size for the iCaRL strategy.

data in the memory, albeit the difference is not as pronounced as for the feature-space KD. We speculate that since in the predictions space we deal with probability distributions, the new and teacher models produce values for both new and past classes, thus it is more convenient to apply the KD to $\mathcal{D} \cup \mathcal{R}$ (e.g., $\mathcal{D}$ and $\mathcal{R}$ act as negative samples for the past and new classifiers, respectively). It is also worth noting that in almost all cases the feature-level KD attains slightly better results than its predictions counterpart. We point out that GEM achieves slightly better results when only rehearsal data are considered, and this may be because it already employs a regularization on the gradients using only the rehearsal data.

The last two rows of Table 2.1 consider the combination of feature-level and predictions-level KDs (the configurations with $\mathcal{D} \cup \mathcal{R}$ in the feature space are not considered since we have shown they highly deteriorate the model performance). The use of the features-space KD applied to $\mathcal{R}$ in conjunction with the predictions-space KD applied to $\mathcal{D} \cup \mathcal{R}$ gives the best results (0.811 and 0.812 for the last acc and avg acc by iCaRL, respectively), proving the effectiveness of integrating both KDs.

Ablation on the Rehearsal Memory Size. Finally, in Figure 2.4 we show the average accuracies achieved by different KDs approaches when using smaller rehearsal memory sizes (we use iCaRL sampling strategy given its superiority). We note the consistency of the performance trend as the memory size changes. In particular, the relative gain in accuracy provided by the KDs is larger when a rehearsal memory with a smaller capacity is used. For instance, the relative gain is around 7.2 points when the memory size is 930, whereas when the memory size is reduced to 231 samples the gain increases up to 9.9. This demonstrates the effectiveness of our approach also for limited-budget SLU systems.

2.4.4 Final Remarks and Future Work

In this work, we described an approach for class-incremental continual learning in a SLU domain. We show that the KD on the rehearsal data is effective if applied to the encoded features. Furthermore, the feature-level MSE loss, when added to the usual predictions-level KD loss, brings additional performance improvements. The efficacy of the approach is particularly evident when the rehearsal memory size is small, making it suitable for low-resource devices. One limitation is the dataset that, although large in terms of size, lacks lexical richness and variety. Thus, in the next section we extend the proposed approach to a more recent and complex end-to-end SLU dataset, the Spoken Language Understanding Resource Package (SLURP) [19], which also features the prediction of multiple entities inside a spoken sentence (e.g., slot filling).

2.5 Sequence-Level Knowledge Distillation for Class-Incremental End-to-End Spoken Language Understanding [32]

2.5.1 Overview

In the previous section, we defined a CIL setting for the problem of intent classification over the FSC dataset and tested multiple CL approaches combining rehearsal and knowledge distillation principles. However, SLU usually involves a more challenging problem than intent classification, which associates an intent label with the entire input sequence: entity classification or slot-filling. In entity classification, some words of the input are to be mapped to pre-defined slot labels. Sometimes, multiple entities are present in a single input, making the problem even harder.

The FSC dataset only entails the problem of intent classification, so we need to find another SLU benchmark to study entity classification under a CIL setting. Consequently, we shift our attention to the SLURP dataset [19], which is a multi-domain benchmark providing different levels of semantic annotations. For this reason, SLURP is considered the largest and most diverse dataset in terms of lexical complexity for SLU. Due to its lexical richness, the problems of intent and entity classification for SLURP are usually treated as a *sequence-to-sequence* (seq2seq) problem [11, 173], where the intents and entities are generated along with the transcriptions in an auto-regressive way. So, unlike the mainstream CL architecture composed of a feature extractor and a classifier, we exploit a transformer-based seq2seq architecture, whose decoder is also affected by forgetting as the encoder. We therefore propose to combine rehearsal with regularization via knowledge distillation (KD) to combat forgetting at both the encoder and decoder levels. We investigate three KD approaches: one is applied to the encoder's output (**audio-KD**), whereas the other two distill the knowledge at the decoder side, either at a local (**token-KD**) or global (**seq-KD**) level. Our experiments show that the seq-KD stands out as the best approach and that integrating multiple KDs at once leads to additional improvements.

In summary, our contributions are three-fold:

- We define a CIL scenario for the SLURP dataset;
- We study how to mitigate forgetting in a seq2seq model, thus moving away from the classical CL pipeline comprising a feature encoder followed by a classifier;
- We propose three KDs losses that effectively lower forgetting and discuss their individual and mutual contributions.

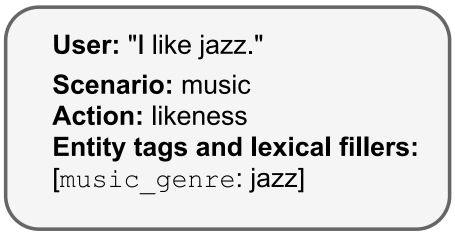

User: "I like jazz."
Scenario: music
Action: likeness
Entity tags and lexical fillers:
[`music_genre`: jazz]

Figure 2.5: Example of annotated utterance from the SLURP dataset. The intent in this case is the pair (music,likeness).

2.5.2 Class-Incremental Learning for SLURP

In this section, we describe how we define the CIL setting for the SLURP dataset [19]. SLURP is a multi-domain dataset for SLU comprising around 56 hours of audio of people interacting with a home assistant (*slurp_ real*), with the addition of 43.5 hours of synthetic recordings (*slurp_ synth*). At present this makes SLURP the biggest and the most diverse dataset in terms of lexical complexity for SLU. Each utterance is annotated with three semantics: *Scenario*, *Action*, and *Entities*. The pair (scenario, action) is defined as *Intent*. Overall, there are 18 unique scenarios, 46 actions (56 if we consider both *slurp_ real* and *slurp_ synth*), 55 entity types, and 69 intents. Figure 2.5 provides an example of an annotated utterance.

We select the scenarios as a splitting criterion to define the tasks of the CIL setting. The full list of scenarios is: [“**alarm**”, “**audio**”, “**calendar**”, “**cooking**”, “**datetime**”, “**email**”, “**general**”, “**iot**”, “**lists**”, “**music**”, “**news**”, “**play**”, “**qa**”, “**recommendation**”, “**social**”, “**takeaway**”, “**transport**”, “**weather**”]. Since the number of scenarios is limited and each scenario provides a high-level concept associated with each utterance, we think that they can closely resemble a practical application that must adapt to new general domains. Additionally, since the intent classification is the chief metric to assess our model against, the use of scenarios as splitting criterion abides by the rule of having only intents related to scenarios available in the current task. Finally, although some actions and entities can be included in multiple scenarios, the overlap is very limited because the majority of the entities and actions are specific to a single scenario. For example, the

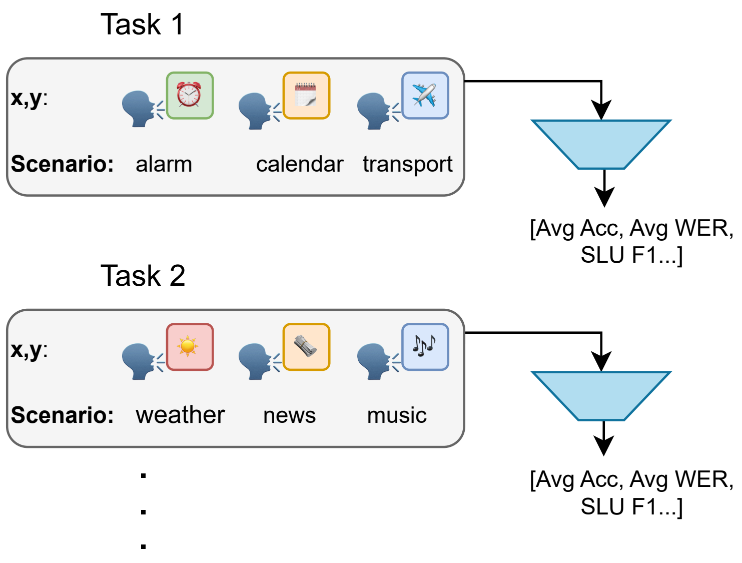

Task 1
x,y:
Scenario: alarm calendar transport
[Avg Acc, Avg WER, SLU F1...]
Task 2
x,y:
Scenario: weather news music
[Avg Acc, Avg WER, SLU F1...]

Figure 2.6: The class-incremental learning setting for the SLURP dataset, where 3 new scenarios are introduced in each task.

action "**taxi**" is only associated with the scenario "**transport**", and the entity "**weather_descriptor**" with the scenario "**weather**". Figure 2.6 shows two consecutive tasks, each introducing 3 new scenarios.

Another critical aspect is the order in which the scenarios are available to the model. In our implementation, the order depends on the cardinality so that the scenarios with the highest cardinality appear first. In this way, we simulate a practical situation in which we endow the model with the sufficient general knowledge, learning the largest scenarios first, that will be useful for learning more specific scenarios.

2.5.3 Method

Similar to the FSC dataset we have discussed in section 2.4.2, we divided the SLURP dataset into K distinct tasks, $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$, based on the scenario labels, so that a scenario is included in one and only one task. The dataset $\mathcal{D}_k$ of the k^{th} task comprises audio signals $\mathcal{X}_k$ with associated transcriptions $\mathcal{Y}_k$, i.e. $\mathcal{D}_k = (\mathcal{X}_k, \mathcal{Y}_k)$. We remind that the CIL setting is challenging because the model must be able to distinguish all classes till task

k, thus at test time the task labels are not accessible (unlike task-incremental learning) [100].

We employ a transformer-based seq2seq ASR architecture, constituted by a Wav2vec 2.0 encoder (WavEnc) [14] followed by a transformer decoder. Let $\mathbf{x} = [x_1, \ldots, x_I]$ be an audio input sequence of length I, and $\mathbf{y} = [y_1, \ldots, y_J]$ be the corresponding output sequence of length J, with $y_j \in \mathcal{V}$, where $\mathcal{V}$ is the set of all possible output subword tokens. The goal of the ASR model is to find the most probable output sequence $\hat{\mathbf{y}}$ given the input sequence $\mathbf{x}$:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x}; \theta), \tag{2.6}$$

where $\mathcal{Y}^*$ is the set of all possible token sequences and θ represents the parameters of the seq2seq model.

Suppose that $p(\mathbf{y}|\mathbf{x}; \theta_k)$ and $p(\mathbf{y}|\mathbf{x}; \theta_{k-1})$ are the output probability distributions of the transformer decoder at task k and $k-1$ parameterized by θ_k and θ_{k-1}, respectively. The model at task $k-1$ can be seen as the teacher model. Let also $\mathcal{R}_k$ be the set of rehearsal data at the beginning of task k. In the following equations, we use $\mathbf{x} \in \mathcal{D}_k$ in place of $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_k$ for brevity. The standard training criterion of rehearsal-based CL methods consists of minimizing the cross-entropy loss over $\mathcal{D}_k \cup \mathcal{R}_k$:

$$\mathcal{L}_{\mathrm{CE}}^k = -\sum_{\mathbf{x} \in \mathcal{D}_k \cup \mathcal{R}_k} \log(p(\mathbf{y}|\mathbf{x}; \theta_k)). \tag{2.7}$$

The main idea of KD is to transfer knowledge from the teacher network $p(\mathbf{y}|\mathbf{x}; \theta_{k-1})$ to a student model, such that the latter mimics the former's behavior. Basically, the KD is used to force the current model to not deviate too much from the teacher, which retains the knowledge of the previous tasks. We point out that the KD, unless otherwise stated, is applied to the sole rehearsal data since the teacher can effectively predict only the data seen in the previous tasks. We propose three different types of KDs: *audio-KD*, *token-KD*, and *seq-KD*. The audio-KD works on the encoder's output level, whereas the other two KDs are applied to the output of the decoder. In this way, we contrast forgetting either on the encoder or on the decoder side (or both, if we combine multiple KDs).

The **audio-KD** forces the encoder's audio embeddings of the current task k to resemble those from the previous task $k-1$. Let $\mathrm{WavEnc}(\mathbf{x}) \in \mathbb{R}^h$ be the Wav2vec 2.0 encoder output followed by a mean operation to squeeze the temporal dimension, where h is the hidden size. We define the audio-KD loss as:

$$\mathcal{L}^k_{\text{audio-KD}} = \sum_{\mathbf{x} \in \mathcal{R}_k} \|\mathrm{WavEnc}_{\theta_{k-1}}(\mathbf{x}) - \mathrm{WavEnc}_{\theta_k}(\mathbf{x})\|^2, \tag{2.8}$$

where $\|\cdot\|$ is the Euclidean distance operator. Eq. 2.8 acts as a regularization term for the encoder.

We can apply a similar reasoning to the decoder, which predicts each word of the transcription in an autoregressive way (in our case we use Byte-Pair Encoding [191], so we will use the term token rather than word to refer to the output units). The **token-KD** forces the current decoder to match the token-level distribution of the teacher. This is a kind of "local" distillation in that the student mimics the teacher for each token of the transcription. The corresponding CE criterion is defined as:

$$\mathcal{L}^k_{\text{tok-KD}} = -\sum_{\mathbf{x} \in \mathcal{R}_k} \sum_{j=1}^{J} p(y_j|\mathbf{x}, \mathbf{y}_{<j}; \theta_{k-1}) \log(p(y_j|\mathbf{x}, \mathbf{y}_{<j}; \theta_k)), \tag{2.9}$$

where $\mathbf{y}_{<j}$ is the output sequence up to token $j-1$.

A potential flaw of this method is that if some initial token distributions are poorly estimated, their bias will be propagated until the end of the sequence. Indeed, a predicted token might be optimal at the current position in the sequence, but as we proceed through the rest of the sentence, it might turn out not to be the optimal one, given that later predicted positions are not already available.

Seq-KD is an alternative approach that trains the student to generate the same output sequence as the teacher, thus working on the sequence level. This can be treated as a "global" distillation since we work on the whole transcription. In practice, we generate a new set of automatic transcriptions with the teacher model using beam search at the end of each task ("soft

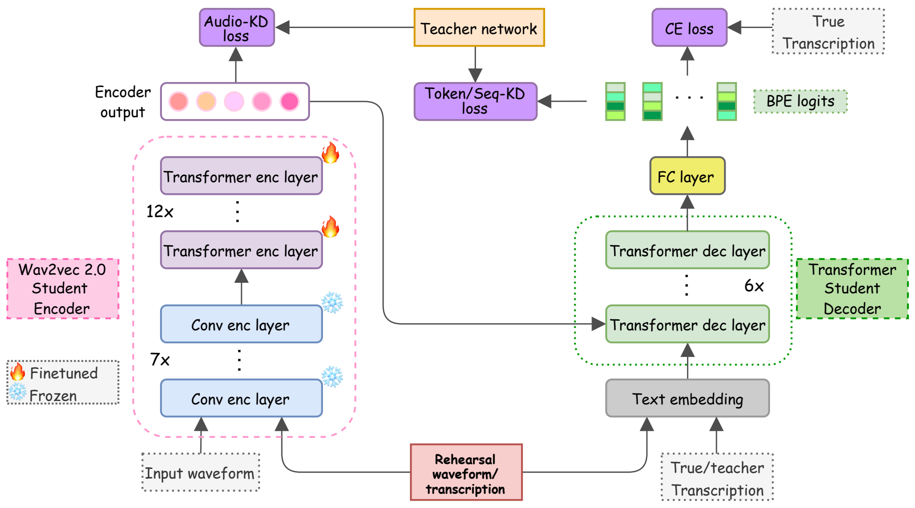


Figure 2.7: Illustration of the learning process in the proposed CIL setting. The model from the current task (student) mimics the behavior of the teacher model through **audio**, **token**, and **sequence KD** losses to counter forgetting.

transcriptions"), and then we use them to train the student network with the CE criterion in the next task. Formally, we add the following CE loss:

$$\mathcal{L}^k_{\text{seq-KD}} = - \sum_{\mathbf{x} \in \mathcal{R}_k} \log(p(\tilde{\mathbf{y}}|\mathbf{x};\theta_k)), \tag{2.10}$$

where $\tilde{\mathbf{y}}$ is the output sequence generated with beam search using the teacher model.

Overall, the total loss to be optimized for task t is:

$$\mathcal{L}^t_{\text{TOT}} = (1 - \lambda_{\text{KD}})\mathcal{L}^t_{\text{CE}} + \sum_{k \in \mathcal{K}} \lambda_{\text{KD}} \mathcal{L}^t_k, \tag{2.11}$$

where $\mathcal{K} = \{\text{audio-KD}, \text{tok-KD}, \text{seq-KD}\}$ and λ_{KD} is a weighting parameter. Depending on whether we employ a single KD or multiple ones, Eq. 2.11 changes accordingly. Figure 2.7 shows the learning process with the three KD losses applied to the transformer architecture.

2.5.4 Experiments and Discussion

Dataset and CIL setting. We conduct experiments on the SLURP dataset [19] using the official train, validation, and test splits, with a ratio of 70:10:20. In all experiments we also use *slurp_synth* only for training. Since very long audio data are harmful for efficient training, we remove the training samples longer than 7 seconds (around 0.004% of the total training dataset).

Concerning the definition of the CIL setting, we experiment on two configurations: 1) the dataset is partitioned into 3 tasks, each comprising 6 scenarios (denoted as SLURP-3); 2) a more challenging configuration wherein the 18 scenarios are distributed across 6 tasks (denoted as SLURP-6).

Pre-processing. As proposed in [11], the intent and entity classification problems are treated as a sequence-to-sequence ASR task, where both intent and entities associated with an utterance are predicted alongside its transcription. In a sense, we build an "augmented" transcription that will be fed to the transformer decoder, prepending the intent to the original transcription, followed by the entities and the corresponding lexical values. The special token *_SEP* is used to separate the intent from the entities and the entities from the original transcription, whereas the token *_FILL* is used to separate each entity from its value. If the original transcription is the one in Fig. 2.5, then the augmented transcription becomes: *music_likeness _SEP music_genre _FILL jazz _SEP I like jazz.*

Model. The encoder is the base Wav2vec 2.0 model [14] pretrained and fine-tuned on 960 hours of Librispeech (a CNN-based feature extractor followed by 12 transformer blocks with hidden size = 768, 8 attention heads, 2048 FFN hidden states). The feature extractor is kept frozen during the training, whereas the transformer blocks are fine-tuned. Then, the transformer decoder includes 6 layers with the same parameters as the encoder. We apply layer normalization to the input raw waveforms. The total number of parameters of the model is around 148M.

Training. We tokenize the transcriptions using Byte-Pair Encoding (BPE) [191], with a vocabulary size of 1k and BPE dropout = 0.1. Both at inference time and for computing the soft labels for the KD-seq, we run beam search with beam width = 20. The number of epochs for each task is {40,25,15} for SLURP-3, whereas {40,25,15,15,15,15} for SLURP-6. The batch size is 32. We use AdamW optimizer [146] with learning rate = $5e^{-5}$ and weight decay = 0.1. We use the validation set for hyperparameters tuning, and for selecting the best model for each task that is used for testing.

CL Baselines and Strategies. Our upper bound is the *offline* method consisting of a single macro-task with all the scenarios (i.e. no incremental learning), while the naive *fine-tuning* approach, which retrains the same model task by task, is our lower bound. We consider two different sampling strategies for the rehearsal approach: 1) a random selection of the samples to retain, and 2) iCaRL [182], which selects the samples closest to their moving barycenter. We provide an example with a memory buffer of size equal to around 5% of the training dataset, and the rest of the experiments use 1%. Finally, we show the result for each KD strategy, as well as their various combinations. The KD weight in Eq. 2.11 is proportional to the fraction of rehearsal data in the mini-batch and is defined as:

$$\lambda_{KD} = \sqrt{\frac{b_{rehe}}{b_{all}}}, \tag{2.12}$$

where b_{rehe} is the number of rehearsal data in the current mini-batch, and b_{all} is the current mini-batch size. This choice has been already explained in section 2.4.3.

Metrics. We evaluate the proposed methods using 4 metrics: the average intent accuracy, **Avg Acc**, defined as the average of the intent accuracies after each task; the intent accuracy after the last task (**Last Acc**); the average **SLU F1** metric for entity classification [19]; the average word error rate, **Avg WER**, after each task.

Main Results.

The performance for both CIL settings, SLURP-3 and SLURP-6, are reported in Table 2.2. First and foremost, we note that, as expected, the fine-tuning approach struggles in both settings, thus incurring catastrophic forgetting. The use of a rehearsal memory (rows Rehe-5% and Rehe-1%) proves to be very effective, even with only 1% of retained data. Therefore, in

Table 2.2: Results in terms of Average Accuracy (↑), Last Accuracy (↑), Average WER (↓), and SLU F1 (↑) for different strategies.

Method	**SLURP-3**				**SLURP-6**			
	Avg Acc	**Last Acc**	**Avg WER**	**SLU F1**	**Avg Acc**	**Last Acc**	**Avg WER**	**SLU F1**
Offline	85.84	-	20.46	70.59	85.84	-	20.46	70.59
Fine-tuning	46.27	18.36	35.82	49.25	33.56	12.42	46.26	37.88
Rehe-5% rand	79.79	74.82	25.79	65.85	77.12	73.11	28.87	63.22
Rehe-1% rand	71.30	61.47	29.13	60.05	66.11	59.37	34.77	55.33
Rehe-1% iCaRL	71.49	61.66	28.62	60.23	67.55	62.55	33.82	56.09
+ **audio-KD**	72.14	63.03	28.68	61.08	68.40	62.83	**32.04**	58.15
+ **token-KD**	71.79	61.54	28.82	**61.88**	68.36	62.53	32.47	58.20
+ **seq-KD**	**76.12**	**68.94**	**28.56**	61.50	**71.56**	**64.82**	32.50	**58.29**

the following experiments we consider 1% of data in the rehearsal memory. We also experiment with a more sophisticated sampling strategy, iCaRL [182], which achieves noteworthy improvements, in particular for SLURP-6 (+1.44% of Avg Acc).

When we focus on the proposed KDs, we can observe that the seq-KD leads to the most substantial improvement for both Avg and Last Acc metrics (+4.63% and +7.28% on SLURP-3). Instead, for WER and SLU F1, all three KDs behave similarly. Note that in our setting, previous intents are not seen anymore, and indeed the KDs help the model remember past scenarios. Conversely, though we expect the utterances to have some scenario-specific words, general speech tokens are spread among the tasks, making forgetting less critical for WER. Nevertheless, for the more challenging SLURP-6, KDs bring a notable enhancement also in terms of WER and SLU F1.

Combining Multiple KDs. In this final section, we investigate whether combining multiple KD approaches results in additional improvement. We try to combine two KDs at a time, and all three KDs together. The results for SLURP-6 are reported in Table 2.3. As expected, the best combinations involve the use of seq-KD. Indeed, when the seq-KD is not included (audio + token), the results are even worse than using the KDs individually. Instead,

Table 2.3: Analysis of different KD combinations on SLURP-6.

Combination	**Avg Acc**	**Last Acc**	**Avg WER**	**SLU F1**
audio + token	68.13	61.50	32.46	57.30
audio + seq	**72.48**	65.25	**31.37**	**60.00**
seq + token	72.07	63.46	33.08	58.25
audio + seq + token	71.83	**65.45**	32.55	58.48

the best combination is given by audio and seq KDs, the two approaches that yield the best improvement if taken singularly. We guess that forcing the encoder output of the current task to be similar to that of the previous task (audio-KD) favors the cross-attention layer of the decoder to attend to the most relevant part of the audio signals. We also mention that using all three KDs leads to satisfactory results, yet slightly worse than seq + audio. We think that, for this last case, since four KDs are involved, the design of the KD weights is more cumbersome, and more experiments are necessary to find the optimal weights.

Finally, in Figure 2.8 we show the trend of the intent accuracy task by task for SLURP-6. We observe that the seq-KD outperforms both audio and token KDs by a large margin on all steps, and its combination with the audio KD leads to further accuracy enhancement.

As a last analysis, we point out that the additional computational burden brought by the proposed KDs is limited for two reasons: 1) the KD losses take into account only the rehearsal samples, which are just a fraction of the entire dataset (i.e., 1%); 2) they just involve an additional forward pass through the teacher model, which is kept frozen during the current task.

2.5.5 Final Remarks and Future Work

In this work, we defined a CIL setting for a challenging SLU dataset, SLURP. To mitigate forgetting, we propose three different KD-based strategies working at different levels in the seq2seq model. Our extensive experiments reveal the superior performance of the seq-KD, and that combining multiple KDs

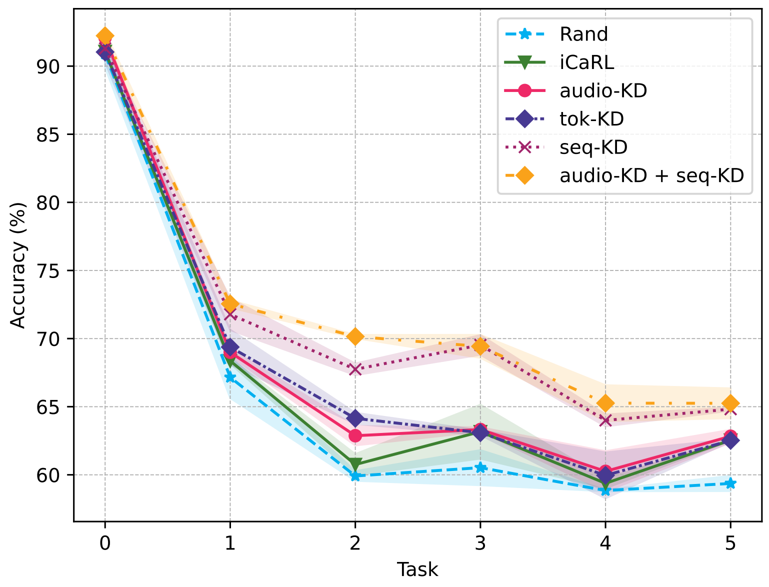


Figure 2.8: The trend of the intent accuracy on the observed tasks for the SLURP-6 setting.

results in additional improvements. An interesting future direction is refining the seq-KD by exploiting multiple beam search hypotheses with their corresponding scores.

2.6 Continual Contrastive Spoken Language Understanding [30]

2.6.1 Overview

In the previous two sections, we have studied the problem of CIL applied to SLU. To mitigate the issue of catastrophic forgetting, we have relied on the combination of *experience replay* and *knowledge distillation* principles. In particular, in [32] we have formulated SLU as a seq-to-seq problem, where the intent labels are generated along with the ASR transcriptions in an autoreregressive manner, and proposed two effective distillation strategies applied to the ASR decoder. However, we speculate that we can exploit the multimodal (i.e., audio-text) nature of our setting to contrast forgetting at the audio and text encoder outputs through the combination of *experience replay* and *contrastive learning*.

Contrastive learning [45, 166] is a popular approach in self-supervised learning, but it can also be used in supervised learning [79] and multimodal learning [180]. Its objective is to learn discriminative feature representations by pushing apart different samples (negatives) and bringing closer similar ones (positives). Whereas experience replay is a well-established approach in continual learning, whereby a bunch of old training samples is collected into a dedicated rehearsal memory buffer and interleaved with the data from the new task [17, 184], only recently has contrastive learning been harnessed to learn representations continually. Both supervised [33, 238] and self-supervised [68, 225] contrastive learning have proven useful in lessening the catastrophic forgetting issue. In the case of supervised CIL, it has been shown that endowing the model with contrastive learning objectives results in more robust representations against catastrophic forgetting. For incremental semantic segmentation, [238] and [258] propose to exploit contrastive learning in conjunction with knowledge distillation. For image classification, [223] advance a contrastive learning strategy based on the vision transformer architecture for online CIL. Motivated by the effectiveness of contrastive learning-based CIL in other domains, we propose **COCONUT** : **CO**ntinual **C**ontrastive sp**O**ken la**N**guage **U**nders**T**anding. COCONUT combines experience replay and contrastive learning principles. Specifically, COCONUT relies on two contrastive learning-based losses that operate on a shared embedding space where the audio and text features are projected.

The first loss, coined *Negative-Student Positive-Teacher* (NSPT), is a modified version of the supervised contrastive learning loss that aims to consolidate what the model has learned in the previous tasks. It also exploits KD [94, 132] to guide the current model (student) to produce representations that resemble the ones obtained with the model from the previous tasks (teacher). *For this reason, this loss is computed only on the rehearsal data (i.e., the anchors)*. A key difference between our loss and the standard contrastive one is that the positive samples are computed using the teacher (the positives only come from the rehearsal data), whereas the negatives

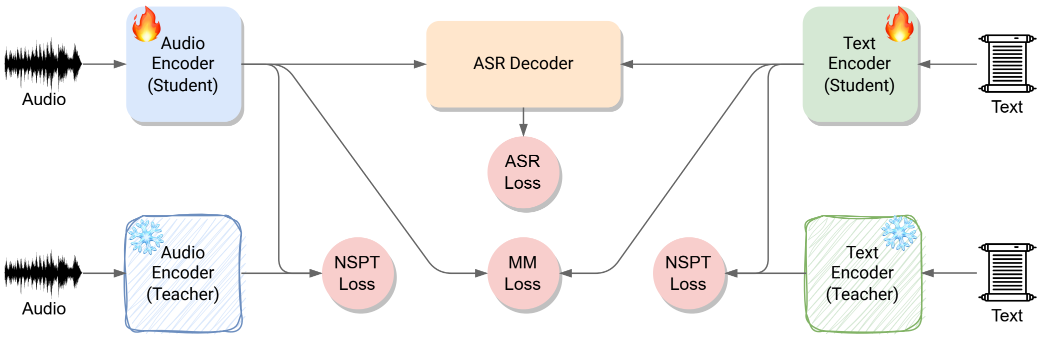


Figure 2.9: Overview of COCONUT 🥥 . It uses two contrastive learning-based losses. The NSPT (negative-student positive-teacher) loss is a supervised contrastive distillation loss that preserves the feature representations of the *past* classes for both audio and text samples. The positive and negative samples are computed with the teacher and student model, respectively. The MM (multimodal) loss aims to align audio and text representations belonging to the same *new* class. COCONUT produces features that are more transferable and resilient to catastrophic forgetting.

are computed with the student. In this way, we avoid stale and scattered representations for the new data.

The second loss is inspired by the recent progress in multimodal representation learning. Considering that for audio-text paired data, audio and text represent the same information but in different ways, it has been shown that aligning their representations results in better performance for various speech-related problems [156, 243, 266]. Therefore, we propose a multimodal (MM) supervised contrastive loss that, *exclusively applied to the current task's data*, brings audio and text representations belonging to the same class into closer proximity in the shared feature space, resulting in features that are more transferable and resilient to forgetting. An overview of COCONUT is illustrated in Figure 2.9.

In summary, our contributions are three-fold:

- We introduce COCONUT 🥥 , a continual learning method that makes use of two supervised contrastive learning objectives to mitigate forgetting for seq2seq SLU models. In particular, through our proposed NSPT loss we provide a detailed study of which models (student/teacher) should be used at the numerator/denominator (positives/negatives) of the contrastive loss tailored for class-incremental learning;

- We conduct extensive experiments on two popular SLU benchmarks demonstrating that COCONUT achieves consistent improvements over the baselines. We also show that it can be combined with KD applied to the ASR decoder, leading to further improvements;
- Finally, we ablate the contribution of each loss and its components, showcasing their pivotal role in COCONUT.

2.6.2 Problem Formulation

For our experiments, we define a CIL for the SLURP and FSC datasets setting following [32] (see section 2.5.2). The only difference is the letter we use for referring to the current task: here we use n rather than k for our notation. Following [32], we define SLU as a ASR-SLU multi-task learning problem. In fact, SLU is considered a more difficult task than ASR and NLU since it involves both acoustic and semantic interpretation [211]. For this reason, it is common practice to include an additional ASR objective such that the SLU labels (in our case the intent labels) and the transcript are generated in an auto-regressive fashion, resulting in a multi-task learning setting [11,173]. By doing this, the text transcript input to the model includes a class intent token that is specific to the actual task.

Let θ be the parameters of a seq2seq ASR model comprising an audio encoder, a text encoder (i.e., embedding layer), and an ASR decoder. Let $\mathbf{x} = [x_0, \ldots, x_{U-1}]$ be an audio input sequence of length U, and $\mathbf{y} = [y_{cls}, y_{sep}, y_0, \ldots, y_{J-3}]$ be the "extended" input transcript of length J, where with the term "extended" we refer to the original transcript $[y_0, \ldots, y_{J-3}]$ augmented with the intent class token y_{cls} and a special separation token y_{sep}. The goal of the ASR model is to find the most likely extended transcript given the input sequence $\mathbf{x}$:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x}; \theta), \tag{2.13}$$

where $\mathcal{Y}^*$ is the set of all token sequences. The predicted intent is obtained extracting y_{cls} from $\hat{\mathbf{y}}$.

2.6.3 Proposed Approach

Standard Rehearsal-based Approach.

We assume the availability of a rehearsal buffer, $\mathcal{M}$, in which we can store a few samples for each class encountered in the previous tasks. During the training phase of task n, $\mathcal{D}_n$, we refer to $\mathcal{B}$ as a mini-batch of samples $(\mathbf{x}, \mathbf{y})$, some of which come from the current task and others from the rehearsal memory. The rehearsal memory is gradually filled up until we reach the last task, so we do not need to employ any replacement strategy to make room for the classes of the new task. To increase the variance of the audio data, we apply SpecAug [170] to the audio waveform $\mathbf{x}$ as a data augmentation transformation. We do not implement any augmentation technique for the transcript $\mathbf{y}$. We encode each modality separately through a dedicated feature encoder. An audio encoder maps each audio input into a feature vector $\mathbf{h}_{\mathrm{A}} \in \mathbb{R}^{U \times d_{\mathrm{A}}}$, where d_{A} is the audio hidden size. Similarly, a text encoder converts each text input into a feature vector $\mathbf{h}_{\mathrm{T}} \in \mathbb{R}^{J \times d_{\mathrm{T}}}$, where d_{T} is the text hidden size. At this point, if no specific CL losses are used, the ASR decoder generates the output sequence in an auto-regressive fashion, cross-attending on the audio encoder's representations $\mathbf{h}_{\mathrm{A}}$. Thus, at task k, we minimize the conventional cross-entropy loss over the current mini-batch $\mathcal{B}$:

$$\mathcal{L}_{\mathrm{ASR}} = -\frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{B}} \log(p(\mathbf{y}|\mathbf{x};\theta)). \tag{2.14}$$

COCONUT 🥥

Preliminaries. We introduce here some notations for our proposed approach. Since we work with audio and text sequences, we need to aggregate the features we obtain with the encoders before computing the contrastive loss. For the audio component $\mathbf{h}_{\mathrm{A}}$ we apply a mean operation over its sequence length, whereas for text we only select the feature related to the intent token. Then, as is common practice in contrastive learning [45, 180], the resulting embeddings go through two separate linear projection layers that map them into a shared embedding space. At inference time, the projection layers are discarded. Therefore, we get the projected embeddings **a** and **t** in the following way:

$$\mathbf{a} = g_{\mathrm{A}}(avg(\mathbf{h}_{\mathrm{A}})), \quad \mathbf{t} = g_{\mathrm{T}}(cls(\mathbf{h}_{\mathrm{T}})), \tag{2.15}$$

where $cls(\cdot)$ is a function that extracts the feature associated with the class token, $g_{\mathrm{A}}(\cdot)$ and $g_{\mathrm{T}}(\cdot)$ are the projection layers, $\mathbf{a} \in \mathbb{R}^{d_{\mathrm{S}}}$ and $\mathbf{t} \in \mathbb{R}^{d_{\mathrm{S}}}$, where d_{S} is the dimension of the shared space.

Furthermore, we introduce some notations for the indices of samples coming from the current mini-batch $\mathcal{B}$. Let $\mathcal{I}_c$ and $\mathcal{I}_r$ represent the set of indices of the *new task* samples and the indices of the samples from the rehearsal memory (*old task* samples) in $\mathcal{B}$, respectively. Also, let $\mathcal{I} = \mathcal{I}_c \cup \mathcal{I}_r$, and we define $\mathcal{P}(k)$ as the set of indices of positive samples (i.e., samples with the same intent token).

The objective of a standard supervised contrastive loss (SCL) [113] is to push the representations of samples with different classes (negative pairs) farther apart while clustering representation of samples with the same class (positive pairs) closely together. Suppose that we get from the projection layers a generic representation $\mathbf{z}_i^D$ for the i-th element in the batch, where $\mathbf{z} = \{\mathbf{a}, \mathbf{t}\}$ and the superscript D denotes whether the representation is computed with the teacher or student model. A generic formulation of the SCL loss takes the following form:

$$\mathcal{L}_{\mathrm{SCL}} = \sum_{k \in \mathcal{I}} \frac{-1}{|\mathcal{P}(k)|} \sum_{p \in \mathcal{P}(k)} \log \frac{\exp(\mathbf{z}_k^D \cdot \mathbf{z}_p^D / \tau)}{\sum_{i \in \mathcal{I}} \exp(\mathbf{z}_k^D \cdot \mathbf{z}_i^D / \tau)}, \tag{2.16}$$

$\tau \in \mathbb{R}^+$ is a fixed temperature scaling parameter.

Supervised Contrastive Distillation Loss (NSPT). This loss combines the benefits of KD with those of contrastive learning [202, 206]. We recall that KD-based methods are very popular in CL and they exploit this paradigm to penalize changes to the model's intermediate or final outputs by fostering the pass of the knowledge accrued in the teacher model onto the student [27,61,182]. Commonly, we denote with *teacher* the model trained in the previous task, and with *student* that trained in the current task. Therefore, since the teacher conveys information about the previous classes, we would like to use it as a guide for the student through a KD objective. In this way, the loss encourages the student to produce audio and text embeddings consistent with those obtained by the teacher. For this reason, only the rehearsal samples are involved in this process as the teacher had no chance to see the current data. Additionally, we want to pull closer embeddings sharing the same intent class (i.e. the positives), while we push away the others (i.e. the negatives, whose class is different). This is obtained via a modified version of the standard supervised contrastive loss tailored for our setting. In fact, a standard one would use the teacher to compute both the positives and the negatives [113]. However, since the teacher is frozen and it is pointless to compute the representations of the samples from the current task using the teacher, we propose to use the student for computing the representations of the negatives. A small fraction of negatives come from the rehearsal buffer, and we also compute them using the student. We show in the ablation studies of section 2.6.6 that using the teacher deteriorates the performance. Therefore, our contrastive distillation loss computes the embeddings of the anchor and its corresponding negatives using the student, while the positives come from the teacher (we call this loss *Negative-Student Positive-Teacher*, NSPT). On the contrary, for the standard contrastive loss both the positives and negatives are computed with the teacher (we call it *Negative-Teacher Positive-Teacher*, NTPT). Figure 2.10 illustrates visually how the NTPT and NSPT work in the shared embedding space. The NSPT loss is computed for both audio and text embeddings, leading to two components, one for each modality, as follows:

$$\mathcal{L}_{\text{NSPT}} = \sum_{k\in\mathcal{I}_r} \frac{-1}{|\mathcal{P}(k)|} \sum_{p\in\mathcal{P}(k)} \Bigg[\underbrace{\log \frac{\exp(\mathbf{a}_k^n \cdot \mathbf{a}_p^{n-1}/\tau)}{\sum_{i\in\mathcal{I}} \exp(\mathbf{a}_k^n \cdot \mathbf{a}_i^n/\tau)}}_{\mathcal{L}_\text{A}} + \underbrace{\log \frac{\exp(\mathbf{t}_k^n \cdot \mathbf{t}_p^{n-1}/\tau)}{\sum_{i\in\mathcal{I}} \exp(\mathbf{t}_k^n \cdot \mathbf{t}_i^n/\tau)}}_{\mathcal{L}_\text{T}} \Bigg], \tag{2.17}$$

where n and $n-1$ denote whether the representation is obtained with the student or teacher, and $\mathcal{L}_\text{A}$ and $\mathcal{L}_\text{T}$ represent the audio and text contributions,

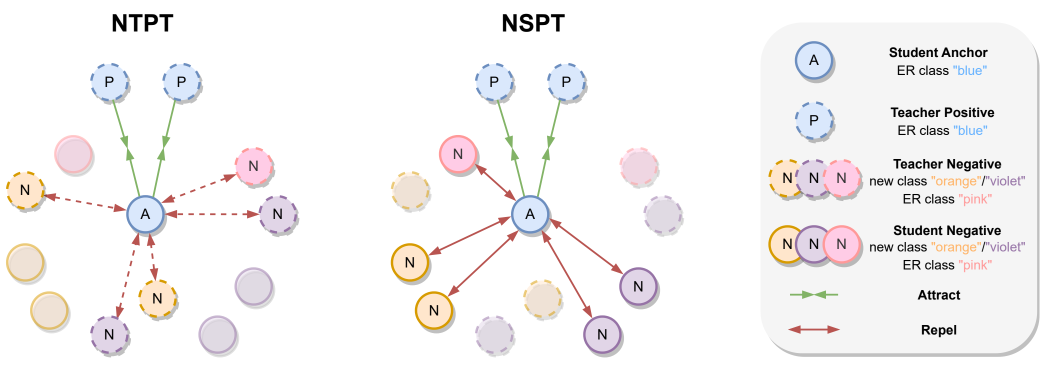


Figure 2.10: Illustration of the NTPT loss and our proposed NSPT loss. Given an anchor sample from the current mini-batch, the NTPT loss computes the negatives and positives using the teacher model (dashed circles). Instead, the NSPT loss computes the positives with the teacher while the negatives are computed with the student model (solid circles). If the features obtained with the teacher are scattered and static (the teacher is frozen), those obtained with the student are more clustered and can be learned during the current task. Best viewed in color.

respectively. We empirically validate that the intuition of the NSPT loss is beneficial in the ablation studies of section 2.6.6.

Supervised Multimodal Contrastive Loss. This loss is introduced for two reasons. First of all, since during the first task (no CL) the NSPT loss is not computed (i.e., we do not have a teacher yet), this means that the projector layers of the model are not trained. This would be a problem from the second task onwards in that the student would distill the knowledge from the teacher with randomly initialized projectors. Second, we want to exploit the multimodal nature of our SLU CIL setting. Consequently, we introduce a multimodal (MM) loss that aims to align audio and text representations belonging to the same new class, and thus training the projectors of the model from the very beginning. This alignment is achieved via a supervised multimodal (i.e., audio-text) contrastive learning objective where feature representations of samples sharing the same intent token are attracted while the others are pushed away. Similar to [121], we use the [CLS] text token (y_{cls}) for performing the multimodal alignment. Furthermore, following [33], we always treat the rehearsal samples as negatives, preventing them from being anchors during the learning process. This design choice is buttressed by two motivations: **1)** rehearsal data have been learned by the previous model already and are preserved via the NSPT loss, and **2)** we encourage the model to produce clusters for the new data that are separated from those of the rehearsal data. The MM loss is defined as:

$$\mathcal{L}_{\text{MM}} = \sum_{k \in \mathcal{I}_c} \frac{-1}{|\mathcal{P}(k)|} \sum_{p \in \mathcal{P}(k)} \left[\log \frac{\exp(\mathbf{a}_k^n \cdot \mathbf{t}_p^n / \tau)}{\sum_{i \in \mathcal{I}} \exp(\mathbf{a}_k^n \cdot \mathbf{t}_i^n / \tau)} + \log \frac{\exp(\mathbf{t}_k^n \cdot \mathbf{a}_p^n / \tau)}{\sum_{i \in \mathcal{I}} \exp(\mathbf{t}_k^n \cdot \mathbf{a}_i^n / \tau)} \right]. \quad (2.18)$$

The first term of the internal loss is the audio-to-text component, whereas the second is the text-to-audio component [255]. The presence of both directions ($A \rightarrow T$ and $T \rightarrow A$) makes the MM loss symmetric. All in all, COCONUT minimizes the following loss:

$$\mathcal{L} = \mathcal{L}_{\text{ASR}} + \lambda_{\text{MM}} \mathcal{L}_{\text{MM}} + \lambda_{\text{NSPT}} \mathcal{L}_{\text{NSPT}}, \quad (2.19)$$

where lambdas are loss-specific weights. Note that during the first task $\mathcal{L}_{\text{NSPT}}$ is not computed.

2.6.4 Experimental Setup and Implementation Details

Datasets and CIL Setting. We evaluate COCONUT on two SLU datasets: the Fluent Speech Commands (FSC) [147] and the Spoken Language Understanding Resource Package (SLURP) [19]. FSC includes 30,043 English utterances, recorded at 16 kHz, resulting in 31 intent classes in total. The SLURP dataset comprises around 56 hours of audio of people interacting with a home assistant (*slurp_real*), with the addition of 43.5 hours of synthetic data (*slurp_synth*). It is considered the most challenging SLU dataset due to its lexical complexity. Each utterance is annotated with 3 semantics: scenario, action, and entity. The pair (scenario, action) defines an intent. Overall, there are 18 scenarios and 69 intents. For our experiments, we only perform intent classification. Following [32], we use the scenario labels as splitting criterion to define the CIL setting. We experiment on two configurations: 1) the datasets are partitioned into 3 tasks, each task comprising 6 scenarios for SLURP (denoted as SLURP-3), and 10 intents for FSC (FSC-3); 2) a more challenging configuration with 6 tasks, each task including 3 scenarios for SLURP (SLURP-6), and 5 intents for FSC (FSC-6).

Implementation Details. For both datasets, the text encoder is a standard text embedding layer with size 768. For the audio encoder, we use a Wav2vec 2.0 base model [14] pre-trained and fine-tuned on 960 hours of Librispeech for SLURP ($\sim$ 94.3M parameters), while we use DistilHuBERT base [35] for FSC ($\sim$ 23.5M parameters). Both encoders have hidden sizes of 768. Since FSC is a less challenging dataset than SLURP, we found that a smaller pre-trained encoder is sufficient to achieve state-of-the-art results. Moreover, experimenting with diverse architectures helps evaluate the generalizability of our proposed method. As in [180], we employ linear projection layers to map from each encoder's representation to the audio-text embedding space, whose dimension is 512. The ASR decoder is transformer-based with 6 layers, hidden size equal to 768, 8 attention heads, and the dimension of the feedforward layers is 2048. We set the temperature τ to 0.1 for both NSPT and MM loss. We train our models with AdamW optimizer [146], and we set the learning rate $= 5e^{-4}$ for the text encoder, the ASR decoder and the classifier, while for the audio encoder we use a smaller learning rate, $5e^{-5}$, because it is pre-trained.

For the tokenization we apply Byte-Pair Encoding (BPE) [191] for SLURP, with a vocabulary size of 1000 and BPE dropout equal to 0.1, whereas for FSC, given the limited number of unique words, we use word tokenization, resulting in 139 tokens. BPE automatically assigns to each intent a dedicated token, whereas for FSC we manually add the intent tokens. Regarding the weight coefficients, we set λ_{MM} to 0.1, and similarly to [63,232] we set λ_{NSPT} to $\frac{L_p}{L_p+L_n}$, where L_p and L_n count the number of past and new classes.

Baselines. Apart from the standard **offline** (1 task, no continual) and **fine-tuning** (no CL strategies) baselines, we compare COCONUT against standard **experience replay** (ER) methods with *random* and *iCaRL* [182] sampling strategies. We note that ER is already a strong baseline for FSC and SLURP. We also point out that adapting standard CL strategies to our setting is not trivial as they are usually proposed for classification tasks

Table 2.4: Results in terms of Average Accuracy (↑), Last Accuracy (↑), and Average WER (↓) for different strategies on FSC and SLURP datasets. All CL methods exploit a buffer whose size is 1% of the training dataset. **Bold** and underscore numbers denote the best and second best method for a specific setting and metric, respectively. We show in the last row that COCONUT and S-KD can be used together, leading to the best results. For simplicity, the values of the last row are not in bold even though attain the best results.

Setting →	**FSC-3**			**FSC-6**			**SLURP-3**			**SLURP-6**		
Metric → **Method ↓**	**Avg Acc**	**Last Acc**	**Avg WER**	**Avg Acc**	**Last Acc**	**Avg WER**	**Avg Acc**	**Last Acc**	**Avg WER**	**Avg Acc**	**Last Acc**	**Avg WER**
Offline	99.28	-	0.48	99.28	-	0.48	84.41	-	17.65	84.41	-	17.65
Fine-tuning	49.13	17.61	36.37	29.92	7.59	54.66	46.65	18.42	28.32	31.90	10.57	34.79
ER rand	79.17	69.81	15.87	68.61	63.71	24.04	71.44	61.88	21.25	66.57	58.22	24.50
ER iCaRL	82.04	74.00	13.45	69.76	64.12	23.22	71.94	63.22	21.06	68.08	62.29	26.05
T-KD	82.11	75.43	12.95	69.08	64.73	23.82	72.44	62.43	21.19	66.95	60.47	**24.26**
A-KD	84.79	78.12	11.54	73.54	67.05	20.36	72.10	63.84	**20.67**	68.52	62.51	24.29
S-KD	84.29	75.31	12.39	73.65	67.71	21.27	**74.28**	**65.95**	21.26	69.91	63.22	**24.26**
COCONUT	**86.39**	**80.21**	**11.08**	**77.09**	**73.80**	**19.05**	72.75	64.62	21.25	**70.17**	**63.66**	24.29
COCONUT+S-KD	87.64	80.45	10.49	77.57	74.01	18.47	75.58	67.39	20.61	71.91	65.41	24.16

and not for auto-regressive tasks. Plus, we report two methods proposed in [32] that combine rehearsal and KD principles: audio-KD (**A-KD**) that applies the KD on the audio features of the rehearsal samples, and seq-KD (**S-KD**) that, at the end of the current task, stores the text transcriptions computed with beam search only for the rehearsal samples and use them as pseudo-transcriptions for the next task. This method operates on the ASR decoder. For the sake of completeness, we also report text-KD (**T-KD**), the text counterpart of the A-KD.

Metrics. Following [63], we report the results in terms of the *Avg Acc*, which is the average of the intent accuracies after each training task, and the *Last Acc*, which is the intent accuracy after the last task. We also report the *Avg WER*, the average of the Word Error Rate (WER) of the extended transcription after each task.

2.6.5 Main Results

In the first two rows of Table 2.4, we include the upper and lower bounds represented by the offline learning (which is in line with the state-of-the-art)

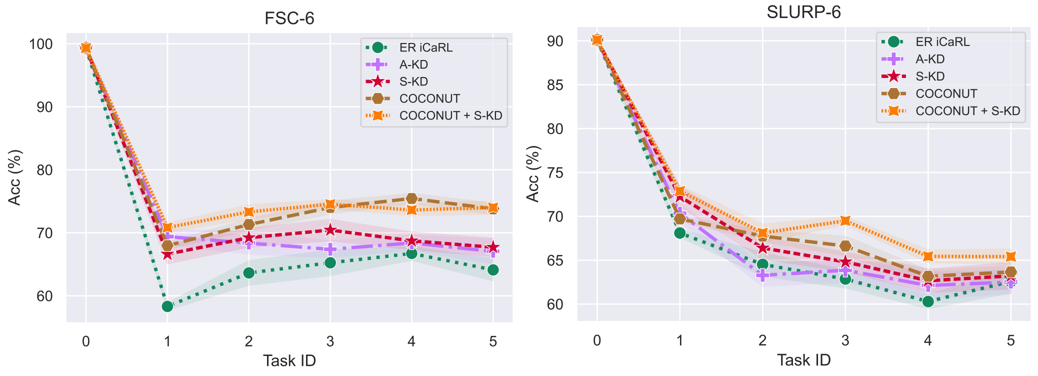


Figure 2.11: *Left*: the trend of the intent accuracy on the observed tasks for the FSC-6 setting. *Right*: the trend of the intent accuracy on the observed tasks for SLURP-6.

and fine-tuning approaches. For the fine-tuning approach, we can notice how catastrophic forgetting deteriorates the knowledge of the prior classes. We then include ER baselines with buffer capacity equal to 1% of the dataset size. From these results we can see that ER-based methods achieve good results for all metrics and configurations, confirming themselves as solid baselines. For FSC, COCONUT outperforms the other baselines by a significant margin, in terms of both accuracy and WER. Its combination with the S-KD leads to additional improvements (last row).

If we turn our focus to SLURP we see that, for the setting with 3 tasks, S-KD turns out to be the best approach in terms of intent accuracy, followed by COCONUT. For the WER, all the methods achieve similar performance and do not provide significant enhancements. We speculate that, as only some words are task-specific while the others are spread across multiple tasks, the text modality is less affected by forgetting. It is also compelling to note that the A-KD always achieves better performance than T-KD, a trend that will also be observed for the NSPT loss in the ablation studies. For SLURP-6, COCONUT slightly surpasses S-KD in terms of accuracy, and performs on par with the others for the WER metric. This indicates that COCONUT scales properly with the number of tasks. Additionally, we point out that, for SLURP, COCONUT provides less noticeable improvements than FSC. This can be attributable to the higher complexity of the dataset due to its

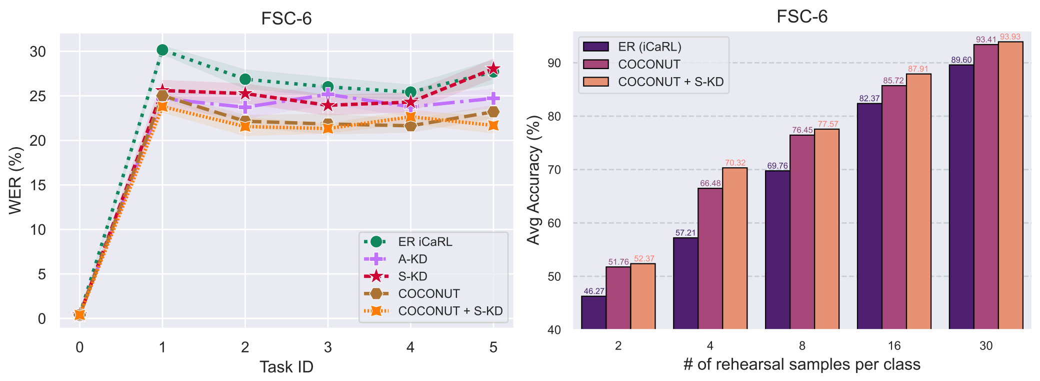


Figure 2.12: *Left*: the trend of the WER on the observed tasks for the FSC-6 setting. *Right*: the accuracy of COCONUT and other methods as a function of the memory size.

larger dictionary and to the larger number of intents with respect to FSC (69 vs. 31). Finally, similar to FSC, the combination of COCONUT with S-KD attains the best results, confirming that fighting forgetting both at the encoders and ASR decoder is an effective solution.

In Fig. 2.11 we illustrate the trend of the intent accuracy after each task for FSC-6 and SLURP-6. For FSC-6, COCONUT outperforms the other baselines by a large margin after each task. For SLURP-6, COCONUT has a similar trend as S-KD, and their combination leads to a noteworthy boost in performance. On the left part of Fig. 2.12 we also show the trend of the WER task by task.

2.6.6 Ablation Study

.

Is COCONUT effective when we vary the buffer memory size? In the right side of Fig. 2.12, we study the trend of COCONUT for different quantities of rehearsal samples per class for the setting FSC-6. Note that 8 samples per class is equivalent to a buffer capacity of 1% of the entire training dataset. The maximum gain provided by COCONUT with respect to the ER baseline is reached for 4 and 8 samples per class (9.27 and 6.69, respectively), while for the extreme cases of 2 and 30 samples, the gap is reduced. This is

Table 2.5: The accuracy of COCONUT and other methods as a function of the memory size for the setting SLURP-6.

Method	**Examples per class**		
	650	1260	2500
iCaRL	59.94	61.87	63.38
COCONUT	68.08	70.17	71.91
COCONUT + S-KD	**70.15**	**71.41**	**72.10**

explained by the fact that when few samples are stored for each class, the effect of the NSPT loss is highly reduced given its reliance on the rehearsal data, whilst in the opposite case the abundance of rehearsal data makes the ER baseline already strong, thereby improving it becomes more challenging. Regarding the latter case we note that when we increase the buffer memory size, we implicitly move toward the offline setting (the upper bound), which is not the objective of this work. In addition to this, we provide a similar study for SLURP-6 in Table 2.5. Note that 1260 samples corresponds to 1% of the training data, which is the % we used for our main results. Similar to what we obtained for the FSC dataset, we see that, as we increase the number of retained samples to 2500, the gain brought by COCONUT and its combination with S-KD is a bit smaller but still significant, and this happens because the iCaRL method becomes a stronger and stronger baseline as we increase the % of data. Also, we notice that adding the S-KD approach is more beneficial when we have fewer samples in the memory since the task is way more challenging.

Ablation on the NSPT Loss. In Table 2.6 we evaluate the difference in performance between the standard NTPT loss and our proposed NSPT loss and some of its variants. Specifically, we study two design properties: **1)** which samples should be used as anchors? **2)** Should the rehearsal negatives be computed using the teacher model rather than the student, unlike the negatives coming from the new task? Regarding point **(1)**, we study the case where the anchor samples are both the rehearsal data (our proposed design) *and* the new data. This means that in the outer sum of Equation

Table 2.6: Ablation on the use of NSPT and NTPT losses.

Dataset →	**FSC-6**		**SLURP-6**	
Metric → **Method** ↓	Avg Acc	Last Acc	Avg Acc	Last Acc
ER iCaRL	69.76	64.12	68.08	62.29
MM	71.12	67.76	68.78	62.94
MM + NTPT	74.05	67.61	68.91	62.57
MM + NSPT-AA	76.30	72.34	69.74	62.54
MM + NSPT-AN	66.37	63.89	64.72	56.84
MM + NSPT	**77.09**	**73.80**	**70.17**	**63.66**

2.17 the samples are picked from $\mathcal{I}$. Note that this design choice requires to compute the loss for all samples in the dataset, thus incurring an appreciable increase in the computational cost. We denote this variant where we **A**blate the **A**nchor design as NSPT-**AA**. As for the second point, we compute the negatives coming from the rehearsal memory using the teacher (the teacher has seen those classes in the previous tasks), whereas the samples from the current task are computed with the student model. The denominators of Equation 2.17 become (we use $\mathbf{z}$ to refer to both $\mathbf{a}$ and $\mathbf{t}$): $\sum_{i\in\mathcal{I}_c} \exp(\mathbf{z}_k^n \cdot \mathbf{z}_i^n/\tau) + \sum_{h\in\mathcal{I}_r} \exp(\mathbf{z}_k^n \cdot \mathbf{z}_h^{n-1}/\tau)$. We call it NSPT-**AN** (**A**blate **N**egatives).

Looking at Table 2.6, we see that for FSC-6, the use of our proposed NSPT loss gives a considerable improvement over the NTPT loss in terms of all three considered metrics. For SLURP-6, the trend is maintained, and now the NTPT even brings a small deterioration over the MM baseline in terms of Last Acc. Also, the MM loss alone contributes positively over the ER baseline for both settings. We recall that it is not possible to study the individual contribution of the NSPT loss because, without the MM loss, the teacher projectors are randomly initialized during the second task (see section 2.6.3). Furthermore, we observe that the design choices of **(1)** and **(2)** are crucial to obtaining superior performance. Regarding the NSPT-**AA** loss, the model is less sensitive to this design choice. However, note that this loss is more expensive as it requires extra computational cost owing to the use of all samples in a mini-batch for its computation, thus making it

Table 2.7: Ablation study of the MM (upper part) and NSPT (bottom part) components. **CLS**: whether only the intent class token is used; **Anchor**: whether ER data are excluded from the anchors. $\mathcal{L}_{\mathbf{A}}/\mathcal{L}_{\mathbf{T}}$: whether the audio/text component of NSPT loss is used.

CLS	**Anchor**	$\mathcal{L}_{\text{A}}$	$\mathcal{L}_{\text{T}}$	**Avg Acc**
				70.10
✓				70.49
	✓			71.09
✓	✓			**71.12**
✓	✓	✓		76.84
✓	✓		✓	73.11
✓	✓	✓	✓	**77.09**

less appealing than our proposed NSPT loss. Instead, the use of the NSPT-**AN** yields a severe degradation in the performance. We suspect that this happens because mixing the teacher and student at the denominators makes the learning process more complex as feature representations of different models interact, inducing more interference and thus leading the model to make more mistakes.

Ablation on the MM Loss. Finally, in Table 2.7 we study the design properties of the MM loss on FSC-6, and with its best configuration, we determine the individual contribution of the audio and text components to the NSPT loss. For the MM loss, we see that using the intent token and preventing the ER data from being anchors brings additional improvements. For the NSPT loss, as was evident for the A-KD and T-KD, with the former giving better results, here we also discover that the audio component is predominant. Plus, the concurrent use of both components brings a moderate increase in accuracy, and this is due to the alignment between audio and text via the MM loss.

On the Impact of the Temperature Parameter. In this section we analyze the role of the temperature parameter in the CIL process for the MM loss (see Equation 2.18) on the FSC-6 setting. We first try to set the value beforehand (0.07, 0.1, 0.2), and then we make the temperature a learnable hyperparameter (initial value is 0.07). Results are reported in

Table 2.8: Ablation study of the temperature τ for the MM loss. We experiment on FSC-6 by setting τ beforehand and making it a learnable hyperparameter as is common practice in offline settings [180]. The light-blue row corresponds to the value we used for our experiments.

Metric $\rightarrow$ **Temp(τ)** $\downarrow$	Avg Acc	Last Acc	Avg WER
0.07	71.06	64.75	**22.07**
0.1	**71.12**	**67.76**	22.88
0.2	71.01	62.35	22.78
`Learnable`	69.05	66.33	24.57

Table 2.8. We can observe that $\tau = 0.1$ is the best configuration for the accuracy metric. Note that, however, the model does not seem very sensitive to the temperature for the Avg Acc, whereas the Last Acc is more influenced. Since the Avg Acc does not change much across the three configurations, yet the Last Acc sways much more, this means that for $\tau = 0.1$ the model struggles more during the initial tasks, but it performs better towards the end of the learning process. On the other hand, learning τ task by task does not seem to be the right choice as the Avg Acc and WER metrics deteriorate with respect to the other three configurations where it is fixed. In fact, we observed that during the first tasks, the model is learning the optimal value for τ until it finds it (this value approximately lies in the range $0.134-0.142$). This initial transitional phase penalizes the accuracy of the first tasks, which in turn leads to a deterioration in the Avg Acc metric.

2.6.7 Final Remarks and Future Work

In this work, we presented COCONUT, a CIL approach that exploits experience replay and contrastive learning paradigms to mitigate forgetting for the problem of E2E class-incremental SLU using a seq-2-seq model. On the one hand, it preserves the previously learned feature representations via an ad-hoc supervised contrastive distillation loss, on the other it contributes to aligning audio and text representations, thus resulting in more transferable and robust to catastrophic forgetting representations. We show that COCONUT outperforms the other baselines and that synergizes with other KD techniques operating on the decoder side. We finally dissect the design choices of COCONUT through specific ablation studies, showcasing that each component is pivotal to attain the best results. Regarding future directions, we note that in principle COCONUT can be exploited in other multimodal settings such as audio-vision or vision-language. Therefore, it would be interesting to study how to exploit COCONUT in other different multimodal scenarios. Also, since these settings usually involve a larger number of classes than ours, we would be able to test how COCONUT scales to the number of tasks.

2.7 Improving Continual Learning of Acoustic Scene Classification Via Mutual Information Optimization [239]

2.7.1 Overview

Acoustic scene classification (ASC) refers to the ability of humans or machines to recognize an environmental sound in a set of acoustic scene classes [18]. Similar to SLU, even though the performance of deep neural network-based models has outperformed humans on this specific task [163], the success of existing ASC models has mostly relied on training with a large fixed set of data and pre-defined acoustic scene classes [18,175,205]. In contrast, humans are intrinsically capable of learning from streams of data with unseen classes, so that they can adapt themselves to the complex environment and evolve their knowledge in a lifelong manner [218].

From the perspective of human perception, recognizing an unseen acoustic scene can be regarded as an ensemble of background noises and sound events [212]. Then humans can associate new sound categories with specific acoustic scenes based on their extensive life experiences. Furthermore, [93] showed that the perception of acoustic scenes in human brains is organized in terms of a general feature match between sounds and the referents in their memory. Therefore, we base our ASC model on the replay-based continual learning approaches, which save a small subset of past data into a memory buffer and replay samples of the memory when it encounters new tasks in the subsequent training process.

Specifically, in this work, we focus on the under-explored domain of continual learning in acoustic scene classification and propose to improve both the training process and the memory selection procedure in our ASC model from the perspective of mutual information (MI) optimization. MI quantifies the amount of information between two variables and has been proven to be helpful in representation learning from an information-theoretic view [13, 22, 92, 166, 210]. We base the optimization of MI on an architecture with a feature extractor followed by a classifier, similar to the architecture used in 2.4.2. Since feature extraction is to disentangle many distinct but informative factors from the data [21], we would like the feature extractor to learn generalized representations of the input audio that is agnostic to acoustic scene classes. Meanwhile, we would like the classifier to learn to map the extracted representations to the correct classes. This section demonstrates that mutual information can help the feature extractor learn task-agnostic knowledge while helping the classifier learn task-specific knowledge. For the feature extractor part, we first present that it is theoretically sound to learn task-agnostic knowledge by maximizing the MI between the feature representations of the original input and an augmented acoustic scene of the same input. For the classifier part, we show that by selecting the memory samples with a combination of surprise and learnability criteria, the samples are expected to be both representative and informative to boost the continual learning performance of the ASC model. We evaluate our method on multiple datasets and continual learning metrics, showing that it can not only decrease the forgetting effect by learning task-specific knowledge, but improve the generalizability of the model by learning task-agnostic knowledge as well.

2.7.2 Method

Problem Statement. Similar to the previous sections, we study a CIL scenario, where now new classes of *acoustic scenes* keep appearing in continuous streams of data. Since CIL does not have access to task identities during inference time, its objective is to build a holistic classifier among all of the seen classes by making use of the label information only.

In our context, a task is defined by a set of train and test data that follows a similar distribution, and in practice, it usually refers to a new set of data that contains data in different classes. For example, one task may consist of acoustic scenes from various vehicles, and another task may include sound events from animals. Consider we have the data streams $\mathcal{X} = \{X_t\}_{t=1}^{T}$ and its corresponding labels $\{Y_t\}_{t=1}^{T}$, where X_t indicates the data at task t and T is the total number of tasks. Note that $Y_{t_i}^{gt} \cap Y_{t_j}^{gt} = \emptyset$ for all $t_i \neq t_j$ under this setting. For the modeling, we use an architecture of a feature extractor Φ and a classifier Θ, where the output of the feature extractor is $\Phi(X) = Z$, which is the feature representation of X and also the input of the classifier such that $\Theta(Z) = Y$, where Y indicates the predicted logits.

At the end of each task t, we inject samples of the input X_t into a memory buffer with a fixed size M, and the memory will be used to select samples for the purpose of replay when the model is learning on subsequent tasks. We will demonstrate the comparison between different sample selection strategies in section 2.7.3.

Augmented Acoustic Scenes. Our mutual information optimization relies on the comparisons between different augmented representations of acoustic scenes, which are also called pseudo-labeled samples. In MI estimation, the augmentations of the same input will be regarded as positive pairs, while those of different inputs are taken as negative pairs, such that the dependency between positive pairs will be maximized and that between negative pairs will be minimized [166]. Following the notations in the previous subsection, we will denote Z as the feature representation of the original input X, and Z' as the encoded feature of an augmentation of input X'. In this work, we simulate the pseudo-labeled samples through different augmentation methods. More specifically, we choose to add Gaussian noise, apply band-stop filtering, or invert along the time axis to perform multiple types of augmentations.

Modeling and Memory Selection via Mutual Information Optimization. By employing an arbitrary model architecture of a feature extractor followed by a classifier, we show that our mutual information optimization can be applied to both modules of the model. We would like the feature extractor Φ to learn task-agnostic knowledge to produce effective latent representations of the input audio, while the classifier Θ to learn task-specific knowledge to map the learned representations to specific acoustic scene classes.

Feature extractor part: To let the feature extractor learn task-agnostic knowledge, we need to guarantee that the encoded representations preserve sufficient information from the original inputs regardless of their classes. Therefore, we will want to maximize the MI between X and Z, so that Z will preserve generic information from X in the modeling of the feature extractor. To better estimate the MI, we have augmented acoustic scenes X' and its corresponding latent representation Z'. Assuming that Z and Z' are conditional independent given data X, we have:

$$\begin{aligned} I(X;Z) &= H(Z) - H(Z|X) \\ &= H(Z) - H(Z|X, Z') \\ &\geq H(Z) - H(Z|Z') = I(Z;Z'), \end{aligned} \tag{2.20}$$

where $I(\cdot,\cdot)$ indicates the mutual information between two variables and $H(\cdot)$ denotes the Shannon entropy or conditional entropy given another random variable. We use the property of conditional independence from Line 1 to Line 2, and the definition of conditional entropy from Line 2 to Line 3. We use the property of conditional independence from Line 1 to Line 2, and the definition of conditional entropy from Line 2 to Line 3. Therefore, we have $I(X;Z) \geq I(Z;Z')$ from Eq. 2.20, which means that maximizing the MI between Z and Z' is equivalent to maximizing the lower bound of the MI between input X and the encoded features Z.

Taking a further step, as from [166], the mutual information between Z

and its augmentation Z' can be estimated through the InfoNCE (NCE stands for noise-contrastive estimation) loss as the lower bound, i.e.,

$$I(Z, Z') \geq \log N + \underbrace{\frac{1}{N} \sum_{i=1}^{N} \log \frac{(f(z_i, z_i')/\tau)}{\sum_{j=1}^{N} (f(z_i, z_j')/\tau)}}_{\triangleq \mathcal{L}_{NCE}(Z,Z')}, \tag{2.21}$$

where z_i is the representation of an individual sample in the batch. z_i and z_i' are regarded as positive sample pairs since they originated from the same sample x_i, and all other z_j's in the batch where $i \neq j$ are regarded as negative pairs. f denotes the exponential of similarity function and τ is the temperature parameter. N is the batch size of the samples, and when it becomes larger, $I(Z, Z')$ will close its gap to the lower bound. Therefore, $I(Z, Z')$ is lower bounded by the InfoNCE loss and we will use the second term on the right-hand side as the approximation of MI in our implementations. Overall, in addition to the task supervision loss, we add the MI estimation to the objective function to train the model as

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}}(\Theta(\Phi(X)), Y) + \lambda \mathcal{L}_{\text{NCE}}(Z, Z'), \tag{2.22}$$

where $\mathcal{L}_{\text{CE}}(\cdot, \cdot)$ denotes the cross entropy loss between the predicted and the ground-truth class logits, and λ is the hyperparameter.

Classifier part: The classifier takes the latent representation Z as input and predicts the logits Y. In contrast to the feature extractor where the primary goal is to capture task-agnostic features, since we would like the classifier to learn task-specific knowledge, we need to wisely select the samples from the memory, such that they can not only bring extra information but also make sure the new information can be learned by the model. Two criteria, *surprise* and *learnability*, can be formalized by measuring the predictive distribution of the new sample with respect to those in the memory to decide its usefulness [201]. In our work, we measure the information carried by memory samples by estimating the MI between the encoded representation Z and the predicted logits Y. In this case, it differs from the original InfoNCE loss with additional information Y. Since we would like to train the classifier to learn task-specific knowledge given the label information, we use Y additionally to determine positive and negative sample pairs. If we further incorporate the label information Y into Eq. 2.21, for each sample feature z_i, we consider z_k as positively paired sample for all $y_k = y_i$. In other words, samples with the same predicted labels with respect to sample i will be constructed as positive pairs. Meanwhile, all pairs of the original and augmented representations z_k and z'_k, along with the augmented representation z'_i, are regarded as positive pairs with respect to z_i, while others are taken as negative pairs, i.e.,

$$\begin{aligned}\mathcal{L}_{\text{NCE}}(Z, \{Z'\}, Y) =& \frac{1}{N}\sum_{i=1}^{N}[\frac{1}{\sum\limits_{k=1}^{N} \mathbb{1}(y_k = y_i)} \sum_{y_k=y_i} (\sum_{\hat{z_i}\in\mathcal{S}_{z_i}} \log(f(z_i, \hat{z}_i)/\tau) \\ &- \log\sum_{j=1}^{N}(\sum_{\hat{z_j}\in\mathcal{S}_{z_j}} f(z_j, \hat{z_j})/\tau))],\end{aligned} \tag{2.23}$$

where $\mathcal{S}_{z_i}$ indicates the set of all of the original and augmented views of the sample z_k that has the same label with z_i, along with the augmented view of itself z'_i. The $\mathbb{1}(\cdot)$ function returns 1 if the condition is true and 0 otherwise. Note that here we use the notations of sets, $\{Z'\}$, to indicate that more than one type of augmentation techniques can be considered.

When we are selecting from the memory at task t, we would like to select the samples that are both representative and informative. It not only needs to carry new information to the existing model, but also should have a high learnability instead of becoming an outlier. To achieve this, we assign a score to the samples and select the samples with the highest scores as:

$$\text{score}_t(Y, Z) = -\mathcal{L}_{\text{NCE}}(Z_{t-1}, \{Z'_{t-1}\}, Y_{t-1}) + \mathcal{L}_{\text{NCE}}(Z_t, \{Z'_t\}, Y_t). \tag{2.24}$$

The first term on the right-hand side of Eq. 2.24 indicates that we would like to be more likely to select samples that minimize the mutual information between Z and Z', given the class logits by the previous model at task $t-1$. In other words, the samples that are more surprising to the model are favored. Similarly, the second term indicates that the samples with higher learnability are more probably to be sampled, since they maximize the MI between Z and

Z' given Y by the current model, which aligns with our objective function. In this way, we will be more likely to select samples that are both representative and informative, so that the model can effectively recall past knowledge and learn from new information at the same time.

2.7.3 Experiments

Datasets. We use the TAU Urban Acoustic Scenes 2022 [91] and Environmental Sound Classification (ESC)-50 [176] as our datasets. TAU Urban Acoustic Scenes consists of 10 classes of acoustic scenes in total, with around 1000 samples for each class. Its acoustic scene classes are mainly about transportation and city noises. ESC-50 is a smaller dataset that is made up of 5-second-long recordings in 50 semantical classes, with 40 samples for each class. This dataset mainly covers the sounds from animals, humans and daily activities, etc. The diversity of these datasets helps us validate the effectiveness of our continual learning methods under different settings. For the task splitting, we split the 10 or 50 classes in the two datasets into 5 sequential tasks $\{X_t, Y_t^{gt}\}_{t=1}^{5}$. Each task contains 2 or 10 different classes respectively.

Baselines. We evaluate the continual learning performance of acoustic scene classification with multiple baseline approaches together with our proposed method as in Tables 2.9 and 2.10. *Fine-tune* means the offline training without any continual learning approaches performed, which is the lower bound of our performance. *Random* is to randomly select samples from the memory with an equal probability. *Herding* indicates herding the embeddings of samples and selects those who are closest to the center of their corresponding class [182]. *GSS* refers to gradient-based sample selection, which aims to maximize the diversity of the gradients of the samples in the memory buffer [8]. *Uncertainty* calculates the uncertainty score of each sample based on the prototypes from the herding method, and selects the samples that the model is less confident of [234]. We will compare the performance of our proposed method with these baselines as different memory selection mechanisms in replay-based continual learning methods.

Experimental Setup. For our feature extractor, we use a Temporal

Convolutional Network (TCN) [16] as the feature extractor and a linear layer as the classifier. The feature extractor takes in the log-Mel spectrogram of the audio input, computed by a Hanning window with the window length of 25ms and the hop length of 10ms. The latent representation Z is represented as 100-dim embedding vectors, which are used to compute the scoring function to sample from the memory. We train the model for 50 epochs for each task with an Adam optimizer [115] and an initial learning rate of $5e^{-4}$. We use the temperature $\tau = 1.0$ for all the experiments.

Evaluation Metrics. Aside from average classification accuracy (Acc), we use backward transfer (BWT) and forward transfer (FWT) as the evaluation metrics [57] to show that our method helps not only learn task-agnostic knowledge, but also preserve the task-specific knowledge. BWT measures the influence of learning task t on the accuracies of all previous tasks $i < t$. The calculation of BWT at task t is defined as:

$$\mathrm{BWT}_t = \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} (a_{t,i} - a_{i,i}), t \in \{2, \cdots, T\}$$

where $a_{i,j}$ denotes the accuracy of task j after learning task i. On the contrary, FWT measures the generalizability of the model by computing the influence of learning task t on the accuracies of future tasks. From [145], we have:

$$\mathrm{FWT}_t = \frac{1}{t-1} \sum_{i=2}^{t} (a_{i-1,i} - \bar{a}_i), t \in \{2, \cdots, T\}$$

where $\bar{a}_i$ indicates the test accuracy at task i with random initialization. Overall, a higher BWT score means a smaller forgetting effect of the model on past task-specific knowledge, while a higher FWT score means a higher generalizability of the model on task-agnostic knowledge to benefit unseen tasks.

Results and Discussion. The experimental results on the TAU Urban Acoustic Scene dataset are shown in Table 2.9. The row of *fine-tune* suffers from catastrophic forgetting, and its accuracy is close to a random guess since we have 5 tasks in total. It is the lower bound of our continual learning

Method	**Memory Size**	**Acc ↑**	**BWT ↑**	**FWT ↑**
fine-tune	-	20.4	-56.0	0.0
Random	0.2k	42.8	-28.5	49.8
	0.5k	49.8	-27.8	54.3
	1k	52.6	-27.0	59.2
Herding [182]	0.2k	51.6	-26.9	56.0
	0.5k	54.3	-26.3	63.3
	1k	56.2	-24.8	65.2
GSS [8]	0.2k	51.9	-25.3	56.5
	0.5k	54.6	-25.8	62.9
	1k	56.1	-24.6	63.7
Uncertainty [235]	0.2k	55.9	-24.5	63.8
	0.5k	57.6	-23.7	67.5
	1k	58.9	-22.8	69.0
MIO (Ours)	0.2k	58.0	-23.5	64.7
	0.5k	60.7	-22.9	69.1
	1k	**64.1**	**-22.5**	**74.8**

Table 2.9: Results for continual learning on TAU Urban Acoustic Scenes with different memory selection methods and sizes.

performances. For the rest of the rows, we can observe that our proposed mutual information optimization (MIO) method achieves the highest score of Acc, BWT, FWT than other memory selection methods, which indicates that it can not only retain the task-specific knowledge in the past, but generalize the task-agnostic knowledge to future unseen classes as well. Another observation is that our method benefits more from a larger memory size, with a higher performance gain from a smaller memory size to a larger one. This intuition aligns well with the property of the estimates of mutual information in Eq. 2.21, where the estimated MI approaches its lower bound when the number of samples N becomes larger.

Table 2.10 presents the results on the ESC-50 dataset with the same set of evaluation metrics. Overall, we can observe a similar tendency with the results in Table 2.9, except that the accuracies become lower because there

Method	**Memory Size**	**Acc ↑**	**BWT ↑**	**FWT ↑**
fine-tune	-	19.1	-58.7	0.0
Random	0.2k	22.5	-52.5	26.6
	0.5k	24.6	-49.7	27.3
	1k	26.2	-47.6	29.7
Herding [182]	0.2k	47.5	-30.8	49.3
	0.5k	49.3	-28.7	50.6
	1k	50.8	-27.9	52.2
GSS [8]	0.2k	48.8	-30.3	49.8
	0.5k	49.6	-29.3	50.8
	1k	50.3	-28.2	51.9
Uncertainty [235]	0.2k	50.9	-28.9	51.6
	0.5k	51.8	-27.6	53.1
	1k	52.9	-27.1	53.9
MIO (Ours)	0.2k	52.1	-28.5	53.4
	0.5k	53.7	-27.4	55.9
	1k	**55.3**	**-26.5**	**57.3**

Table 2.10: Results for continual learning on Environmental Sound Classification-50 dataset with different memory selection methods and memory sizes.

are 50 classes in total and less number of samples per class. We also plot the change of the accuracies over the sequential tasks in Figure 2.13. From the figure, we can observe that our method has the least forgetting effect with a smaller decrease in accuracy. In contrast, *fine-tuning* has the largest drop with accuracy close to $\frac{1}{t}$, where t is the number of tasks that the model has experienced. It is noteworthy that from task 2 to task 3, some continual learning methods have an increased accuracy instead of a decreasing one. We speculate that this phenomenon is due to the shared properties of the acoustic scene classes that these tasks consist of.

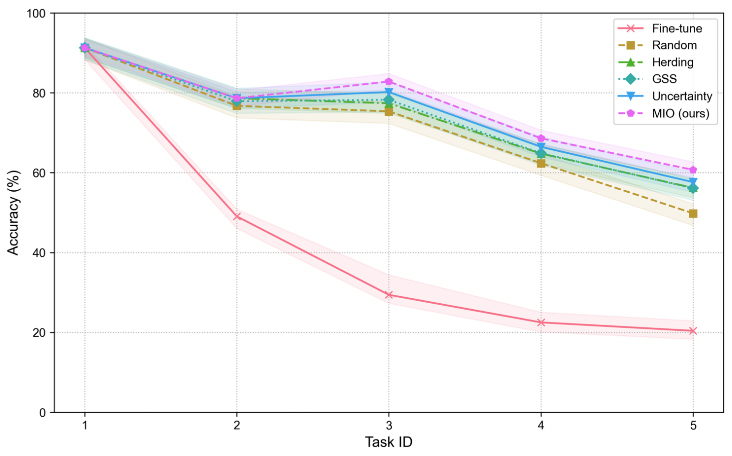


Figure 2.13: Average Acc (%) over tasks in sequential order for different methods. The accuracies are calculated on the test sets of the seen tasks so far.

2.7.4 Final Remarks

In this work, we presented a replay-based continual learning approach in the acoustic scene classification task with mutual information estimation. We propose to optimize different levels of the model to learn task-agnostic and task-specific knowledge from the perspective of mutual information, and select samples from the memory buffer that are both representative and informative. We demonstrate that our proposed method outperforms other continual learning algorithms by both a lower forgetting effect and higher generalizability.

2.8 Summary of the Contributions

In this chapter, we tackle the problem of SLU under a class-incremental learning setting. We start by defining a CIL setting for the problem of intent classification, and we propose a method that exploits both rehearsal and knowledge distillation to mitigate catastrophic forgetting. Then, we extend our investigation by also encompassing the task of slot-filling. To do so, we first define a CIL setting for the SLURP dataset, the largest and most diverse dataset in terms of lexical complexity for SLU. Since we treat the task of slot-filling as a seq2seq problem, we introduce three different knowledge distillation methods that combat forgetting at both the encoder and decoder levels. Furthermore, we propose COCONUT, a new method that makes use of knowledge distillation, contrastive learning, and rehearsal to endow seq-to-seq models with lifelong learning capabilities. Finally, we target the problem of continual learning in acoustic scene classification and we propose a method that improves both the training process and the memory selection procedure using the concept of mutual information.

Chapter 3

Parameter-Efficient Transfer Learning of Audio and Speech Foundation Models

3.1 PETL Taxonomy

Parameter-Efficient Transfer Learning (PETL), or Parameter-Efficient Fine-tuning, refers to the process of adapting a pre-trained model to a specific task or domain while minimizing the number of additional parameters and computational resources required [88]. This approach is particularly advantageous when working with large-scale language models, as fine-tuning them from scratch (i.e., full fine-tuning) can be computationally prohibitive and incur catastrophic forgetting. By adjusting a limited subset of parameters, this technique enables efficient customization of the model to new tasks and domains.

PEFT strategies can be categorized into four main groups [85]:

- **Additive** PETL, which inserts a small amount of trainable parameters to the pre-trained model;

- **Selective** PETL, which updates a subset of the pre-trained model parameters;

- **Reparameterized** PETL, which builds and updates a low-rank parameterization of the model's parameters;
- **Hybrid** PETL, which tries to combine the advantages of multiple PETL approaches.

3.1.1 Additive PETL Strategies

Additive PETL approaches keep the pre-trained backbone of the model intact, introducing only a minimal number of trainable parameters strategically placed within the architecture. During fine-tuning for a specific downstream task, only the weights of these additional parameters are updated, significantly reducing the storage, memory, and computational resource requirements. We discuss popular additive PEFT methods below. *Adapter-tuning* methods insert small modules within the model's layers. The adapter layer generally uses a down-projection to project the input to a lower-dimensional space specified by bottleneck dimension r, followed by a nonlinear activation function, and an up-projection. The concept of adapters in the natural language processing field was first introduced in [97], where the adapter module is placed sequentially after both the self-attention and feed-forward layers (referred to as *Houlsby* adapter). [174], instead, propose to add the adapter layer only after the feed-forward layer to improve the computational efficiency (*Pfeiffer* adapter). Multiple works have proposed to insert *parallel* adapters to enhance the model's parallelism [65, 88, 123, 265]. To improve performance and generalization, various multi-task learning strategies have been proposed for adapters, including *AdapterFusion* [174], *AdaMix* [221], *PHA* [259], *AdapterSoup* [49], *MerA* [89], and *Hyperformer* [155]. These methods typically fuse pre-trained adapters [89, 174] or use a shared hypernetwork ([155]) to store multi-task information, reducing computational costs and the number of trainable parameters.

Prompt-based techniques involve adding learnable soft prompts to the input of the pre-trained model to enhance its performance on specific tasks. This approach avoids modifying the model's core architecture, making it efficient and flexible. Various methods, such as *Prefix-tuning* [128], *p-tuning*

[144], and *Adaptive Prefix Tuning* [256], have been proposed to optimize the learning and application of soft prompts. These methods involve adding learnable vectors to the input sequence or to the attention mechanism of the model. Recent advancements, including *Instance-Dependent Prompt Generation* (IDPG) [233], *Late Prompt Tuning* (LPT) [143], and *Selective Prompt Tuning* (SPT) [264], focus on generating instance-specific prompts to further improve performance. These methods utilize techniques like prompt generators and adaptive gating mechanisms to tailor the prompts to the specific input. To stabilize training and accelerate convergence, methods like *SPoT* [215], *Transferable Prompt Tuning* (TPT) [200], *InfoPrompt* [231], and *PTP* (Prompt Tuning with Perturbation-based regularizer) [42] have been developed. These techniques leverage techniques like prompt transfer, mutual information maximization, and perturbation-based regularization to improve the training process.

Finally, it is worth mentioning techniques like *IA*^{3} [139] and *SSF* [134] that introduce additional parameters during fine-tuning to enhance performance without increasing inference costs. These methods involve rescaling or linearly transforming the activations of the pre-trained model, which can be merged into the model weights during inference.

3.1.2 Selective PETL Strategies

Selective PETL strategies involve selectively fine-tuning a subset of the model's parameters rather than adding new parameters. Several methods have been proposed to identify the most important parameters for fine-tuning. These include *diff pruning* [81], which uses a learnable binary mask to select parameters, *PaFi* [136], which selects parameters based on their magnitude, and *FishMask* [203] and *Fish-Dip* [53], which use Fisher information to identify important parameters. Additionally, LTSFT [9] leverages the Lottery Ticket Hypothesis to identify important parameters.

To improve hardware efficiency, structured parameter masking has been explored. This approach involves organizing parameter masking in regular patterns, as opposed to random unstructured masking. *FAR* [216] employs structured pruning strategies to group and eliminate parameters. *Bitfit* [250] and *Xattn Tuning* [72] focus on fine-tuning specific modules, such as bias parameters or cross-attention layers. *SPT* (sensitivity-aware visual parameter-efficient fine-tuning) [87] identifies sensitive parameters and applies targeted PEFT techniques to optimize the fine-tuning process.

3.1.3 Reparameterized PETL Strategies

Reparametirized strategies build a low-dimensional reparameterization of the original model parameters during training while transforming the weights back to enhance inference efficiency. *Low-Rank Adaptation* (LoRA) [101] is a popular reparameterization technique that approximates the original weight matrices of the model with two low-rank matrices. These low-rank matrices can be fine-tuned to adapt the model to specific tasks without modifying the original pre-trained weights. Despite LoRA being simple, efficient, and competitive, several works have built upon it to enhance the performance. For example, *DyLoRA* dispenses with the challenging requirement of selecting a specific rank value beforehand, and it dynamically adjusts the rank of the low-rank matrices during training to optimize performance. *AdaLoRA* [253] employs singular value decomposition to adaptively prune the low-rank matrices. *SoRA* [58] simplifies the LoRA architecture by removing the orthogonality constraint and using a gating mechanism. *LoRA+* [86] proposes to set different learning rates for the LoRA matrices to enhance performance and fine-tuning speed at no extra computational cost.

Other than LoRA, it is worth describing other reparameterization methods. For example, *Compacter* [111], *KronA* [65] and *KAdaptation* [90] introduce lightweight adapter modules by parameterizing the low-rank matrices using the Kronecker product. *VeRA* [119] employs a single pair of frozen low-rank matrices shared across all layers, with trainable scaling vectors. *DoRA* [142] decomposes model weights into magnitude and direction components, fine-tuning only the directional component using LoRA.

3.1.4 Hybrid PETL Strategies

PETL strategies' effectiveness can vary significantly across different tasks and model architectures. This has led to a growing interest in exploring combinations of PETL methods and optimizing their configurations. *UniPELT* [157]: integrates LoRA, prefix-tuning, and adapters into each pre-trained model's block, using a gating mechanism to control the activation of each sub-module. *S4* [41] explores the design space of various PETL methods (Adapter, Prefix, BitFit, LoRA) and identifies optimal configurations for different layer groups.

To further optimize PETL configurations, researchers have turned to neural architecture search (NAS): *NOAH* [254] employs NAS to discover the most effective PETL configurations for different tasks, considering Adapter, LoRA, and Visual Prompt Tuning. *AUTOPEFT* [263] uses a high-dimensional multi-dimensional Bayesian optimization approach to search for optimal PETL configurations, including serial adapters, parallel adapters, and prefix tuning.

3.2 Parameter-efficient Transfer Learning of Audio Spectrogram Transformers [29]

3.2.1 Overview

Leveraging large pre-trained models for downstream tasks has become a cornerstone of several machine learning domains like natural language processing and audio/speech processing. The typical paradigm involves adapting the whole model to each downstream task [149, 222] (i.e., *full fine-tuning*). Despite achieving remarkable results, this approach leads to a specialized model for each task, which is unfeasible when fine-tuning a model on numerous downstream tasks.

To alleviate this issue, the research community is increasingly focusing on parameter-efficient transfer-learning (PETL) methods, whereby only a small amount of extra parameters is learned for each task while keeping the pre-trained model frozen [88, 133, 236]. In doing so, the risk of catastrophic forgetting the pre-trained model's knowledge is also highly reduced, a common problem in continual learning scenarios [27, 32]. For example, **prompt-tuning** methods insert trainable continuous vectors in the input or hidden state of the model, known as prompts [104, 126]. Alternatively, low-rank modules called **adapters**, which follow a bottleneck architecture with a very small intermediate dimension, are introduced into each layer. Another popular method, **LoRA** (Low-Rank Adaptation), leverages low-rank matrix decomposition of pre-trained weight matrices [101]. Several variants of LoRA have been recently proposed to enhance the original implementation leading to better performance and stability [142, 253].

Recently, PETL methods have garnered much attention in the audio and speech fields. For example, [47,130,138] provide extensive experiments on the use of PETL approaches and their combination for self-supervised learning speech models. Also for automatic speech recognition adapters have proven to be an effective solution [112, 207]. For audio classification, the Audio Spectrogram Transformer (AST) [74] obtains superb results, standing out as the state-of-the-art model for several downstream tasks. The problem of how to efficiently transfer the knowledge of the AST is of crucial importance, especially given the typical computational and storage constraints of audio devices. *Surprisingly, this topic has received minimal attention* [190]. Therefore, driven by 1) the absence of previous works, 2) the excellent results obtained by PETL methods in different domains for transformer models, and 3) the need to efficiently adapt the AST model to several downstream tasks, we ask the following question:

(Q) *Can we exploit state-of-the-art PETL methods for the efficient fine-tuning of AST to audio/speech downstream tasks?*

We methodically investigate this research question **(Q)**, and to do so we provide a framework whereby we can study the performance attained by several PETL methods on five audio/speech benchmarks. Furthermore, from our experiments, we notice that the Bottleneck adapter struggles to achieve on-par performance with respect to the full fine-tuning approach for speech tasks. We conjecture that this is attributable to the overly simplistic design of the Bottleneck adapter, where only linear layers are adopted, which hinders a complete learning of the task at hand. As a consequence, we propose a new adapter design that hinges upon the convolution module of the Conformer model. Our proposed *Conformer adapter* highly benefits from the introduction of the depthwise convolution layer, which allows not only to capture local spatial correlations but also trim down the number of parameters, thus bridging the gap with the full fine-tuning method.

We carry out extensive experiments leading to multiple findings:

- Among the standard PETL methods, *LoRA* and *Houlsby Bottleneck adapter* achieve the best performance overall, with *LoRA* using fewer parameters;
- Our proposed **Conformer adapter** provides considerable improvements over the Bottleneck adapter, surpassing or attaining performance parity with respect to the full fine-tuning approach while using only 0.29/0.59% parameters compared to it for the Pfeiffer/Houlsby configuration;
- We study the PETL methods under few-shot settings and their scalability with respect to the number of trainable parameters, validating the efficacy of our proposed adapter;
- We show empirically that the kernel size of the depthwise convolution is a key parameter to attain the best performance;
- We show that the Conformer adapter can be also harnessed for the efficient fine-tuning of another pre-trained model like Wav2Vec 2.0.

3.2.2 Methodology

AST Model Recap. The Audio Spectrogram Transformer (AST) is an attention-based model that achieves state-of-the-art results on various audio and speech tasks [74, 76]. The AST model receives as input audio spectrograms that are patchified and then a linear projection is applied to each patch. This results in a sequence of N tokens of size $d = 768$, which we refer to as $\mathbf{X}_{in} \in \mathbb{R}^{N \times d}$. AST comprises 12 attention layers, each of which is composed of two sub-layers: a *multi-head self-attention* (MHSA) sub-layer and a fully-connected feed-forward (FF) sub-layer. The output of each transformer layer, $\mathbf{X}_{out} \in \mathbb{R}^{N \times d}$ (we omit for simplicity the index of the layer), is computed as follows:

$$\mathbf{X}_{out} = \hat{\mathbf{X}} + \mathrm{FF}(\mathrm{LN}(\hat{\mathbf{X}})), \hat{\mathbf{X}} = \mathbf{X}_{in} + \mathrm{MHSA}(\mathrm{LN}(\mathbf{X}_{in})). \tag{3.1}$$

Both blocks, MHSA and FFN, include residual connections and layer normalizations (LN) [12], with the LN applied within the residual branch (i.e., Pre-LN).

Overview of Parameter-efficient Transfer Learning Methods. We now introduce the PETL techniques we used in our experiments: LoRA, prompt/prefix-tuning, and adapter-tuning.

LoRA [101] introduces trainable low-rank matrices into transformer layers to approximate the weight updates. For a pre-trained weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d_k}$, LoRA represents its update with a low-rank decomposition $\mathbf{W} + \Delta W = \mathbf{W} + \mathbf{AB}$, where $\mathbf{A} \in \mathbb{R}^{d \times r}$, $\mathbf{B} \in \mathbb{R}^{r \times d}$ are learnable and $r << d$. LoRA typically applies this update to the query and value projection matrices, $\mathbf{W}_q$ and $\mathbf{W}_v$, in the MHSA sub-layer. LoRA computes the query and value matrices like this:

$$\mathbf{Q}/\mathbf{V} = \mathbf{X}_{in}\mathbf{W}_{q/v} + s \cdot \mathbf{X}_{in}\mathbf{A}_{q/v}\mathbf{B}_{q/v}, \tag{3.2}$$

where s is a tunable scalar hyperparameter.

Prefix-tuning/Prompt-tuning [126, 128]. Prefix-tuning [128] inserts learnable continuous embeddings (i.e., *prompts*) to the keys and values of the MHSA block at every layer. Prompt-tuning [104, 126], instead, prepends the prompts in the input space after the projection layer. Following [104], we consider the "*shallow*" prompt-tuning version (SPT) where all the prompts are prepended to the first transformer layer, and the "*deep*" version (DPT) by prepending the prompts uniformly to each transformer layer.

Bottleneck Adapter [97, 174]. Adapters are light subnetworks that are inserted into every transformer layer. To keep the number of parameters limited, adapters exploit a *bottleneck* architecture. The input sequence of hidden dimension d is first down-projected (parametrized by $\mathbf{W}_{down}) \in \mathbb{R}^{d \times r}$ into a low-dimensional space with size r (the bottleneck dimension), followed by a non-linear activation function $f(\cdot)$ (e.g., ReLU), and then up-projected back to the original dimension d ($\mathbf{W}_{up} \in \mathbb{R}^{r \times d}$). We refer to this design as **Bottleneck** adapter and is the established choice in the NLP domain [88,97]. Adapter-tuning is a flexible approach in that we can identify multiple ways in which an adapter can be included in a transformer layer, resulting in different configurations. For example, the adapter can be inserted only after the FF block, (*Pfeiffer* [174]), or after both the MHSA and FF blocks (*Houlsby* [97]). Furthermore, the adapter can be included *sequentially*, either after the FF block [97] (sequential Pfeiffer) or after both FF and MHSA blocks [155] (sequential Houlsby), or *parallel* to only the FFN block [44, 88], or parallel to both FFN and MHSA blocks [107]. Mathematically, if we consider, as an example, the *Pfeiffer* configuration in which the Bottleneck adapter is placed *sequentially* after the FF block and we let $\mathbf{X}_{\mathrm{FF}} = \mathrm{FF}(\mathrm{LN}(\hat{\mathbf{X}}))$, following the notation in Eq. 3.1, the output is:

$$\mathbf{X}_{out} = \hat{\mathbf{X}} + \mathbf{X}_{\mathrm{FF}} + f(\hat{\mathbf{X}}\mathbf{W}_{down})\mathbf{W}_{up}. \tag{3.3}$$

Conformer Adapter (Ours). As we will show in Section 3.2.4, the Bottleneck adapter attains competitive results for audio classification tasks, whereas for speech tasks the gap with the full fine-tuning approach is sizeable. We speculate that this happens because the linear design of the Bottleneck adapter is not sufficient to disentangle the task at hand. For this reason, we propose to leverage the key block of the Conformer [80], a bleeding-edge model for several speech processing tasks: the *convolution module*. This module highly relies on the depthwise convolution, which is appealing for our PETL setting for two main reasons: **1)** it is used for capturing spatial correlations, a crucial aspect for speech downstream tasks, which the Bottleneck adapter fails to accomplish, and **2)** compared to a standard convolution, it requires fewer parameters, thus making it suitable for parameter-efficient

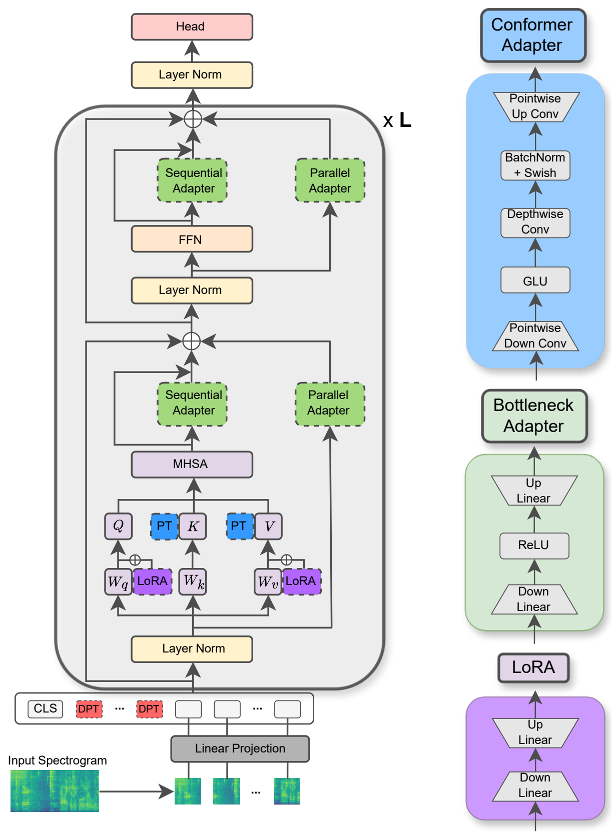


Figure 3.1: **Left**: illustration of the AST model and the integration of PETL methods into it. We use blocks with dashed outlines to characterize the added modules by those methods. **Right**: the inner structure of LoRA, Bottleneck adapter and our proposed Conformer adapter.

adapters. Therefore, we propose a new adapter that uses the convolution module as the building block and we call it *Conformer adapter* (see Fig. 3.1, right). Specifically, the first pointwise convolution down-projects the input sequence to a dimension of $2r$. Then, the Gated Linear Unit (GLU) halves the hidden dimension to the bottleneck one, r. At this point, the intermediate sequence undergoes the depthwise convolution layer with kernel size equal to k (refer to Section 3.2.5 for the analysis on this hyper-parameter), as well as the Batch Normalization and Swish activation. Finally, the dimension of the sequence is up-projected to the original d through a pointwise convolution. We show the effectiveness of our Conformer adapter in Section 3.2.4.

3.2.3 Implementation Details

All our experiments can be reproduced following the instructions at the official github repository here.

Datasets. We evaluate the PETL methods on four audio/speech downstream classification tasks. (1) **Audio classification**: we use the ESC-50 and UrbanSound8K (US8K) datasets. ESC-50 (ESC) [176] consists of $2,000$ 5-second-long environmental audio recordings of 50 classes. US8K [187] includes $8,732$ labeled sound excerpts of urban sounds from 10 classes. (2) **Keyword spotting**: Speech Commands V2 (GSC) [226] has $105,829$ 1-sec recordings of 35 speech commands. (3) **Intent classification**: Fluent Speech Commands (FSC) [147] includes $30,043$ English utterances spanning 31 classes. (4) **Emotion Recognition**: IEMOCAP (IEM) [25] comprises $10,039$ utterances from 10 distinct speakers with 4 emotional classes: *neutral*, *happy*, *sad*, *angry*.

Baselines. We include the *full fine-tuning* method (FFT), which fine-tunes the full pre-trained AST model; and *linear probing*, which only fine-tunes the classification head. We then study various PETL methods: shallow prompt-tuning (SPT), deep prompt-tuning (DPT), prefix-tuning (Pref-T), and BitFit [250], which is a common baseline that fine-tunes only the bias terms of the pre-trained backbone. SPT adds all the 300 prompts to the input of the first transformer layer, whereas DPT adds 25 prompts to each transformer layer. Pref-T adds 24 tokens to each layer. We then include LoRA and Bottleneck and Conformer (ours) adapters. The dimension of the intermediate space for adapters and LoRA is $r = d/\text{RR}$, where $d = 768$ is the hidden dimension of the AST model and RR is the reduction rate. Unless otherwise stated, r is set to 12, 8, and 6 for Bottleneck adapter, Conformer, and LoRA, respectively. In this way, the resulting number of parameters is roughly the same. For LoRA, following [101], the scaling factor is set to $s = \alpha/\text{RR}$, where $\alpha = 16$ leads to the best results (i.e., $s = 8$). We also note that each adapter module is added in parallel to only the MHSA layer (Pfeiffer) or both the MHSA and FF layers (Houslby). For this reason, Houlsby adapters require twice as many parameters as Pfeiffer. Inserting the adapters sequentially leads to slightly worse results, so we do not include these results. Finally, for the speech tasks we set the kernel size of the depthwise convolution layer to 31, which is the original value proposed in [80], while for audio tasks we found that $k = 8$ gives the best results (we refer the reader to Section 3.2.5 for a detailed analysis).

Training Details. For all experiments we use the AST model pre-trained on ImageNet-21K [55] and AudioSet [71] provided by the Huggingface Transformers library. The model has around 85.5 million parameters, 12 layers, and the hidden size is 768. For the ablation studies, we also use Wav2Vec 2.0, a well-established pre-trained model for speech tasks [14]. It has around 94M parameters and the same number of layers and hidden size as AST. For all datasets, we use AdamW optimizer [146] with cosine annealing scheduler and weight decay set to 0.1. The initial learning rate is 0.005 for adapters and LoRA, while for the three prompt-tuning methods is 0.01. Except for US8K that does not provide a validation set by default, for the others we set the hyper-parameters using the validation set. For the ESC and US8K datasets, we run 5-fold and 10-fold cross-validation as suggested in the original papers.

3.2.4 Main Results

Table 3.1 presents the performance comparisons among the various PETL methods. The following observations can be drawn: ❶ our Conformer adapter attains the best performance on average, bringing remarkable improvements over the Bottleneck adapter, with the best configurations leading to up to {4.9%, 22.4%} extra performance improvement on {GSC, FSC}, the two datasets that exhibit the biggest mismatch between the downstream tasks and the data used for pre-training the AST model. Furthermore, our adapter approaches the FFT baseline for GSC, whereas for FSC it is capable of exceeding it by more than 3 points. **Yet, our Conformer Pfeiffer/Houlsby adapter only uses 0.29/0.59% parameters compared to the FFT baseline**. ❷ If we focus on the audio classification tasks, we note good improvements with respect to US8K (it also manages to outstrip FFT by 0.26 points), while for ESC-50 our adapter performs on par with the Bottleneck

Table 3.1: Performance evaluations of the PETL methods on 4 datasets for AST. Best and second-best performances for each dataset are coloured in green and red, respectively.

Method	**Par**	**ESC**	**US8K**	**GSC**	**FSC**	**Avg**
FFT	85M	87.48	84.31	97.31	93.29	90.07
Linear	9/40K	75.85	77.93	41.78	27.52	55.77
BitFit	102K	86.05	82.17	85.51	63.85	79.40
SPT-300	230K	84.30	79.73	75.28	40.85	70.04
DPT-25	230K	86.52	83.67	89.18	68.60	81.99
Pref-T 24	221K	82.93	81.39	83.46	55.75	75.88
LoRA	221K	86.45	83.83	93.61	76.00	84.97
Bottleneck Adapter						
Pfeiffer	249K	**88.38**	83.44	91.33	73.19	84.09
Houlsby	498K	88.00	82.80	91.75	78.71	85.32
Conformer Adapter						
Pfeiffer	271K	88.30	**84.57**	96.28	95.48	**91.16**
Houlsby	542K	85.97	83.59	96.16	**96.34**	90.51

adapter. We point out that the Bottleneck adapter outperforms the FFT baseline and it can be already considered a strong approach, so using a more complex design like ours does not improve the performance accuracy. ❸ For the Bottleneck and Conformer adapters, the Houlsby configuration leads to better results for speech classification tasks, where having more parameters is beneficial (Houlsby configuration uses twice as many parameters as Pfeiffer), while for audio tasks Pfeiffer achieves better performance accuracy. ❹ Among the other PETL methods, we point out that LoRA achieves good results on average, beating the Bottleneck Pfeiffer adapter on 3 out of 4 benchmarks.

3.2.5 Ablation Studies

In this section, we study the efficacy of our proposed adapter under different settings such as few-shot learning and different pre-trained models (e.g., Wav2Vec 2.0). For the Bottleneck/Conformer adapters, we use the Pfeiffer configuration.

Table 3.2: Few-shot analysis for the ESC-50 and GSC datasets.

	ESC				**GSC**			
	Examples per class							
Method	1	2	4	8	2	8	32	64
DPT-25	32.7	44.3	57.0	71.9	**9.4**	**18.7**	43.1	57.1
LoRA	31.8	42.2	58.8	70.7	6.8	15.2	41.8	59.8
Bottleneck	**33.0**	**45.5**	**60.2**	**72.8**	7.2	16.0	47.9	66.6
Conformer	30.7	41.0	56.2	71.1	5.9	15.5	**58.7**	**77.5**

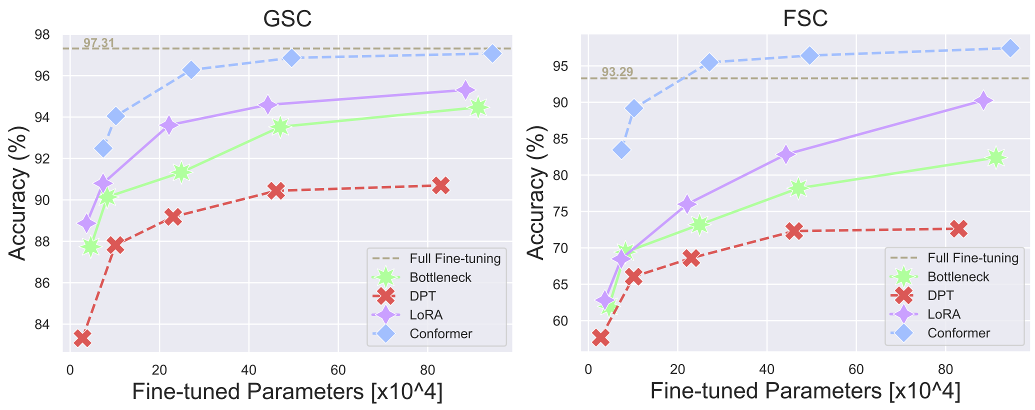


Figure 3.2: Scaling trend as more trainable parameters for each PETL method are used for GSC **(Left)** and FSC **(Right)** datasets.

Few-shot Analysis. We evaluate our proposed adapters for few-shot parameter-efficient transfer learning. This scenario is challenging because, in addition to the constraint on the number of trainable parameters, only a few samples are labeled per class. We report the accuracy results for ESC and GSC datasets in Table 3.2. We see that, whereas for ESC the Bottleneck adapter attains the best results, for GSC the gap between this and the Conformer adapter is more than 10 points. This again confirms that our proposed adapter is the best choice for speech tasks.

Scaling Abilities. We now want to verify whether our proposed adapter also performs better when fewer parameters (e.g., 50K) or more parameters (up to 1M) are allocated. We restrict our analysis to the Pfeiffer configuration and GSC/FSC datasets. In Fig. 3.2 we observe that the Conformer

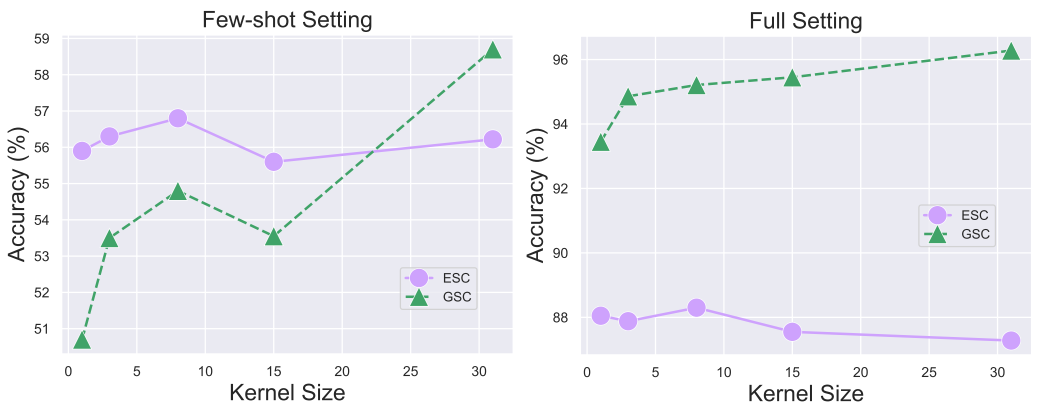


Figure 3.3: Impact of the kernel size for the ESC and GSC datasets on the few-shot **(Left)** and full **(Right)** settings.

adapter, regardless of the number of parameters, outstrips the other PETL approaches. In turn, LoRA turns out to be the second best method, and it exhibits strong scaling properties when more parameters are used, bringing better results than Bottleneck adapter and DPT. We point out that for FSC, the best result obtained with LoRA requires roughly 900K parameters, whereas the Conformer adapter only requires around 100K to achieve the same accuracy.

On the Kernel Size of the Conformer Adapter. We study the impact of the kernel size k of the Conformer adapter on the ESC and GSC datasets both for the few-shot (4/32 samples) and full (i.e., no few-shot) settings. We let k vary from 1 to 31, and we point out that setting $k = 31$ only adds roughly 2.8K parameters with respect to $k = 1$. In Fig. 3.3 we observe that for ESC, the results are not much influenced by k, and setting $k = 8$ provides the best results. On the contrary, for a more challenging dataset like GSC we see that increasing k leads to better results for both the few-shot and full settings, with the former being more sensitive to it. We speculate that this happens because a larger kernel allows learning more global features, and this is more beneficial when the number of features is small (for the Conformer adapter $r = 8$). On the contrary, since ESC is an easier dataset, a smaller kernel is sufficient to achieve optimal results.

Table 3.3: Results of Bottlenck and Conformer adapters for Wav2Vec 2.0 on GSC, FSC and IEMOCAP (IEM) datasets.

Method	**Par**	**GSC**	**FSC**	**Par**	**IEM**	**Avg**
FFT	90M	98.16	99.58	90M	70.05	89.26
Linear	24K	84.93	30.95	4K	36.82	50.90
Bottle	250K	94.96	96.41	895K	48.32	79.90
Conf	272K	**95.24**	**98.33**	927K	**55.81**	**83.13**

Additional Results for Wav2Vec 2.0. Finally, we want to verify whether the proposed Conformer adapter can be efficiently harnessed for another pre-trained model. In this direction, we consider Wav2Vec 2.0 [14]. We test the Bottleneck and Conformer adapters on FSC and GSC, as well as IEMOCAP [25], a benchmark for emotion recognition. For IEMOCAP, we increase the number of parameters to roughly 900K as it is a more challenging dataset. As we can see from Table 3.3, the performance gap between the adapter approaches and the FFT is large for IEMOCAP. Nonetheless, we can observe that the Conformer adapter reaches accuracy results of 55.81, with an improvement of more than 6 points over the Bottleneck adapter. For the other two datasets, the Conformer adapter turns out again to surpass the Bottleneck adapter.

3.2.6 Final Remarks

In this work, we studied the problem of parameter-efficient transfer learning for the AST model. To do so, we establish a framework that allows us to examine the performance achieved by the most common PETL methods across five audio/speech benchmarks. We also propose a new adapter module that relies on the Conformer convolution module, making effective use of the depthwise convolution. We show that our proposed adapter turns out to be competitive with the full fine-tuning approach and outperforms the established Bottleneck adapter, as well as LoRA and prompt-tuning methods. The Conformer adapter also provides strong results under few-shot settings, when we vary the number of parameters and if applied to another pre-trained model like Wav2Vec 2.0. Finally, we study the role of kernel size, underscoring its pivotal role in achieving peak performance.

3.3 Efficient Fine-tuning of Audio Spectrogram Transformers via Soft Mixture of Adapters [28]

3.3.1 Overview

In the previous section, we have studied multiple PETL techniques for adapting the AST and Wav2Vec 2.0 pre-trained models to multiple downstream tasks. From Table 3.1 we saw that the bottleneck adapter struggles to achieve performance parity with respect to the full fine-tuning approach for the speech downstream tasks. For this reason, in this section we propose the use of the mixture of experts (MoE) paradigm to enhance the performance.

Very recently, Mixture of Experts (MoE) models have shown remarkable results in natural language processing, pushing large language models to the limit, facilitating the effective scaling of Transformers and State Space Models while concurrently reducing computational costs [51, 106, 177, 249]. The MoE paradigm relies on the idea that sub-modular components, the *experts*, can specialize in different inputs and scale the model's capacity. While most works have focused on the use of MoE during the pre-training stage, only few works have leveraged MoE for efficient fine-tuning [161, 221, 249]. In the latter case, each expert is usually represented by a single adapter, and the model is referred to as Mixture of Adapters (MoA). However, these works usually target language-based tasks, whereas pure audio/speech classification tasks have not been taken into account before. Therefore, in this paper, we investigate the use of MoA for the Audio Spectrogram Transformer (AST), a powerful foundation model achieving state-of-the-art results on various audio/speech tasks [74], and we ask the following question:

(Q) *Can we leverage MoAs for the efficient fine-tuning of AST to audio/speech downstream tasks?*

To answer the above research question **(Q)**, we study the MoA's adoption for PETL of AST on four popular audio and speech benchmarks. Specifically, we propose to adapt a recent *sparse* version of MoE called *Soft-MoE* [178] to our PETL setting, whereby each expert only handles a small number of slots that are the result of a weighted combination of all input tokens. We call it **Soft-MoA**, and we compare it with the standard single adapter approach and with the dense version of MoA that requires each adapter to process all the input tokens (we refer to it as Dense-MoA). *By doing this, we are able to scale the number of adapters while keeping the computational cost limited as well as updating only a small fraction of parameters, thus leveraging the strengths of both the MoE and PETL paradigms.* We empirically show that both Soft and Dense MoA outperform the single adapter approach, both for the Pfeiffer and Houlsby configuration, leading to accuracy improvement of up to 2.5%; also, Soft-MoA attains performance parity with Dense-MoA while drastically trimming down the training cost. Finally, we further demonstrate the effectiveness of Soft-MoA by carrying out extensive ablation experiments revealing that **❶)** both Soft and Dense-MoA gains over the single adapter strategy are more evident when fewer parameters are available, **❷)** Soft-MoA is robust to "*expert imbalance*", thus ensuring that all experts are involved in the learning process, and **❸)** Soft-MoA attains the best performance accuracy when few slots (1/2) and several experts are used rather than the opposite case as multiple slots tend to learn redundant information.

3.3.2 Methodology

In this sub-section, we first give a brief recap of the AST model and the standard single adapter approach. Then, we present the details of the Dense-MoA and Soft-MoA approaches.

AST Model Recap. The Audio Spectrogram Transformer (AST) is an attention-based model that achieves state-of-the-art results on various audio and speech tasks [74, 76]. The AST model receives as input audio spectrograms that are patchified and then a linear projection is applied to each patch. This results in a sequence of L tokens of size $d = 768$, which we refer to as $\mathbf{X} \in \mathbb{R}^{L \times d}$. AST comprises 12 attention layers, each of which is composed of two sub-layers: a multi-head self-attention (MHSA) and a fully-connected feed-forward (FFN) module.

3.3.3 Adapters

Adapters are light subnetworks that are inserted into every layer of the AST model. To keep the parameters limited, adapters exploit a bottleneck architecture. The input sequence of hidden dimension d is first down-projected into a low-dimensional space with size r (the bottleneck dimension), and then up-projected back to the original dimension d. A non-linear activation function is also applied in-between the two fully-connected layers.

While the bottleneck adapter is the most common design, recent works have also explored convolution-based adapters mainly for vision tasks (e.g., Convpass) [40, 107]. In addition to this, adapters usually follow a Pfeiffer [174] or Houlsby [97] configuration: the former places the adapter parallel or sequentially to the MHSA or FFN sub-layer, whereas the latter includes the adapter on both sub-layers.

Dense MoA. It encompasses a set of N *"expert" adapters* $E_1, \dots E_N$ and a *router network* R that learns the optimal distribution over the adapters for a given input sequence. In its simplest form [67, 249], the router is a dense fully-connected layer with weights $\mathbf{W} \in \mathbb{R}^{d \times N}$ followed by a *softmax* function that takes as input the sequence $\mathbf{X}$ and merges the output of each adapter using the gating scores $g_1, \dots g_N$ to yield the output sequence $\mathbf{Y}$:

$$g_i = R(\mathbf{X})_i = \mathrm{softmax}(\mathbf{XW}), \tag{3.4}$$

$$\mathbf{Y} = \sum_{i=1}^{N} g_i \cdot E_i(\mathbf{X}). \tag{3.5}$$

If all the N adapters take part in the computation of the output of a given input (scaled by the router's distribution), then we refer to this as *Dense*-MoA (alternatively we can think of this as *ensemble* MoA). Whereas this approach would cater to exact computation of gradients and end-to-end-learning, it would also incur a substantial increase in computational costs since each input token is computed by every expert rather than a single expert. To circumvent the above issue, we propose to adapt a recent method called *Soft Mixture of Experts* [178] to our PETL setting where each expert is an adapter, and we call it *Soft-MoA*. Note that in our setting only the adapters are actually learned whilst the backbone model is frozen.

Soft-MoA. Rather than feeding all input tokens to each expert, Soft-MoA passes a different weighted soft combinations of all input tokens to each expert. Unlike other sparse techniques like Top-k [194] whereby only the k experts that are assigned the highest router's probability are activated, Soft-MoA provides fully-differentiable operations, better training stability, and immunity to "token dropping" and "expert imbalance" issues [178]. In practice, each adapter processes p slots, and each slot has a corresponding d-dimensional vector of parameters. These parameters are denoted by $\mathbf{\Phi} \in \mathbb{R}^{d \times (N \cdot p)}$. The input slots, $\tilde{\mathbf{X}}$, are computed as the convex combination of all the L input tokens:

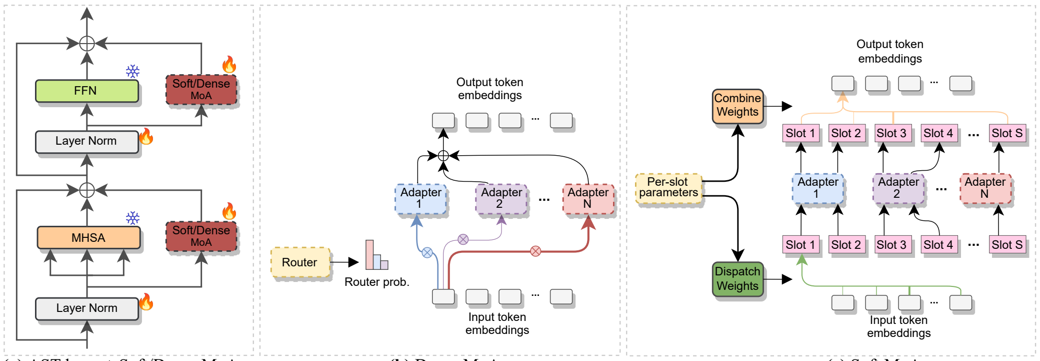
$$\tilde{\mathbf{X}} = \mathbf{D}^\top \mathbf{X}, \quad \mathbf{D}_{i,j} = \frac{\exp((\mathbf{X\Phi})_{i,j}}{\sum_{h=1}^{L} \exp((\mathbf{X\Phi})_{h,j})}. \tag{3.6}$$

$\mathbf{D}$ is called the *dispatch weights* and corresponds to applying a softmax along the columns of $\mathbf{X\Phi}$. At this point, each adapter processes the corresponding slots: $\tilde{\mathbf{Y}}_i = E_{\lfloor i/p \rfloor}(\tilde{\mathbf{X}}_\mathbf{i})$. Finally, the output tokens $\mathbf{Y}$ are the result of a convex combination of all $(N \cdot p)$ slots:

$$\mathbf{Y} = \mathbf{C}\tilde{\mathbf{Y}}, \quad \mathbf{C}_{i,j} = \frac{\exp((\mathbf{X\Phi})_{i,j}}{\sum_{h=1}^{N \cdot p} \exp((\mathbf{X\Phi})_{i,h})}. \tag{3.7}$$

The matrix $\mathbf{C}$ is referred to as the *combine weights*, and is equivalent to applying a softmax over the rows of $\mathbf{X\Phi}$.

We provide an overview of Soft and Dense MoA in Figure 3.4. Finally, for our experiments, following [29] that show that inserting the adapter in



(a) AST layer + Soft/Dense MoA. **(b)** Dense MoA. **(c)** Soft MoA.

Figure 3.4: **(a)** For each AST layer, the Soft/Dense MoA blocks are inserted parallel to MHSA (Pfeiffer) or parallel to both MHSA and FFN sub-layers (Houlsby). **(b)** Illustration of Dense-MoA, whereby each expert contribution, scaled by the router's distribution (thickness of the arrows), is summed to produce the final output. **(c)** In Soft-MoA, each expert only processes a subset of slots (here 2), and each slot accepts as input a weighted combination of all input tokens (thickness of the arrows). Note that the trainable parameters are represented by dashed blocks. Best viewed in color.

parallel achieves better performance than sequentially, we place the MoA block *parallel* to the MHSA layer only (i.e., Pfeiffer) or *parallel* to both the MHSA and FFN layers (i.e., Houlsby). The number of slots p is an hyperparameter, and we elaborate on its optimal value on Section 3.3.6.

3.3.4 Implementation Details

For our experiments, we mainly follow the implementation details of [29] (i.e., Section 3.2) to provide a fair comparison. All our experiments can be reproduced following the instructions at the official github repository here.

Datasets. We evaluate the PETL methods on three audio/speech downstream classification tasks. (1) **Audio classification**: we use the ESC-50 and UrbanSound8K (US8K) datasets. ESC-50 [176] consists of 2,000 5-second-long environmental audio recordings of 50 classes. US8K [187] includes 8,732 labeled sound excerpts of urban sounds from 10 classes. (2) **Keyword spotting**: Speech Commands V2 [226] has 105,829 1-second recordings of 35 speech commands. (3) **Intent classification**: Fluent Speech

Commands (FSC) [147] includes 30,043 English utterances spanning 31 classes.

PETL baselines. We include two traditional fine-tuning strategies: **full fine-tuning** (Full-FT), which finetunes the full pre-trained AST model; and **linear probing**, which only fine-tunes the classification head. Following [29], we include some common PETL baselines: **BitFit** [250], **deep prompt-tuning** (DPT) [104], **prefix-tuning** (Pref-T) [128] and **LoRA** [101]. For the analysis of MoA, we take into account both *Bottleneck* [97] and *Convpass* [107] adapters. We report **Dense** and **Soft-MoA** (D/S-MoA) with 14 or 7 adapters for the *Pfeiffer* and *Houlsby* configuration, respectively, and we compare them with the standard implementation using a single adapter per layer (**Single**).

Training Details. For all experiments we use the AST model pre-trained on ImageNet-21K [55] and AudioSet [71] provided by the Huggingface Transformers library [229]. The model has around 85.5 million parameters, and the hidden size is 768. For the LoRA, DPT, etc. baselines, we use the same training parameters described in Section 3.2. For MoA experiments, we use AdamW optimizer with cosine annealing scheduler and weight decay set to 0.1. For the ESC-50 and US8K datasets, we run 5-fold and 10-fold cross validation as suggested in the original papers. Except US8K that does not provide a validation set by default, for the others we set the hyper-parameters using the validation set.

3.3.5 Main Results

Table 3.4 presents the performance comparisons between the single adapter approach and Soft/Dense MoA, as well as some other common PETL methods. The single adapter approach has bottleneck dimension equal to 24, whereas Soft/Dense-MoA include 14 adapters, each with bottleneck dimension 1, and one slot is used for each adapter. From table 3.4 we observe that both MoAs outperform the single adapter, leading to up to 2.5 % performance improvement on average for the Bottleneck case, while for Convpass we notice that Soft-MoA is slightly worse than Dense-MoA, but still better than the single adapter. In general, the biggest gain is obtained with the FSC

Table 3.4: Performance evaluations of Dense and Soft-MoA on 4 benchmarks for the Pfeiffer configuration. We report the top-1 accuracy for each dataset, the average over the four datasets (**Avg**), and the average train step time in milliseconds (**Time**).

Method	**# par**	**ESC-50**	**US8K**	**GSC**	**FSC**	**Avg**	**Time(ms)**
Full FT	85.5M	87.48	84.31	97.31	93.29	90.07	645
Linear	9-40K	75.85	77.93	41.78	27.52	55.77	226
BitFit	102K	86.05	82.17	85.51	63.85	79.40	513
DPT	230K	86.52	83.67	89.18	68.60	81.99	561
Pref-T	221K	82.93	81.39	83.46	55.75	75.88	529
LoRA	221K	86.45	83.83	93.61	76.00	84.97	525
Bottleneck Adapter							
Single	470K	88.65	83.36	93.53	78.19	85.93	513
D-MoA 14	535K	89.55	84.30	93.89	82.43	**87.54**	**1689**
S-MoA 14	535K	89.08	84.88	93.91	82.48	**87.59**	626
Convpass Adapter							
Single	491K	87.93	83.38	93.47	77.62	85.60	515
D-MoA 14	535K	89.30	84.32	93.70	83.52	**87.71**	**1727**
S-MoA 14	535K	88.43	84.29	93.36	80.36	**86.61**	638

dataset (up to 5.5 and 7.6 %). Indeed, FSC is the more challenging dataset as it includes longer speech audio data, thus we argue that multiple adapters can specialize in learning different information, and consequently leading to better performance. We also notice that the GSC dataset does not benefit much from the use of MoA architecture. We surmise that a single adapter already achieves very competitive performance and so the use of multiple smaller adapters is not helpful. Another pivotal aspect is the extra computational cost brought by MoAs, estimated as the average train step time in milliseconds. Whereas Dense-MoA incurs a considerable increase in time (more than 3x with respect to the single adapter), S-MoA, instead, requires only a limited extra time, while guaranteeing on-par performance.

Finally, we test Soft-MoA's efficacy for the Houlsby configuration, where the MoA block is also inserted parallel to the FFN sub-layer. Table 3.5 confirms the superiority of both MoAs over the single adapter.

Table 3.5: Results of D/S-MoA for the Houlsby configuration. The number of parameters coincides with Pfeiffer as we still use 14 adapters split equally between MHSA and FFN layers.

Method	ESC-50	US8K	GSC	FSC	Avg
Bottleneck Adapter					
Single	88.00	82.80	91.75	78.71	85.32
D-MoA 7	87.33	83.78	94.11	82.64	**86.97**
S-MoA 7	87.13	83.77	93.67	81.41	**86.50**
Convpass Adapter					
Single	87.15	82.75	92.55	77.79	85.06
D-MoA 7	87.31	83.77	93.20	82.26	**86.63**
S-MoA 7	88.13	83.87	92.69	81.69	**86.60**

3.3.6 Ablation Studies

We now conduct some ablation studies to evaluate the effectiveness of Soft-MoA under different settings. We focus on the Pfeiffer Bottleneck configuration, and on the FSC dataset.

Increasing the Parameters Budget. We examine the methods' behaviour as we increase the number of trainable parameters. For the single adapter, we increase the parameters by making the bottleneck dimension r larger, while for MoAs we keep it to 1 and we increase the number of adapters. From Figure 3.5 **(Left)** we observe that Soft-MoA outperforms the single adapter, although when more and more parameters are available the two methods tend to achieve similar results, thus showing that using a single adapter is a good alternative when scaling the number of parameters is sustainable.

Few-big vs Many-small Adapters. We now investigate how the MoA methods scale with respect to the number of adapters N. Regardless of N, we fix the number of learnable parameters to around 900K to have a fair comparison. In this way, we want to figure out if having more adapters with a smaller bottleneck dimension is better than having a few but "bigger" (in terms of parameters) adapters. The Figure 3.5 **(Middle)** shows that Dense-MoA, due to its intrinsic dense structure, reaches the peak performance when

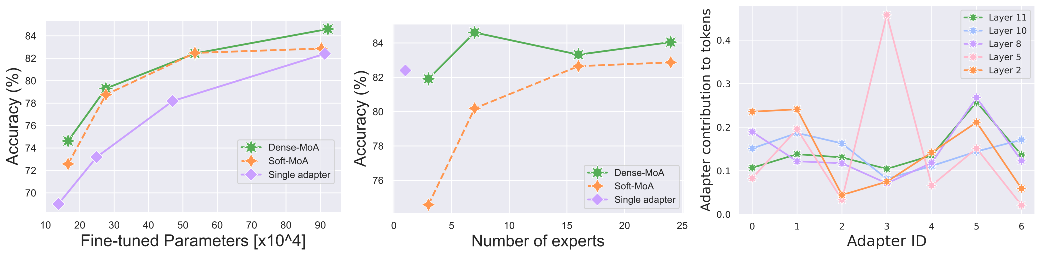


Figure 3.5: **(Left)**. The accuracy trend as more parameters are used. **(Middle)**. The effect of the number of adapters given a fixed parameters budget. **(Right)**. Adapters contribution to the output tokens for various layers. Results reported for FSC.

$N = 7$, and then adding more adapters does not lead to additional improvement. On the contrary, Soft-MoA depends heavily on N, and only when this number is large enough does it attain good performance. This trend is in line with that of the original Soft MoE paper [178].

Adapters Contribution to the Output Tokens and Classes. By design, the computation of the final output tokens depends on a linear combination of all the adapters' slots. We want to verify whether all adapters contribute to the output sequence. We fix one slot per adapter and consider 7 adapters, and we approximate the contribution of each adapter by averaging their coefficients in the linear combinations for all output tokens. We average over all the batches of the test set and report the adapter contribution for different layers in Figure 3.5 **(Right)**. We see that some adapters have a bigger impact than others, but all of them contribute to the final output tokens. Therefore, Soft-MoA does not suffer from the expert imbalance issue, namely few adapters monopolize the output contribution while the others are overshadowed, an issue that affects other routing strategies like Top-k [178, 194]. In addition to this, *we compute the contribution of each adapter to each class*. To do this, for each sample of each class, we compute the contribution of each adapter and then we average over the total number of samples per class (for this reason the sum of each row of the heatmap does not sum to 1). We observe from Figure 3.6 that some adapters specialize more for some classes than others (adapter 0 has a high contribution for classes 26-29, adapter 1 for classes 7, 20-22). We also see that the adapter with ID 5 has a strong

N/p	Acc
2/14	78.52
4/6	80.26
6/4	81.65
8/3	82.36
12/2	**83.24**
24/1	**82.87**

Table 3.6: Optimal trade-off between the number of adapters N and slots p.

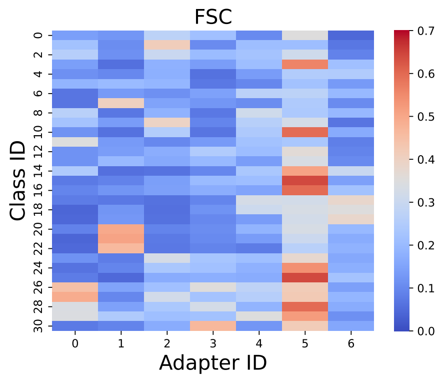


Figure 3.6: Distribution of expert activation frequencies per class.

contribution for several classes.

Optimal Trade-off between Slots and Adapters. The number of slots p is an important hyper-parameter of Soft-MoA, thus we examine its optimal value. We notice that if we set the number of slots equal to the number of tokens L, Soft-MoA boils down to Dense-MoA, so it is crucial to keep p small. For our experiments, depending on the dataset, L is between 100 and 500, and setting p up to 14 is a reasonable choice. We report the results for FSC in Table 3.6 and we see that, with the same number of trainable parameters, having more adapters with few slots brings better results than having few adapters but many slots. We speculate that this happens because multiple slots corresponding to the same adapter might have a tendency to learn similar concepts and become redundant, whereas using more adapters ends up learning more diverse information.

3.3.7 Final Remarks

In this work, we proposed Soft Mixture of Adapters to efficiently fine-tune the AST model on various audio/speech downstream tasks. Soft-MoA relies on multiple adapters that take as input a soft convex combination of all the input tokens, thus reducing the computational cost of the dense counterpart. Extensive experiments on 4 benchmarks show that Soft-MoA performs on par with Dense-MoA, and it outperforms the single adapter strategy, confirming itself as a strong method also for parameter-efficient transfer learning settings. To strengthen our analysis, we carry out ablation studies revealing that Soft and Dense MoA provide bigger gains over the single adapter when the parameters budget is limited. We also show that Soft-MoA scales better with the number of adapters and that it is sufficient to use only 1 or 2 slots to achieve the optimal performance.

3.4 Summary of the Contributions

In this chapter, we provide a framework for investigating various parameter-efficient transfer learning methods aimed at efficiently fine-tuning foundation models for audio and speech. Besides, we propose two key advancements:

1. A novel adapter architecture inspired by the Conformer model, designed to enhance performance.

2. The application of a mixture-of-experts paradigm to enable specialization of different adapter modules based on the input data.

Chapter 4

`Llama-AVSR`: a New Multimodal Large Language Model with Strong Audio-Visual Speech Recognition Abilities

In Chapter 3, we have studied how to efficiently transfer the knowledge of pre-trained audio and speech foundation models to perform multiple downstream classification tasks. In this chapter, instead, we go one step further and we also investigate how to leverage the reasoning and understanding capabilities of pre-trained *large language models* (LLMs) for generative tasks like (automatic) audio speech recognition ASR, visual speech recognition VSR, and audio-visual speech recognition AVSR. This is motivated by the ever-growing abilities of LLMs to process multimodal inputs and the absence of an exhaustive research study of how LLMs can be used to carry out the ASR, VSR, and AVSR tasks.

Recent advancements in LLMs [1, 10, 64, 77, 105, 209, 260] have yielded impressive emergent abilities, including instruction tuning [168, 172, 227], In-Context Learning [23, 59], and Chain-of-Thought reasoning [220, 228]. While LLMs excel at understanding and generating text, their reliance on textual data limits their ability to perceive and reason about other modalities (e.g., vision and audio). In contrast, large vision [116, 251] and audio [14, 43, 98] models are adept at processing visual and audio information but often struggle with complex reasoning tasks that require understanding and manipulating abstract concepts.

Recognizing the complementary strengths of LLMs and large vision/audio models, researchers have begun to integrate these models to create a new class of models known as Multimodal Large Language Models (MLLMs) [15,56,73, 140,141,150,160,208,246]. Formally, MLLMs are LLM-based models that can process, reason with, and generate a wide range of multimodal information, including text, images, videos, and audio. A typical MLLM is composed of three modules: a pre-trained modality-specific encoder, a pre-trained LLM, and a modality-specific projector. This architecture draws inspiration from human cognition, where modality encoders (e.g., image or audio encoders) function like human eyes or ears, receiving and pre-processing sensory input. The LLM, akin to the human brain, processes and reasons with this pre-processed information. The modality projector acts as a bridge, aligning different modalities to enable seamless communication between the encoder and the LLM.

In line with these extraordinary results, in the audio and speech domain, it is been shown that an LLM can be equipped with speech recognition abilities by just concatenating the audio tokens, computed with an audio encoder, and the text tokens to achieve state-of-the-art results [39, 66, 248]. On the contrary, tasks like VSR and AVSR, which also exploit noise-invariant lip movement information, have received little or no attention. To bridge this gap, we propose Llama-AVSR, a new MLLM with strong audio-visual speech recognition capabilities (see Section 4.1). It leverages pre-trained audio and video encoders to produce modality-specific tokens which, together with the text tokens, are processed by a pre-trained LLM (e.g., Llama3.1-8B) to yield the resulting response in an auto-regressive fashion. Llama-AVSR requires a small number of trainable parameters as only modality-specific projectors and LoRA modules are trained whereas the multimodal encoders and LLM are kept frozen. We evaluate our proposed approach on LRS3, the largest public AVSR benchmark, and we achieve new state-of-the-art results for the tasks of ASR and AVSR with a WER of 0.81 our results, we investigate the key factors that underpin the effectiveness of Llama-AVSR: the choice of the pre-trained encoders and LLM, the efficient integration of LoRA modules, and the optimal performance-efficiency trade-off obtained via modality-aware compression rates.

4.1 Large Language Models are Strong Audio-Visual Speech Recognition Learners [31]

By integrating both auditory and visual data, audio-visual speech recognition (AVSR) aims to enhance the capabilities of speech recognition systems [3, 83, 151, 196]. Notably, the additional use of visual lip movement information is highly beneficial in environments characterized by background noise or ambient speech, enhancing noise robustness [75, 197]. Multiple works have shown that having access to a wide amount of labeled audio-visual data is the key to obtaining strong results [36, 193]. However, such a reliance on large-scale transcribed datasets of up to 100K samples is prohibitively expensive and time-consuming. Therefore, recent methods have focused on different paradigms. A vast class of works relies on the Self-Supervised Learning (SSL) paradigm [79] by pre-training on large-scale datasets of unlabeled videos and then fine-tuning on a few hundred hours of labeled videos [83,84,99,135,196]. For example, AV-HuBERT [196] learns to predict cluster assignments from masked audio-visual data. U-HuBERT [99] builds upon AV-HuBERT and exploits unlabeled unimodal and multimodal speech data during the pre-training phase. Other methods like RAVEn [83] and its extension, BRAVEn [84], use a student-teacher framework where the teacher model's weights are updated via an exponential moving average of the student's using lightweight Transformer predictors. Another line of research [151, 153, 245] proposes to use publicly-available pre-trained ASR models to automatically annotate large-scale audio-visual datasets.

Large language models (LLMs) have shown exceptional abilities in handling natural language tasks [64, 105]. Their impressive generalization and instruction-following capabilities have spurred the development of multimodal LLMs (MLLMs) [160, 208], thus making it possible to process other modalities rather than just text. MLLMs have obtained remarkable performance in several tasks like vision-language [6, 127, 140, 141], video understanding [48,129], and audio understanding [56,73,118] to name a few. Some works also propose unified MLLMs that handle multiple modalities at the same time [131, 150, 247].

Recent research has also highlighted the effectiveness of LLMs for the task of automatic speech recognition (ASR) [39,66,102,154,248] and visual speech recognition [244]. An LLM can be endowed with speech recognition abilities by conditioning on the audio embeddings and by training lightweight projector layers [154] or LoRA modules [66]. Given the excellent performance achieved by LLM-empowered ASR models and the ability of LLMs to process multiple modalities simultaneously [131,247], we aim to explore whether LLMs can be adapted to perform the task of AVSR. In this way, the LLM would rely on multimodal tokens that convey the same information but from complementary perspectives. For example, the LLM can highly benefit from additional video tokens in the presence of noisy acoustic environments. Furthermore, there is a notable absence of comprehensive research that systematically explores the integration of ASR, VSR, and AVSR tasks using LLMs. Driven by these considerations, we pose the following research question:

(Q) *How can we leverage powerful* ***LLMs*** *to carry out the tasks of audio, visual, and audio-visual speech recognition?*

To tackle **(Q)**, we propose a new framework, `Llama-AVSR`, which harnesses pre-trained LLMs (e.g., Llama-based [64, 209]) and audio/video encoders for the tasks of ASR, VSR, and AVSR. Our approach involves extracting modality-specific feature representations from pre-trained encoders. These features are then downsampled to reduce computational complexity and projected into the LLM's embedding space using lightweight projectors, resulting in audio/video tokens. By concatenating these tokens with the text tokens, we integrate information from all modalities. These combined tokens are fed into the LLM that generates the transcriptions in an auto-regressive way. `Llama-AVSR` only trains the projectors and the LLM's LoRA module while keeping the pre-trained encoders and LLM frozen. In this way, we significantly reduce the number of trainable parameters compared to traditional methods that train the entire pipeline. Moreover, the modularity of our `Llama-AVSR` framework facilitates the seamless integration of various pre-trained encoders and LLMs of different sizes. This flexibility allows us to easily adapt our model to meet specific requirements of the size-performance trade-off.

`Llama-AVSR` achieves new state-of-the-art results on the LRS3 dataset for the tasks of ASR (**0.81**%) and AVSR (**0.77**%) by training only 42 and 57 million parameters, respectively. For VSR, our method outperforms prior works using LLM [244] and attains comparable performance with state-of-the-art methods [83, 151]. Moreover, we experimentally analyze the **key factors** that lead to the effectiveness of `Llama-AVSR` for the three tasks, observing that: **❶)** the choice of the pre-trained audio and video encoders as well as of the LLM has a big impact on the final performance, **❷)** incorporating LoRA modules into the LLM and video encoder are highly beneficial to improve the overall performance whilst requiring limited additional parameters, **❸)** the selection of the compression rate is crucial to finding the optimal trade-off between performance and efficiency for all three tasks.

4.2 Method

Our method, `Llama-AVSR`, leverages the capabilities of pre-trained audio and video encoders as well as LLMs for carrying out the tasks of ASR, VSR, and AVSR. It comprises three main components: **1)** *modality-specific pre-trained encoders* (i.e., audio and video), **2)** *modality-specific projectors*, and **3)** an *LLM*. This architecture is referred to as **Multimodal LLM** (MLLM) as the LLM produces text responses in an auto-regressive way given a sequence of multimodal input tokens (audio/video + text). Our approach is illustrated in Figure 4.1. Due to its versatility and streamlining pipeline, we adopt the *decoder-only*-based approach [137, 141, 160], which combines pre-trained

LLMs and multimodal inputs through lightweight connectors, rather than the *cross-attention*-based approach [6, 64, 208], which incorporates multimodal tokens through a cross-modal attention mechanism. We delve into the details of each MLLM's building block below.

Modality-specific Pre-trained Encoders. We exploit pre-trained audio (e.g., Whisper [181]) and video (e.g., AV-HuBERT [196]) encoders to extract meaningful audio and video features to be harnessed by the LLM. These pre-trained encoders are maintained frozen throughout the training process. For the task of VSR only, we found empirically that adding a LoRA module to the video encoder brings additional improvements at a small overhead cost of around 6M parameters.

Modality-specific Projector. The projector, sometimes called connector [129, 131], bridges the pre-trained encoders and the LLM by translating audio and visual features into understandable tokens that the LLM can process. The quality of these tokens significantly influences the MLLM's performance. Furthermore, the projector plays a crucial role in terms of efficiency since it determines how many tokens will be processed by the LLM, which handles most of the computational load. For instance, 6 seconds of audio-visual features yield 450 frames, thus processing these long sequences with the LLM presents substantial computational challenges. Given their simplicity and popularity, we opt to employ linear projectors, which excel at capturing fine-grained details by preserving local audio and visual patterns without loss due to their inherent frame-wise transformation [34]. However, they usually struggle in terms of efficiency and scalability as the number of features (before the projector) and tokens (after) remains constant. We tackle this by downsampling the audio and video features to reduce their sequence length: we first concatenate K (we call it *compression rate*) consecutive features along the hidden dimension (i.e., the sequence length/hidden size is reduced/increased by a factor K), and then the projector maps the audio/video features into audio/video tokens through two linear layers to match the text tokens hidden size. Finally, audio and video tokens are concatenated with the textual tokens, ready to be processed by the LLM.

LLM. The goal of the LLM is to generate instruction-following responses

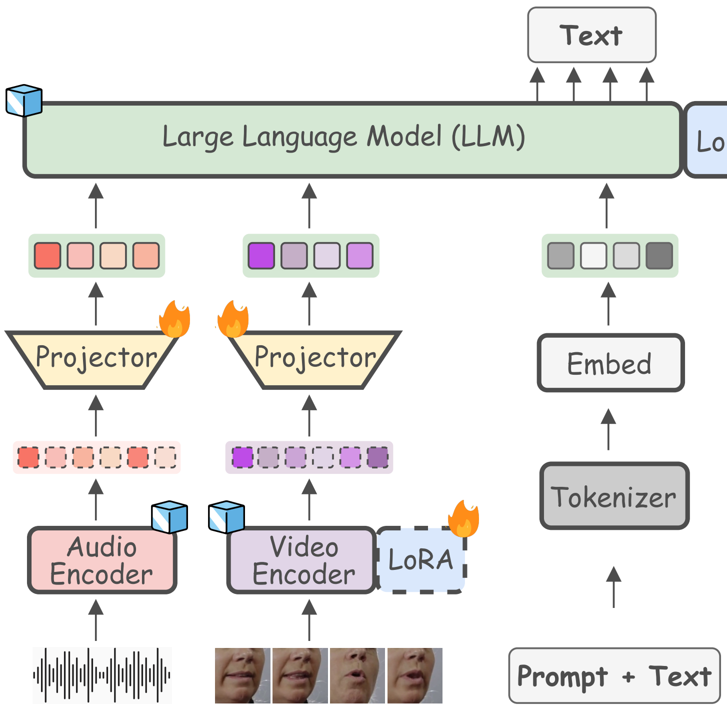


Figure 4.1: Illustration of `Llama-AVSR`. Audio and video features are extracted via pre-trained encoders and subsequently downsampled and projected into the LLM space through modality-specific projectors. The resulting audio and video tokens are concatenated with the textual ones and processed by the LLM. The video encoder is equipped with the LoRA module only for the VSR task (dashed outline). 🔥 and 🧊 means that the block is trained and kept frozen, respectively.

given a sequence of multimodal inputs. In addition to the text tokens, the LLM processes: 1) audio tokens for the ASR task, 2) video tokens for the task of VSR, and 3) both audio and video tokens for the task of AVSR. Therefore, the LLM digests audio and/or video and text (instruction/prompt + transcription) tokens while generating the text response (i.e., the transcription) in an auto-regressive fashion. Formally, if we consider the task of AVSR, the multimodal input comprises audio $\mathbf{X}_{\mathsf{aud}}$, video $\mathbf{X}_{\mathsf{vid}}$, and text $\mathbf{X}_{\mathsf{text}}$ tokens. The LLM predicts the response $\mathbf{Y} = \{y_i\}_{i=1}^{N}$ conditioned on the multimodal input tokens, where N represents the number of tokens. Accordingly, the probability of the target response $\mathbf{Y}$ is computed by:

$$p(\mathbf{Y}|\mathbf{X}_{\mathsf{aud}}, \mathbf{X}_{\mathsf{vid}}, \mathbf{X}_{\mathsf{text}}) = \prod_{i=1}^{N} p(y_i|\mathbf{X}_{\mathsf{aud}}, \mathbf{X}_{\mathsf{vid}}, \mathbf{X}_{\mathsf{text}}, y_{<i}), \tag{4.1}$$

where $y_{<i}$ is the generated output sequence up to token $i-1$. In all our experiments the LLM is kept frozen while we add a LoRA module to align the LLM responses with the multimodal inputs.

4.3 Implementation Details

Datasets. We conduct our experiments on LRS3 [4], the largest publicly available dataset for audio-visual speech recognition. It includes 433 hours of transcribed English video clips from TED talks. We also report the results when training on only the 30-hour "trainval" set of LRS3 (denoted "low-resource" setting in [83]). Additionally, following [151], we use the pre-trained Whisper model [181] to generate the transcriptions of the English-speaking videos from VoxCeleb2 [50], resulting in additional $1,326$ hours. We also report the results from this setting: LRS3 + VoxCeleb2 ($1,756$ hours). In this way, we test the effectiveness of our proposed method in 3 different settings based on the amount of labeled hours.

Pre-processing. We follow [151] for the pre-processing of the datasets. For the video part, we crop the mouth region of interests (ROIs) through a bounding box of 96×96. Each frame is normalised by subtracting the mean and dividing by the standard deviation of the training set. For audio, we only apply z-normalisation per utterance.

Training/Inference Details. We augment visual inputs through horizontal flipping, random cropping, and adaptive time masking, whereas for audio we only apply adaptive time masking. For training, similar to [151], we sample bubble noise from the NOISEX dataset [214] using a uniform distribution from the range [-5, 0, 5, 10, 15, 20, ∞]dB and add it to the clean speech signal. For our experiments, we use Whisper-medium [181] and AV-HuBERT Large [196] as pre-trained audio and video encoders. These encoders are kept frozen, and only for the VSR task do we add a LoRA module to the video encoder (we use a rank = 64, resulting in roughly 6M additional parameters). For AV-HuBERT, we use the checkpoint pre-trained on LRS3 + VoxCeleb2. The projectors consist of two linear layers with ReLU activation in between ($\sim$ 15M parameters). For the tasks of ASR and VSR, we apply a compression rate K of 3 for the settings with 433 and 1756 hours, and 2 for the low-resource setting with 30 hours. For the task of AVSR, we set K equal to 4 and 2 for the audio and video tokens, respectively. We refer the reader to Section 4.5 for a detailed analysis of K. For the main experiments, we use the pre-trained Llama3.1-8B [64] as our LLM, while for the ablation studies we also experiment with TinyLLama [252], Llama2-7B [209], and Llama2-13B [209] to compare LLMs of different sizes. The LLM is coupled with a LoRA module ($\sim$ 27M parameters). The prompt for the LLM is "`Transcribe {task_prompt} to text.`", where `task_prompt` $\in$ {"`speech`", "`video`", "`speech and video`"} depending on the task we study. We train our model for 10 epochs with the AdamW optimizer with cosine annealing scheduler and weight decay set to 0.1. The learning rate is set to 1e-3 for ASR and AVSR tasks, and 5e-4 for VSR. For decoding, we use beam search with a beam width of 15 and temperature of 0.6 and only the most probable tokens with probabilities that add up to top_p = 0.9 are kept for generation.

4.4 Main Results

We report the results in terms of Word Error Rate (WER) obtained by `Llama-AVSR` in Table 4.1. We compare our approach with multiple state-of-the-art works based on the considered task (ASR, VSR, AVSR) and the number of labeled hours. We also include the number of trainable parameters and the encoder(s) employed by each method, which may utilize a pre-trained model (as in our case, VSP-LLM [244] and Whisper-Flamingo [185]) or be trained from scratch [36, 151, 196] (i.e., Transformer or Conformer). For the task of ASR ("audio-only setting"), `Llama-AVSR` sets a new state-of-the-art with a WER of 0.81% by training on LRS3 + VoxCeleb2 (1756 hours), and it also outperforms the other methods when using 433 and 30 hours achieving

Table 4.1: WER (%) of `Llama-AVSR` and prior works on the LRS3 dataset. We report results based on the task (ASR, VSR, AVSR) and on the labeled hours (30, 433, 1756).

Method	**Encoder(s)**	**Trainable Par. (M)**	**Labeled Hours**	**WER ↓**
		Audio-Only Setting		
RAVEn [83]	Transformer	328	30/433	1.9/1.4
BRAVEn [84]	Transformer	328	30/433	1.7/1.1
CM-seq2seq [152]	Conformer	250	433	2.3
Fast Conformer [24]	Conformer	197	435	1.6
AV-HuBERT [196]	Transformer	325	433	1.3
Whisper-finetuned [185]	Whisper	1550	433	2.3
auto-avsr [151]	Conformer	243	1902/3448	1.0/1.0
`Llama-AVSR`	AV-HuBERT A	40	433	1.4
`Llama-AVSR`	Whisper	**42**	30	**1.5**
`Llama-AVSR`	Whisper	**42**	433	**1.1**
`Llama-AVSR`	Whisper	**42**	1756	**0.79**
		Video-Only Setting		
RAVEn [83]	Transformer	328	30/433	24.8/24.4
BRAVEn [84]	Transformer	328	30/433	20.0/20.1
AV-data2vec [135]	Transformer	325	30/433	30.8/28.5
auto-avsr [151]	Conformer	250	433	36.3
AV-HuBERT [196]	Transformer	325	433	26.9
VSP-LLM [244]	AV-HuBERT V	17	433	26.7
auto-avsr [151]	Conformer	250	1902	23.5
LP Conformer [36]	Conformer	570	100K	12.8
`Llama-AVSR`	AV-HuBERT V	**48**	30	**28.4**
`Llama-AVSR`	AV-HuBERT V	**48**	433	**25.3**
`Llama-AVSR`	AV-HuBERT V	**48**	1756	**24.0**
		Audio-Visual Setting		
CM-seq2seq [152]	Conformer	250	433	2.3
Whisper-Flamingo [185]	Whisper	631	433	1.5
CMA [114]	Transformer	500	433	1.5
auto-avsr [151]	Conformer	425	1902/3448	1.0/0.9
Fast Conformer [24]	Conformer	197	1687	0.9
ViT3D-CM [193]	Transformer	/	90K	1.6
LP Conformer [36]	Conformer	570	100K	0.9
`Llama-AVSR`	AV-HuBERT AV	59	433	1.3
`Llama-AVSR`	Whisper + AV-HuBERT V	**57**	433	**0.95**
`Llama-AVSR`	Whisper + AV-HuBERT V	**57**	1756	**0.77**

WER results of 1.5% and 1.1%, respectively. Remarkably, our approach only requires **42**M trainable parameters, which are far fewer compared to all the other methods. We also report the WER achieved by fine-tuning completely a Whisper encoder-decoder model on LRS3 as reported in [185]. Not only does this approach attain a WER much higher than `Llama-AVSR` (2.3% vs 1.1%), but it also requires the updates of more than 1.5B parameters. Moreover, their model harnesses Whisper-Large while `Llama-AVSR` uses the Medium-size version. We also report the case in which AV-HuBERT is used as the audio encoder in place of Whisper ("AV-HuBERT A").

For the video-only setting, `Llama-AVSR` outstrips VSP-LLM, which is the only prior LLM-based method that exploits a pre-trained video encoder (AV-HuBERT) and LLM (Llama2-7B) and adds an extra task during training (visual speech translation). In addition to this, when we train on 433 hours we see that `Llama-AVSR` outperforms various methods like auto-avsr and AV-HuBERT. However, the gap with methods like RAVEN and BRAVEn is more noticeable. We speculate that more sophisticated projectors such as those proposed in recent works [34, 129] could produce more fine-grained and expressive tokens and so bridge this gap, yet we leave it for future works. We also observe that adding more training data results in further improvement, achieving performance parity with respect to auto-avsr and RAVEn. Finally, in Table 4.1 we also report the WER obtained by a recent method, LP Conformer [36], which uses around 100K hours, showcasing that for the task of VSR scaling the training data leads to superior performance.

Finally, for the task of AVSR, we obtain two new SOTA of 0.95% and 0.77% when using 433 and 1756 hours. This is achieved by using Whisper as the audio encoder and AV-HuBERT as the video encoder. If, instead, we use AV-HuBERT to process both audio and video ("AV-HuBERT AV"), the results are slightly worse, in line with what we observed for the ASR task. We point out that our model outperforms methods that exploit tens of thousands of hours [36, 193]. By comparing with the ASR results, we notice that the additional use of video data is more helpful when we train on LRS3 only (1.1% $\rightarrow$ 0.95%), whereas the gain is reduced when more data are available (0.81% $\rightarrow$ 0.77%), and this trend in line with previous works [135, 151].

4.5 Analysis on the Key Factors of `Llama-AVSR`

Exploring Different Encoders and LLMs for `Llama-AVSR`. We conduct multiple ablation studies on various components of `Llama-AVSR` to understand their impact on the final performance. We focus on the setting

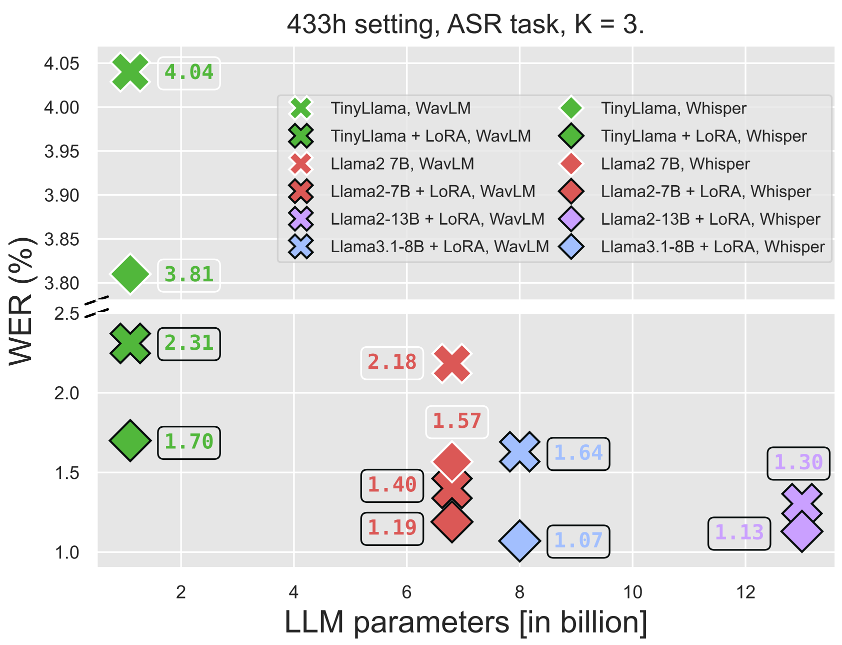


Figure 4.2: WERs of multiple `Llama-AVSR` configurations for ASR task.

with 433 hours. For the ASR task, we study four different LLMs from the Llama family with increasing size: *TinyLlama* (1.1B parameters), *Llama2-7B*, *Llama2-13B*, and the more recent *Llama3.1-8B*. These LLMs are depicted in different colors in Figure 4.2. In addition to this, we ablate the choice of the pre-trained audio encoder, and we compare *Whisper-medium* [181] with *WavLM Large* [43] (diamond and cross markers, respectively). Both encoders have around 300M parameters. We also report the WER obtained *with* (black outline's marker) and *without* (white outline's marker) LoRA at the LLM side. In Figure 4.2, we can observe the following trends: **1)** adding LoRA to the LLM is highly beneficial regardless of the LLM and encoder used. For example, for TinyLlama + WavLM/Whisper, we can improve the performances from WERs of 4.04/3.81% to 2.31/1.70%. A similar trend is noticed for Llama2-7B, albeit the gain is reduced. **2)** The choice of the encoder is also crucial, and this trend becomes more evident when LoRA is not used. This is because better encoded representations enhance the LLM's ability to comprehend, especially in scenarios without LoRA, which typically assists the LLM in aligning the encoded speech features with the pre-learned

Table 4.2: WER results for different configurations of `Llama-AVSR` for the VSR task. "/" means LoRA is not used.

Encoder	**LoRA Position**	**LLM**	
		Tiny-Llama	**Llama3.1-8B**
AV-HuBERT	/	30.2	28.4
AV-HuBERT	LLM	29.2	26.9
AV-HuBERT	LLM + enc	**28.3**	**25.3**
RAVEn	LLM + enc	**34.2**	**32.4**

textual space. **3)** We observe similar performance when LoRA and Whisper are used among the three biggest LLMs, although LLama3.1-8B surpasses Llama2-13B while using almost half as many parameters. Finally, we highlight how a smaller and less powerful LLM such as TinyLLama can surpass Llama2-7B and approach Llama3.1-8B when the right configuration is set, showcasing the importance of the audio encoder and LoRA module.

For VSR, we compare two video encoders: *AV-HuBERT* [196] and *RAVEn* [83]. Since the LoRA module is applied both to the LLM and video encoder, we study 3 configurations: **1)** no LoRA module is applied, **2)** the LoRA module is added to the LLM, and **3)** LoRA modules are applied to both. Table 4.2 shows that applying LoRA modules to both leads to gains of almost 3 points for both TinyLlama and Llama3.1-8B. Furthermore, AV-HuBERT achieves much better performance than RAVEn. This could be attributed to the fact that the hidden units discovered through clustering SSL features (e.g., HuBERT [98], AV-HuBERT [196]) are closely related to linguistic information, such as phonemes. This relationship may allow the LLM to more easily adapt the representations of cluster-based SSL models, compared to others (e.g., RAVEn).

Efficiency-Performance Trade-off for ASR and VSR Tasks. Since most of the computational load lies in the LLM, it is quite common to reduce the number of tokens. At the same time, the loss in resolution inevitably results in a performance's drop. Consequently, it is crucial to find a compromise in terms of efficiency and performance. In this direction, we report the

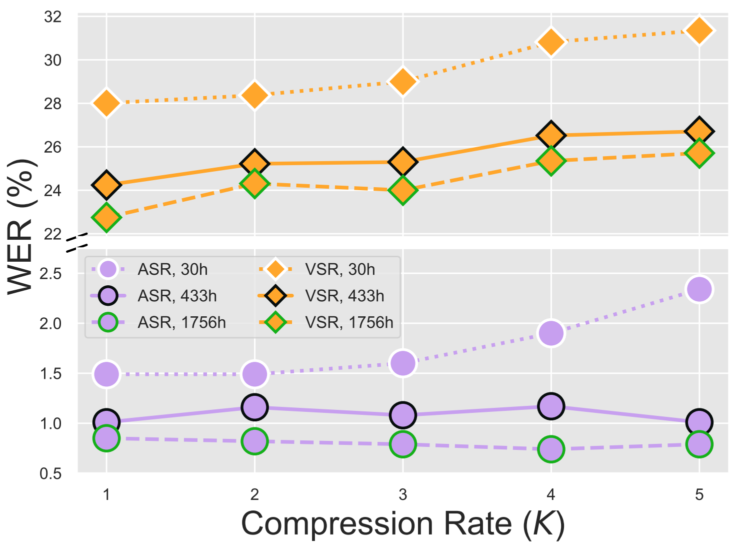


Figure 4.3: WER trend as a function of K for the ASR and VSR tasks.

WER by varying the compression rate K in the range of [1-5] for both the ASR and VSR tasks. For ASR, a compression rate of 1 (i.e., no compression) means the LLM processes on average 584 audio tokens, while the LLM processes 117 tokens for 5. Figure 4.3 shows that for the ASR task we can compress the audio tokens up to a factor 5 without impacting performance, consistently reducing the number of tokens processed by the LLM. Reasonably, for the 30h setting the pooling factor is more crucial as the model is trained on substantially less data. For VSR, however, increasing K leads to gradually worse results across all settings (the WER increase is around 2.5-3 points when pushing K to 5), indicating the need for a more careful balance between efficiency and performance.

AVSR Ablation Studies. We study the optimal audio and video compression rates for the AVSR task (433h setting). We use Whisper and AV-HuBERT as audio and video encoders. We point out that the temporal resolution of Whisper's output features is 50 fps, whereas that of video features is 25 fps. We report the results when tested with different acoustic noise levels as well as in a clean setting (SNR level $= \infty$) in Table 4.3. Similar to [151], we inject babble noise from the NOISEX dataset. We also include the performance when the video modality is not included (i.e., ASR task)

Table 4.3: Results of `Llama-AVSR` for various audio and video compression rates in noisy conditions. ∞ means no noise is injected.

Compression Rate		**SNR Level (dB)**					
A	**V**	∞	5	2	0	-2	-5
3	/	1.1	3.0	6.5	12.3	23.4	63.1
2	4	1.0	2.3	3.9	4.5	10.0	18.2
3	3	1.1	2.2	3.9	4.4	10.0	17.6
4	2	**0.9**	2.2	3.8	**4.2**	**9.5**	16.9
4	1	**0.9**	2.3	**3.7**	**4.2**	**9.5**	**16.4**

and we see that the performance highly deteriorates as the noise increases. Instead, the additional use of the video tokens is beneficial as it compensates for the noisy audio tokens. The best configurations suggest that `Llama-AVSR` can tolerate higher compression rates for the audio tokens (up to 4) compared to video tokens, both for the clean and noisy settings. This is supported by the results presented in Figure 4.3, which revealed that audio tokens can tolerate a higher K compared to video, and by the fact that in extreme noisy conditions (SNR level $= -2/5$ dB) the LLM relies more on video tokens. Finally, we find that without compressing the video tokens (i.e., the last row), we obtain no performance gain except in the case of severe noise conditions.

4.6 Final Remarks

In this work, we present `Llama-AVSR`, a Multimodal LLM that leverages pre-trained audio/video encoders and an LLM for the tasks of ASR, VSR, and AVSR. By training only lightweight projectors and LoRA modules, we endow a pre-trained LLM with audio and visual speech recognition abilities and understanding. `Llama-AVSR` achieves new state-of-the-art results on the LRS3 dataset for the tasks of ASR and AVSR, and comparable performance with previous works for VSR. We also shed light on the key factors that contribute to the powerful results obtained by `Llama-AVSR`: the choice of the pre-trained encoders and LLM, the use of LoRA modules, and the optimal audio and video compression rates to balance efficiency and performance.

Chapter 5

Conclusion

In this thesis, we have explored how to transfer and adapt the knowledge of pre-trained models in the speech domain and beyond. We have discussed the benefits and limitations of full fine-tuning, showcasing its potential drawbacks in terms of computational cost and preservation of prior knowledge in different contexts.

To address these challenges, we propose several innovative techniques. We start by defining a class-incremental learning setting for the tasks of spoken language understanding (SLU) and acoustic scene classification. For SLU, we propose multiple techniques combining the principles of experience replay and knowledge distillation to mitigate catastrophic forgetting for encoder-classifier models [27] and seq2seq auto-regressive models (i.e., encoder-decoder) [32]. In the audio domain, experience replay is combined with mutual information to learn both task-specific and task-agnostic knowledge [239].

Then, we focus on adapting efficiently large pre-trained audio and speech foundation models to several downstream tasks. To do so, we introduce a new framework whereby we can study the performance achieved by different PETL methods on multiple audio and speech benchmarks. Furthermore, to enhance the existing PEFT methods we propose two novel methods: 1) we advance a new adapter design that exploits the convolution module of the conformer architecture to boost the performance while keeping the parameter's count minimal [29]. The key ingredient is the introduction of the depthwise convolution that captures spatial correlations. This new adapter module allows us to obtain on average better performance results than full fine-tuning the model whilst updating only **0.29/0.59**% parameters compared to it. 2) We leverage the Mixture of Experts (MoE) paradigm to scale the number of adapters while keeping the computational cost limited as well as updating only a small fraction of the parameters. In this way, we show that our proposed SMoA method combines the benefits of both MoE and PETL to outperform the baseline with a single adapter.

Finally, we propose `Llama-AVSR`, a new multimodal LLM with strong audio-visual speech recognition abilities to carry out the tasks of ASR, VSR, and AVSR. It leverages pre-trained audio and video encoders to produce modality-specific tokens which, together with the text tokens, are processed by a pre-trained LLM (e.g., Llama3.1-8B) to yield the resulting response in an auto-regressive fashion. In this way, we are capable of adapting the knowledge of unimodal pre-trained models (audio, video, and text) to build a multimodal system that processes all the modalities at the same time. `Llama-AVSR` pushes the boundaries of audio-visual speech recognition by setting new state-of-the-art results in the largest audio-visual benchmark, LRS3 [4]. Interestingly, the parameter-efficient transfer learning leitmotiv is also present in `Llama-AVSR` since the all pre-trained models are kept frozen and just equipped with lightweight LoRA modules to facilitate the multimodal alignment.

5.1 Future Directions

We propose several promising avenues for future research to further enhance our results.

Combining Continual Learning with PETL and MLLMs. A compelling research direction involves integrating continual learning with PETL and MLLMs. This would combine together the three main topics of this thesis. A suitable application for this scenario is multi-lingual audio-visual speech recognition, where the MLLM must learn multiple languages sequentially. To mitigate catastrophic forgetting, we can equip the model with a novel pool of task-specific prompts or adapter modules.

Improving Visual Feature Representations. While Llama-AVSR sets new state-of-the-art performance results for ASR and AVSR tasks, a performance gap persists in VSR compared to certain state-of-the-art methods. We speculate that employing a simple linear projector to map video tokens into the LLM space may not be optimal, necessitating a more sophisticated design. Recent research has explored advanced projector techniques. For example, [34] introduces two locality-enhanced projectors that preserve local visual feature context efficiently. Another interesting line of research leverages feature representations from multiple layers of the visual encoder, rather than solely relying on the final layer output, enabling the LLM to process multi-granularity visual features [26, 242].

Better Compression Techniques and Model Scaling. We have observed that Llama-AVSR's performance is sensitive to the number of processed tokens. Excessive compression can significantly degrade WER results. Therefore, investigating advanced compression techniques that minimize token count while preserving performance is essential for faster inference. Furthermore, our findings indicate that larger LLMs yield superior results. This motivates the exploration of knowledge distillation techniques to bridge the performance gap between LLMs of varying sizes, thereby enabling the deployment of powerful LLMs even in resource-constrained scenarios.

Chapter 6

Acknowledgments

Multiple papers in this thesis has been funded by different projects.

[30] has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101135798, project Meetween (My Personal AI Mediator for Virtual MEETtings BetWEEN People).

[29] and [28] have been funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission-EU.

Bibliography

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Tameem Adel, Han Zhao, and Richard E Turner. Continual learning with adaptive weights (claw). *arXiv preprint arXiv:1911.09514*, 2019.

[3] Triantafyllos Afouras, Joon Son Chung, Andrew Senior, Oriol Vinyals, and Andrew Zisserman. Deep audio-visual speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, 44(12):8717–8727, 2018.

[4] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman. Lrs3-ted: a large-scale dataset for visual speech recognition. *arXiv preprint arXiv:1809.00496*, 2018.

[5] Bhuvan Agrawal, Markus Müller, Samridhi Choudhary, Martin Radfar, Athanasios Mouchtaris, Ross McGowan, Nathan Susanj, and Siegfried Kunzmann. Tie your embeddings down: Cross-modal latent spaces for end-to-end spoken language understanding. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7157–7161. IEEE, 2022.

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

few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[7] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018.

[8] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

[9] Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. *arXiv preprint arXiv:2110.07560*, 2021.

[10] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1, 2024.

[11] Siddhant Arora, Siddharth Dalmia, Pavel Denisov, Xuankai Chang, Yushi Ueda, Yifan Peng, Yuekai Zhang, Sujay Kumar, Karthik Ganesan, Brian Yan, et al. Espnet-slu: Advancing spoken language understanding through espnet. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7167–7171. IEEE, 2022.

[12] Jimmy Lei Ba. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[13] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019.

[14] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[15] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[16] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[17] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8218–8227, 2021.

[18] Daniele Barchiesi, Dimitrios Giannoulis, Dan Stowell, and Mark D Plumbley. Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, 32(3):16–34, 2015.

[19] Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. Slurp: A spoken language understanding resource package. *arXiv preprint arXiv:2011.13205*, 2020.

[20] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 583–592, 2019.

[21] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[22] Malik Boudiaf, Imtiaz Ziko, Jérôme Rony, José Dolz, Pablo Piantanida, and Ismail Ben Ayed. Information maximization for few-shot learning. *Advances in Neural Information Processing Systems*, 33:2445–2457, 2020.

[23] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[24] M. Burchi et al. Multilingual audio-visual speech recognition with hybrid ctc/rnn-t fast conformer. In *ICASSP*, pages 10211–10215, 2024.

[25] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42:335–359, 2008.

[26] Yue Cao, Yangzhou Liu, Zhe Chen, Guangchen Shi, Wenhai Wang, Danhuai Zhao, and Tong Lu. Mmfuser: Multimodal multi-layer feature fuser for fine-grained vision-language understanding. *arXiv preprint arXiv:2410.11829*, 2024.

[27] Umberto Cappellazzo, Daniele Falavigna, and Alessio Brutti. An investigation of the combination of rehearsal and knowledge distillation in continual learning for spoken language understanding. In *Interspeech*, 2023.

[28] Umberto Cappellazzo, Daniele Falavigna, and Alessio Brutti. Efficient fine-tuning of audio spectrogram transformers via soft mixture of adapters. In *Interspeech*, 2024.

[29] Umberto Cappellazzo, Daniele Falavigna, Alessio Brutti, and Mirco Ravanelli. Parameter-efficient transfer learning of audio spectrogram transformers. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2024.

[30] Umberto Cappellazzo, Enrico Fini, Muqiao Yang, Daniele Falavigna, Alessio Brutti, and Bhiksha Raj. Continual contrastive spoken language understanding. In *ACL (Findings)*, 2024.

[31] Umberto Cappellazzo, Minsu Kim, Honglie Chen, Pingchuan Ma, Stavros Petridis, Daniele Falavigna, Alessio Brutti, and Maja Pan-

tic. Large language models are strong audio-visual speech recognition learners. In *ICASSP*, 2025.

[32] Umberto Cappellazzo, Muqiao Yang, Daniele Falavigna, and Alessio Brutti. Sequence-level knowledge distillation for class-incremental end-to-end spoken language understanding. In *Interspeech*, 2023.

[33] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 9516–9525, 2021.

[34] Junbum Cha, Wooyoung Kang, Jonghwan Mun, and Byungseok Roh. Honeybee: Locality-enhanced projector for multimodal llm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13817–13827, 2024.

[35] Heng-Jui Chang, Shu-wen Yang, and Hung-yi Lee. Distilhubert: Speech representation learning by layer-wise distillation of hidden-unit bert. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7087–7091. IEEE, 2022.

[36] Oscar Chang, Hank Liao, Dmitriy Serdyuk, Ankit Shahy, and Olivier Siohan. Conformer is all you need for visual speech recognition. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10136–10140. IEEE, 2024.

[37] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–547, 2018.

[38] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

[39] Chen Chen, Ruizhe Li, Yuchen Hu, Sabato Marco Siniscalchi, Pin-Yu Chen, Ensiong Chng, and Chao-Han Huck Yang. It's never too late: Fusing acoustic information into large language models for automatic speech recognition. *arXiv preprint arXiv:2402.05457*, 2024.

[40] Hao Chen, Ran Tao, Han Zhang, Yidong Wang, Xiang Li, Wei Ye, Jindong Wang, Guosheng Hu, and Marios Savvides. Conv-adapter: Exploring parameter efficient transfer learning for convnets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1551–1561, 2024.

[41] Jiaao Chen, Aston Zhang, Xingjian Shi, Mu Li, Alex Smola, and Diyi Yang. Parameter-efficient fine-tuning design spaces. *arXiv preprint arXiv:2301.01821*, 2023.

[42] Lichang Chen, Heng Huang, and Minhao Cheng. Ptp: Boosting stability and performance of prompt tuning with perturbation-based regularizer. *arXiv preprint arXiv:2305.02423*, 2023.

[43] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[44] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022.

[45] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[46] Zhiyuan Chen and Bing Liu. *Lifelong machine learning.* Springer Nature, 2022.

[47] Zih-Ching Chen, Chin-Lun Fu, Chih-Ying Liu, Shang-Wen Daniel Li, and Hung-yi Lee. Exploring efficient-tuning methods in self-supervised speech models. In *2022 IEEE spoken language technology workshop (SLT)*, pages 1120–1127. IEEE, 2023.

[48] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024.

[49] Alexandra Chronopoulou, Matthew E Peters, Alexander Fraser, and Jesse Dodge. Adaptersoup: Weight averaging to improve generalization of pretrained language models. *arXiv preprint arXiv:2302.07027*, 2023.

[50] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.

[51] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

[52] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *arXiv preprint arXiv:2309.16588*, 2023.

[53] Sarkar Snigdha Sarathi Das, Ranran Haoran Zhang, Peng Shi, Wenpeng Yin, and Rui Zhang. Unified low-resource sequence labeling by sample-aware dynamic sparse finetuning. *arXiv preprint arXiv:2311.03748*, 2023.

[54] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[55] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[56] Soham Deshmukh, Benjamin Elizalde, Rita Singh, and Huaming Wang. Pengi: An audio language model for audio tasks. *Advances in Neural Information Processing Systems*, 36:18090–18108, 2023.

[57] Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint arXiv:1810.13166*, 2018.

[58] Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. *arXiv preprint arXiv:2311.11696*, 2023.

[59] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[60] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[61] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision–ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part XX 16*, pages 86–102. Springer, 2020.

[62] Arthur Douillard and Timothée Lesort. Continuum: Simple management of complex continual learning scenarios. *arXiv preprint arXiv:2102.06253*, 2021.

[63] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dy-

namic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295, 2022.

[64] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[65] Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*, 2022.

[66] Yassir Fathullah, Chunyang Wu, Egor Lakomkin, Junteng Jia, Yuan Shangguan, Ke Li, Jinxi Guo, Wenhan Xiong, Jay Mahadeokar, Ozlem Kalinli, et al. Prompting large language models with speech recognition abilities. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13351–13355. IEEE, 2024.

[67] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[68] Enrico Fini, Victor G Turrisi Da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2022.

[69] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[70] Rui Gao and Weiwei Liu. Ddgr: Continual learning with deep diffusion-based generative replay. In *International Conference on Machine Learning*, pages 10744–10763. PMLR, 2023.

[71] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter.

Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE, 2017.

[72] Mozhdeh Gheini, Xiang Ren, and Jonathan May. Cross-attention is all you need: Adapting pretrained transformers for machine translation. *arXiv preprint arXiv:2104.08771*, 2021.

[73] Sreyan Ghosh, Sonal Kumar, Ashish Seth, Chandra Kiran Reddy Evuru, Utkarsh Tyagi, S Sakshi, Oriol Nieto, Ramani Duraiswami, and Dinesh Manocha. Gama: A large audio-language model with advanced audio understanding and complex reasoning abilities. *arXiv preprint arXiv:2406.11768*, 2024.

[74] Yuan Gong, Yu-An Chung, and James Glass. Ast: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*, 2021.

[75] Yuan Gong, Sameer Khurana, Leonid Karlinsky, and James Glass. Whisper-at: Noise-robust automatic speech recognizers are also strong general audio event taggers. *arXiv preprint arXiv:2307.03183*, 2023.

[76] Yuan Gong, Cheng-I Lai, Yu-An Chung, and James Glass. Ssast: Self-supervised audio spectrogram transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10699–10709, 2022.

[77] Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

[78] Stephen T Grossberg. *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*, volume 70. Springer Science & Business Media, 2012.

[79] Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A survey on self-supervised learning: Algorithms, ap-

plications, and future trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[80] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

[81] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020.

[82] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

[83] Alexandros Haliassos, Pingchuan Ma, Rodrigo Mira, Stavros Petridis, and Maja Pantic. Jointly learning visual and auditory speech representations from raw data. *arXiv preprint arXiv:2212.06246*, 2022.

[84] Alexandros Haliassos, Andreas Zinonos, Rodrigo Mira, Stavros Petridis, and Maja Pantic. Braven: Improving self-supervised pretraining for visual and auditory speech recognition. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11431–11435. IEEE, 2024.

[85] Zeyu Han, Chao Gao, Jinyang Liu, Sai Qian Zhang, et al. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[86] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

[87] Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. Sensitivity-aware visual parameter-efficient fine-tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11825–11835, 2023.

[88] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[89] Shwai He, Run-Ze Fan, Liang Ding, Li Shen, Tianyi Zhou, and Dacheng Tao. Mera: Merging pretrained adapters for few-shot learning. *arXiv preprint arXiv:2308.15982*, 2023.

[90] Xuehai He, Chunyuan Li, Pengchuan Zhang, Jianwei Yang, and Xin Eric Wang. Parameter-efficient model adaptation for vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 817–825, 2023.

[91] Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen. Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions. *arXiv preprint arXiv:2005.14623*, 2020.

[92] Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International conference on machine learning*, pages 4182–4192. PMLR, 2020.

[93] Kristi Hendrickson, Matthew Walenski, Margaret Friend, and Tracy Love. The organization of words and environmental sounds in memory. *Neuropsychologia*, 69:67–76, 2015.

[94] Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[95] James Horlock and Simon King. Discriminative methods for improving named entity extraction on speech data. 2003.

[96] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 831–839, 2019.

[97] 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 *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

[98] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460, 2021.

[99] Wei-Ning Hsu and Bowen Shi. u-hubert: Unified mixed-modal speech pretraining and zero-shot transfer to unlabeled modality. *Advances in Neural Information Processing Systems*, 35:21157–21170, 2022.

[100] Y Hsu. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

[101] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[102] Yuchen Hu, Chen Chen, Chao-Han Huck Yang, Ruizhe Li, Chao Zhang, Pin-Yu Chen, and EnSiong Chng. Large language models are efficient learners of noise-robust speech recognition. *arXiv preprint arXiv:2401.10446*, 2024.

[103] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

[104] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.

[105] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[106] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[107] Shibo Jie and Zhi-Hong Deng. Convolutional bypasses are better vision transformer adapters. *arXiv preprint arXiv:2207.07039*, 2022.

[108] Quentin Jodelet, Xin Liu, Yin Jun Phua, and Tsuyoshi Murata. Class-incremental learning using diffusion model for distillation and replay. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3425–3433, 2023.

[109] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetful learning for domain expansion in deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[110] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16071–16080, 2022.

[111] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021.

[112] Samuel Kessler, Bethan Thomas, and Salah Karout. An adapter based pre-training for efficient and scalable self-supervised speech representation learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3179–3183. IEEE, 2022.

[113] 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*, 33:18661–18673, 2020.

[114] Sungnyun Kim, Kangwook Jang, Sangmin Bae, Hoirin Kim, and Se-Young Yun. Learning video temporal dynamics with cross-modal attention for robust audio-visual speech recognition. *arXiv preprint arXiv:2407.03563*, 2024.

[115] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[116] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[117] 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*, 114(13):3521–3526, 2017.

[118] Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. *arXiv preprint arXiv:2402.01831*, 2024.

[119] Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

[120] Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Maxim Bazhenov, Douglas Blackiston, Josh Bongard, Andrew P Brna, Suraj

Chakravarthi Raja, Nick Cheney, Jeff Clune, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

[121] Gukyeong Kwon, Zhaowei Cai, Avinash Ravichandran, Erhan Bas, Rahul Bhotika, and Stefano Soatto. Masked vision and language modeling for multi-modal representation learning. *arXiv preprint arXiv:2208.02131*, 2022.

[122] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR, 2018.

[123] Tao Lei, Junwen Bai, Siddhartha Brahma, Joshua Ainslie, Kenton Lee, Yanqi Zhou, Nan Du, Vincent Zhao, Yuexin Wu, Bo Li, et al. Conditional adapters: Parameter-efficient transfer learning with fast inference. *Advances in Neural Information Processing Systems*, 36:8152–8172, 2023.

[124] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information fusion*, 58:52–68, 2020.

[125] Timothée Lesort, Andrei Stoian, and David Filliat. Regularization shortcomings for continual learning. *arXiv preprint arXiv:1912.03049*, 2019.

[126] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[127] 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 *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[128] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[129] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *European Conference on Computer Vision*, pages 323–340. Springer, 2025.

[130] Yingting Li, Ambuj Mehrish, Rishabh Bhardwaj, Navonil Majumder, Bo Cheng, Shuai Zhao, Amir Zadeh, Rada Mihalcea, and Soujanya Poria. Evaluating parameter-efficient transfer learning approaches on sure benchmark for speech understanding. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[131] Yunxin Li, Shenyuan Jiang, Baotian Hu, Longyue Wang, Wanqi Zhong, Wenhan Luo, Lin Ma, and Min Zhang. Uni-moe: Scaling unified multimodal llms with mixture of experts. *arXiv preprint arXiv:2405.11273*, 2024.

[132] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[133] Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*, 2023.

[134] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *Advances in Neural Information Processing Systems*, 35:109–123, 2022.

[135] Jiachen Lian, Alexei Baevski, Wei-Ning Hsu, and Michael Auli. Av-data2vec: Self-supervised learning of audio-visual speech representations with contextualized target representations. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE, 2023.

[136] Baohao Liao, Yan Meng, and Christof Monz. Parameter-efficient fine-tuning without introducing new latency. *arXiv preprint arXiv:2305.16742*, 2023.

[137] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26689–26699, 2024.

[138] Tzu-Han Lin, How-Shing Wang, Hao-Yung Weng, Kuang-Chen Peng, Zih-Ching Chen, and Hung-yi Lee. Peft for speech: Unveiling optimal placement, merging strategies, and ensemble techniques. *arXiv preprint arXiv:2401.02122*, 2024.

[139] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

[140] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.

[141] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[142] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

[143] Xiangyang Liu, Tianxiang Sun, Xuanjing Huang, and Xipeng Qiu. Late prompt tuning: A late prompt could be better than many prompts. *arXiv preprint arXiv:2210.11292*, 2022.

[144] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 2023.

[145] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[146] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[147] Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. Speech model pre-training for end-to-end spoken language understanding. *arXiv preprint arXiv:1904.03670*, 2019.

[148] Yi Luo and Nima Mesgarani. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing*, 27(8):1256–1266, 2019.

[149] Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. Full parameter fine-tuning for large language models with limited resources. *arXiv preprint arXiv:2306.09782*, 2023.

[150] Chenyang Lyu, Minghao Wu, Longyue Wang, Xinting Huang, Bingshuai Liu, Zefeng Du, Shuming Shi, and Zhaopeng Tu. Macaw-llm: Multi-modal language modeling with image, audio, video, and text integration. *arXiv preprint arXiv:2306.09093*, 2023.

[151] Pingchuan Ma, Alexandros Haliassos, Adriana Fernandez-Lopez, Honglie Chen, Stavros Petridis, and Maja Pantic. Auto-avsr: Audio-visual speech recognition with automatic labels. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[152] Pingchuan Ma, Stavros Petridis, and Maja Pantic. End-to-end audio-visual speech recognition with conformers. In *ICASSP 2021-2021 IEEE*

International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7613–7617. IEEE, 2021.

[153] Pingchuan Ma, Stavros Petridis, and Maja Pantic. Visual speech recognition for multiple languages in the wild. *Nature Machine Intelligence*, 4(11):930–939, 2022.

[154] Ziyang Ma, Guanrou Yang, Yifan Yang, Zhifu Gao, Jiaming Wang, Zhihao Du, Fan Yu, Qian Chen, Siqi Zheng, Shiliang Zhang, et al. An embarrassingly simple approach for llm with strong asr capacity. *arXiv preprint arXiv:2402.08846*, 2024.

[155] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489*, 2021.

[156] Ilaria Manco, Emmanouil Benetos, Elio Quinton, and György Fazekas. Contrastive audio-language learning for music. *arXiv preprint arXiv:2208.12208*, 2022.

[157] Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*, 2021.

[158] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

[159] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[160] Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, et al. Mm1: Methods, analysis & insights from multimodal llm pre-training. *arXiv preprint arXiv:2403.09611*, 2024.

[161] Ambuj Mehrish, Abhinav Ramesh Kashyap, Li Yingting, Navonil Majumder, and Soujanya Poria. Adaptermix: Exploring the efficacy of mixture of adapters for low-resource tts adaptation. *arXiv preprint arXiv:2305.18028*, 2023.

[162] Martin Menabue, Emanuele Frascaroli, Matteo Boschini, Enver Sangineto, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Semantic residual prompts for continual learning. In *ECCV*, 2024.

[163] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. Assessment of human and machine performance in acoustic scene classification: Dcase 2016 case study. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 319–323. IEEE, 2017.

[164] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539, 2014.

[165] Fei Mi, Liangwei Chen, Mengjie Zhao, Minlie Huang, and Boi Faltings. Continual learning for natural language generation in task-oriented dialog systems. *arXiv preprint arXiv:2010.00910*, 2020.

[166] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[167] Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34:30298–30312, 2021.

[168] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[169] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

[170] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[171] Dongmin Park, Seokil Hong, Bohyung Han, and Kyoung Mu Lee. Continual learning by asymmetric loss approximation with single-side overestimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3335–3344, 2019.

[172] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

[173] Yifan Peng, Siddhant Arora, Yosuke Higuchi, Yushi Ueda, Sujay Kumar, Karthik Ganesan, Siddharth Dalmia, Xuankai Chang, and Shinji Watanabe. A study on the integration of pre-trained ssl, asr, lm and slu models for spoken language understanding. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 406–413. IEEE, 2023.

[174] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020.

[175] Karol J Piczak. Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th international workshop on machine learning for signal processing (MLSP)*, pages 1–6. IEEE, 2015.

[176] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.

[177] Maciej Pióro, Kamil Ciebiera, Krystian Król, Jan Ludziejewski, and Sebastian Jaszczur. Moe-mamba: Efficient selective state space models with mixture of experts. *arXiv preprint arXiv:2401.04081*, 2024.

[178] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. *arXiv preprint arXiv:2308.00951*, 2023.

[179] Libo Qin, Tianbao Xie, Wanxiang Che, and Ting Liu. A survey on spoken language understanding: Recent advances and new frontiers. *arXiv preprint arXiv:2103.03095*, 2021.

[180] 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 *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[181] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[182] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[183] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[184] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

[185] Andrew Rouditchenko, Yuan Gong, Samuel Thomas, Leonid Karlinsky, Hilde Kuehne, Rogerio Feris, and James Glass. Whisper-flamingo: Integrating visual features into whisper for audio-visual speech recognition and translation. *arXiv preprint arXiv:2406.10082*, 2024.

[186] Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Tutorials*, pages 15–18, 2019.

[187] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 1041–1044, 2014.

[188] Michael Saxon, Samridhi Choudhary, Joseph P McKenna, and Athanasios Mouchtaris. End-to-end spoken language understanding for generalized voice assistants. *arXiv preprint arXiv:2106.09009*, 2021.

[189] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pages 4528–4537. PMLR, 2018.

[190] Nithish Muthuchamy Selvaraj, Xiaobao Guo, Adams Kong, Bingquan Shen, and Alex Kot. Adapter incremental continual learning of efficient audio spectrogram transformers. *arXiv preprint arXiv:2302.14314*, 2023.

[191] Rico Sennrich. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[192] Seunghyun Seo, Donghyun Kwak, and Bowon Lee. Integration of pre-trained networks with continuous token interface for end-to-end spoken language understanding. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7152–7156. IEEE, 2022.

[193] Dmitriy Serdyuk, Otavio Braga, and Olivier Siohan. Transformer-based video front-ends for audio-visual speech recognition for single and multi-person video. *arXiv preprint arXiv:2201.10439*, 2022.

[194] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[195] Yilin Shen, Xiangyu Zeng, and Hongxia Jin. A progressive model to enable continual learning for semantic slot filling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1279–1284, 2019.

[196] Bowen Shi, Wei-Ning Hsu, Kushal Lakhotia, and Abdelrahman Mohamed. Learning audio-visual speech representation by masked multimodal cluster prediction. *arXiv preprint arXiv:2201.02184*, 2022.

[197] Bowen Shi, Wei-Ning Hsu, and Abdelrahman Mohamed. Robust self-supervised audio-visual speech recognition. *arXiv preprint arXiv:2201.01763*, 2022.

[198] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

[199] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio

Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11909–11919, 2023.

[200] Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. On transferability of prompt tuning for natural language processing. *arXiv preprint arXiv:2111.06719*, 2021.

[201] Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. *arXiv preprint arXiv:2204.04763*, 2022.

[202] Siqi Sun, Zhe Gan, Yu Cheng, Yuwei Fang, Shuohang Wang, and Jingjing Liu. Contrastive distillation on intermediate representations for language model compression. *arXiv preprint arXiv:2009.14167*, 2020.

[203] Yi-Lin Sung, Varun Nair, and Colin A Raffel. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205, 2021.

[204] Richard S Sutton. Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 8, 1986.

[205] Naoya Takahashi, Michael Gygli, Beat Pfister, and Luc Van Gool. Deep convolutional neural networks and data augmentation for acoustic event detection. *arXiv preprint arXiv:1604.07160*, 2016.

[206] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019.

[207] Katrin Tomanek, Vicky Zayats, Dirk Padfield, Kara Vaillancourt, and Fadi Biadsy. Residual adapters for parameter-efficient asr adaptation to atypical and accented speech. *arXiv preprint arXiv:2109.06952*, 2021.

[208] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv:2406.16860*, 2024.

[209] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[210] Yao-Hung Hubert Tsai, Martin Q Ma, Muqiao Yang, Han Zhao, Louis-Philippe Morency, and Ruslan Salakhutdinov. Self-supervised representation learning with relative predictive coding. *arXiv preprint arXiv:2103.11275*, 2021.

[211] Gokhan Tur and Renato De Mori. *Spoken language understanding: Systems for extracting semantic information from speech.* John Wiley & Sons, 2011.

[212] Michele Valenti, Stefano Squartini, Aleksandr Diment, Giambattista Parascandolo, and Tuomas Virtanen. A convolutional neural network approach for acoustic scene classification. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1547–1554. IEEE, 2017.

[213] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[214] A Varga. Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Elsevier Speech Commun*, 2(3):247, 1992.

[215] Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer. *arXiv preprint arXiv:2110.07904*, 2021.

[216] Danilo Vucetic, Mohammadreza Tayaranian, Maryam Ziaeefard, James J Clark, Brett H Meyer, and Warren J Gross. Efficient fine-tuning of bert models on the edge. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1838–1842. IEEE, 2022.

[217] Liyuan Wang, Mingtian Zhang, Zhongfan Jia, Qian Li, Chenglong Bao, Kaisheng Ma, Jun Zhu, and Yi Zhong. Afec: Active forgetting of negative transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22379–22391, 2021.

[218] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[219] Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 184–193, 2021.

[220] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[221] Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. Adamix: Mixture-of-adaptations for parameter-efficient model tuning. *arXiv preprint arXiv:2205.12410*, 2022.

[222] Yingzhi Wang, Abdelmoumene Boumadane, and Abdelwahab Heba. A fine-tuned wav2vec 2.0/hubert benchmark for speech emotion recognition, speaker verification and spoken language understanding. *arXiv preprint arXiv:2111.02735*, 2021.

[223] Zhen Wang, Liu Liu, Yajing Kong, Jiaxian Guo, and Dacheng Tao. Online continual learning with contrastive vision transformer. In *European Conference on Computer Vision*, pages 631–650. Springer, 2022.

[224] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022.

[225] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[226] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

[227] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[228] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[229] T Wolf et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[230] George August Wright, Umberto Cappellazzo, Salah Zaiem, Desh Raj, Lucas Ondel Yang, Daniele Falavigna, Mohamed Nabih Ali, and Alessio Brutti. Training early-exit architectures for automatic speech recognition: Fine-tuning pre-trained models or training from scratch. In *2024*

IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW), pages 685–689, 2024.

[231] Junda Wu, Tong Yu, Rui Wang, Zhao Song, Ruiyi Zhang, Handong Zhao, Chaochao Lu, Shuai Li, and Ricardo Henao. Infoprompt: Information-theoretic soft prompt tuning for natural language understanding. *Advances in Neural Information Processing Systems*, 36, 2023.

[232] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 374–382, 2019.

[233] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, VG Vydiswaran, and Hao Ma. Idpg: An instance-dependent prompt generation method. *arXiv preprint arXiv:2204.04497*, 2022.

[234] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6619–6628, 2019.

[235] Yang Xiao, Xubo Liu, James King, Arshdeep Singh, Eng Siong Chng, Mark D Plumbley, and Wenwu Wang. Continual learning for on-device environmental sound classification. *arXiv preprint arXiv:2207.07429*, 2022.

[236] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148*, 2023.

[237] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3014–3023, 2021.

[238] Guanglei Yang, Enrico Fini, Dan Xu, Paolo Rota, Mingli Ding, Moin Nabi, Xavier Alameda-Pineda, and Elisa Ricci. Uncertainty-aware contrastive distillation for incremental semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2567–2581, 2022.

[239] Muqiao Yang, Umberto Cappellazzo, Xiang Li, and Bhiksha Raj. Improving continual learning of acoustic scene classification via mutual information optimization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7105–7109, 2024.

[240] Muqiao Yang, Ian Lane, and Shinji Watanabe. Online continual learning of end-to-end speech recognition models. In *Interspeech*, 2022.

[241] Muqiao Yang, Xiang Li, Umberto Cappellazzo, Shinji Watanabe, and Bhiksha Raj. Evaluating and improving continual learning in spoken language understanding. In *Interspeech*, 2024.

[242] Huanjin Yao, Wenhao Wu, Taojiannan Yang, YuXin Song, Mengxi Zhang, Haocheng Feng, Yifan Sun, Zhiheng Li, Wanli Ouyang, and Jingdong Wang. Dense connector for mllms. *arXiv preprint arXiv:2405.13800*, 2024.

[243] Rong Ye, Mingxuan Wang, and Lei Li. Cross-modal contrastive learning for speech translation. *arXiv preprint arXiv:2205.02444*, 2022.

[244] Jeong Hun Yeo, Seunghee Han, Minsu Kim, and Yong Man Ro. Where visual speech meets language: Vsp-llm framework for efficient and context-aware visual speech processing. *arXiv preprint arXiv:2402.15151*, 2024.

[245] Jeong Hun Yeo, Minsu Kim, Shinji Watanabe, and Yong Man Ro. Visual speech recognition for languages with limited labeled data using automatic labels from whisper. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10471–10475. IEEE, 2024.

[246] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

[247] Shoubin Yu, Jaehong Yoon, and Mohit Bansal. Crema: Multimodal compositional video reasoning via efficient modular adaptation and fusion. *arXiv preprint arXiv:2402.05889*, 2024.

[248] Wenyi Yu, Changli Tang, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Connecting speech encoder and large language model for asr. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12637–12641. IEEE, 2024.

[249] Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermiş, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. *arXiv preprint arXiv:2309.05444*, 2023.

[250] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

[251] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

[252] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

[253] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[254] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022.

[255] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive learning of medical visual representations from paired images and text. In *Machine Learning for Healthcare Conference*, pages 2–25. PMLR, 2022.

[256] Zhen-Ru Zhang, Chuanqi Tan, Haiyang Xu, Chengyu Wang, Jun Huang, and Songfang Huang. Towards adaptive prefix tuning for parameter-efficient language model fine-tuning. *arXiv preprint arXiv:2305.15212*, 2023.

[257] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13208–13217, 2020.

[258] Danpei Zhao, Bo Yuan, and Zhenwei Shi. Inherit with distillation and evolve with contrast: Exploring class incremental semantic segmentation without exemplar memory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):11932–11947, 2023.

[259] Hao Zhao, Jie Fu, and Zhaofeng He. Prototype-based hyperadapter for sample-efficient multi-task tuning. *arXiv preprint arXiv:2310.11670*, 2023.

[260] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[261] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[262] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. *arXiv preprint arXiv:2205.13218*, 2022.

[263] Han Zhou, Xingchen Wan, Ivan Vulić, and Anna Korhonen. Autopeft: Automatic configuration search for parameter-efficient fine-tuning. *Transactions of the Association for Computational Linguistics*, 12:525–542, 2024.

[264] Wei Zhu and Ming Tan. Spt: learning to selectively insert prompts for better prompt tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11862–11878, 2023.

[265] Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. Counter-interference adapter for multilingual machine translation. *arXiv preprint arXiv:2104.08154*, 2021.

[266] Yi Zhu, Zexun Wang, Hang Liu, Peiying Wang, Mingchao Feng, Meng Chen, and Xiaodong He. Cross-modal transfer learning via multi-grained alignment for end-to-end spoken language understanding. In *INTERSPEECH*, pages 1131–1135, 2022.

[267] 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*, 109(1):43–76, 2020.