

# Privacy-preserving large language models using adapters

Pau de Batlle Juan

---



Universitat  
Pompeu Fabra  
*Barcelona*

# PRIVACY-PRESERVING LARGE LANGUAGE MODELS USING ADAPTERS

TREBALL DE FI DE GRAU DE  
Pau de Batlle Juan

Javier Segovia Aguas

Grau en Enginyeria en Sistemes Audiovisuals

Curs 2023 - 2024



Universitat  
Pompeu Fabra  
*Barcelona*

Escola  
d'Enginyeria

# Acknowledgements

First of all I would like to thank my thesis directors Javier Segovia Aguas and Enric Perera i Bel for their support and advice during these last 5 months.

I would also like to show my appreciation to all the T2C team, who has helped me grow both as a professional and also as a person, challenging me to improve and be better.

Finally I am immensely grateful to my family for putting up with my intense schedule during these last weeks. Specially Iris Gallego, for her unconditional support and accompanying me during this adventure.

# Abstract

As Large Language Models (LLMs) grow, computational costs for training and inference increase significantly. Parameter Efficient Fine-Tuning (PEFT) techniques reduce costs by minimizing parameter modifications during re-training. However, fine-tuning often involves private data, risking data leakage or unauthorized access. This research explores adapter-based fine-tuning, a PEFT method adding small trained linear layers, evaluating performance and privacy preservation. Using a 1.6B open-source model and private commercial data, adapters are trained on the full dataset, incremental portions, and non-overlapping subsets to investigate learning capabilities and potential for obtaining varied knowledge from one base model. Benchmarking on standard datasets is also performed to assess any changes in reasoning and knowledge, comparing the adapter-enhanced model to the original. The study demonstrates adapter-based fine-tuning as an efficient, privacy-preserving approach for rapid task adaptation leveraging private data.

# Resum

A mesura que els grans models de llenguatge (LLM) creixen, els costos computacionals d'entrenament i d'inferència augmenten significativament. Les tècniques d'afinació de paràmetres eficient (PEFT) redueixen els costos minimitzant els paràmetres actualitzats durant el reentrenament. No obstant això, l'afinació sovint implica l'ús de dades privades, amb el risc que comporta que puguin sortir a la llum. Aquest treball explora l'afinament basat en adaptadors, un mètode PEFT que afegeix petites capes lineals entrenades per avaluar el rendiment i la preservació de la privadesa. Utilitzant un model de codi obert 1.6B i dades comercials privades, els adaptadors s'entrenen sobre el conjunt de dades complet, parts incrementals i subconjunts no superposats per investigar les capacitats d'aprenentatge i el potencial per obtenir coneixements variats a partir d'un model base. També es realitza una comparació utilitzant conjunts de dades estàndard per avaluar qualsevol canvi en el raonament i el coneixement entre el model original i el modificat. L'estudi demostra l'afinament basat en adaptadors com un enfocament eficient i de preservació de la privadesa per a una adaptació ràpida de tasques aprofitant dades privades.

# Contents

<b>List of figures</b>	<b>iv</b>
<b>List of tables</b>	<b>v</b>
<b>1 INTRODUCTION</b>	<b>1</b>
<b>2 RELATED WORK</b>	<b>3</b>
2.1 Neural Networks & Concepts . . . . .	3
2.2 Attention is All You Need . . . . .	4
2.2.1 Transformer Architecture . . . . .	4
2.2.2 Attention & Multi-Head Attention . . . . .	5
2.3 Parameter Efficient Fine-Tuning . . . . .	7
2.3.1 Prompt Based . . . . .	7
2.3.2 Adapters . . . . .	7
2.4 Task-Specific Adapter Merging . . . . .	8
2.5 Privacy in LLMs . . . . .	9
<b>3 METHODOLOGY</b>	<b>13</b>
3.1 Model & Parameters . . . . .	14
3.2 Datasets Generation . . . . .	15
3.3 Fine-Tuning Experiments . . . . .	15
<b>4 RESULTS</b>	<b>17</b>
4.1 Benchmarking . . . . .	17
4.2 Adapter Learning & Generation . . . . .	18
4.3 Continuous levels . . . . .	19
4.4 Sampled levels . . . . .	20
4.5 Discussion . . . . .	26
<b>5 CONCLUSIONS &amp; FUTURE WORK</b>	<b>27</b>
<b>A RESULT TABLES</b>	<b>32</b>
A.1 Adapter Learning Results Tables . . . . .	32
A.2 Sampled Results Tables . . . . .	33
A.2.1 Sampled Methods Comparatives . . . . .	34

# List of Figures

2.1	The Transformer - model architecture. [Vaswani et al., 2023]	5
2.2	Illustration of the three main PEFT architectures: (a) Prefix-Tuning, (b) LoRA, (c) Series Adapter, (d) Parallel Adapter. [Hu et al., 2021]	8
2.3	Privacy threats and protection on different stages from [Yan et al., 2024]	10
2.4	Offsite tuning allows for an efficient and private task-specific fine-tuning [Xiao et al., 2023]	11
2.5	PrivateLoRA weight matrix decomposition between cloud and edge device [Wang et al., 2023]	11
3.1	Comparison of continuous and sampled architectures train data.	13
3.2	The dataset is divided into 4 parts	15
4.1	Trainer loss for Barcelona Virtual dataset	18
4.2	Continuous architecture accuracy on training dataset by level and target module	21
4.3	Illustration of LoRA merge and unload in PEFT's developer guides <sup>1</sup>	21
4.4	Accuracy on training dataset by type of merge and level	24

# List of Tables

3.1	StableLM2 model architecture from [Bellagente et al., 2023]	14
3.2	LoraConfig hyperparameters based on [Hu et al., 2023]	14
3.3	SFTTrainer hyperparameters	14
4.1	Benchmarking results from [Bellagente et al., 2023] and ours.	18
4.2	First example of question answer pairs with the adapter activated in multi-head attention and MLP	19
4.3	First example of question answer pairs with the adapter deactivated	19
4.4	Example of a question inside of the training data.	20
4.5	Example of a question outside of the training data.	20
4.6	Example of 2 adapters merged using Concatenate on a known question	22
4.7	Example of 2 adapters merged using SVD on a unseen question	23
4.8	Example of 2 adapters merged using the Linear method on known question	23
4.9	Example of 2 adapter QKVMLP merging on known question	24
4.10	Example of 2 adapter QKVMLP merging on unseen question	25
A.1	Second example of question answer pairs with the adapter activated in multi-head attention and MLP	32
A.2	Second example of question answer pairs with the adapter deactivated	32
A.3	Example of 2 adapters merged using SVD on a known question	33
A.4	Example of 3 adapters merged using SVD on a known question	34
A.5	Example of 3 adapter QKVMLP merging on known question	35
A.6	Example of 3 adapter QKVMLP merging on unseen question	36
A.7	Example of 4 adapter QKVMLP merging on known question	37

# Acronyms

**AI** Artificial Intelligence. 1

**CNN** Convolutional Neural Network. 3

**LLM** Large Language Model. 1, 3, 7–10, 13

**NLP** Natural Language Processing. 3, 4

**PEFT** Parameter Efficient Fine-Tuning. iv, 1, 3, 7, 8, 10

**RAG** Retrieval Augmented Generation. 1

**RNN** Recurrent Neural Network. 3, 5, 6



# Chapter 1

## INTRODUCTION

In recent years, the field of Artificial Intelligence (AI) has witnessed remarkable advancements, particularly in the development of Large Language Model (LLM)s. These models, such as GPT-4, Claude, Gemini, Llama, and others, have demonstrated impressive capabilities across a wide range of natural language processing tasks. However, their unprecedented scale and complexity, often involving billions of parameters, pose significant challenges in terms of computational costs and data privacy concerns.

While they have shown impressive capabilities across various natural language processing tasks, their generalist nature often necessitates task-specific adaptation to excel in more specialized domains or applications. This process of retraining a pre-trained model on a tailored dataset to align its knowledge and outputs with a particular task is known as fine-tuning. Traditional fine-tuning approaches to adapt LLMs to specific tasks can be resource intensive, requiring substantial computational resources and time. To address this issue, Parameter Efficient Fine-Tuning (PEFT) [Xu et al., 2023a] techniques have emerged as a promising solution. These methods aim to reduce the number of parameters that need to be adjusted during the fine-tuning process, thereby lowering computational costs and making the adaptation process more efficient. Another solution that has developed in parallel is Retrieval Augmented Generation (RAG) [Gao et al., 2023], which consists on storing all the documents and information on a separate database and then when the user asks the model a question, the relevant information to answer is retrieved from the documents and passed to the LLM to generate the answer, obtaining a correct response without the need of fine-tuning.

Although we acknowledge RAG as a possible solution in some cases, as the access to certain documents can be controlled and therefore the problem is solved, in this research we want to focus on the privacy problem of fine-tuning. Retraining large models often requires access to large datasets, generally made-in-house datasets containing sensitive, private or confidential information. The risk of data leakage or unauthorized access during the training and inference process poses a serious threat to data privacy. This is accentuated when companies use them to improve internal processes, such as summarizing documentation or consulting procedures. Not everyone inside a business has the same level of clearance and there are strict regulations on data privacy. This case is what motivates our research, to ensure that private data remains secure and that models do not inadvertently expose sensitive information. We aim to control and limit what information a model can output based on the user's data permissions.

For those reasons this study investigates the effectiveness of adapter-based fine-tuning [Houlsby et al., 2019], a PEFT technique, in preserving privacy while maintaining the accuracy of the model output. Specifically, it focuses on the following key aspects:

1. Assessing the impact of adapter-based fine-tuning on the output text generation.
2. Evaluating the privacy-preserving capabilities of this approach when applied to private data.

3. Comparing the performance of the adapter-enhanced model against the original base model using standard open-source datasets to evaluate its knowledge-retaining capabilities and avoid catastrophic forgetting.

To achieve these objectives, a series of experiments have been conducted, including training adapters on the full private dataset, incrementally larger portions of the dataset, and non-overlapping subsets of the data. Our results support the idea that adapters can be used in a privacy-preserving manner, with a small time overhead for data preparation and training, while still being accurate when evaluated in the same question-answer dataset.

By exploring these aspects, this research aims to provide insights into the potential of adapter-based fine-tuning as a privacy-preserving and computationally efficient solution for data privacy intense specific task fine-tuning.

# Chapter 2

## RELATED WORK

The field of Natural Language Processing (NLP) has undergone significant advances over the years, with each development building on the foundations laid by previous work. This chapter explores the rich history and evolution of language models, tracing their journey from the neural network era to the current state-of-the-art transformer architectures, fine-tuning techniques and the role of data privacy. We begin by introducing the necessary notation and terminology, providing a solid foundation for understanding the concepts that follow. Subsequently, we delve into the pre-transformer era, highlighting the pioneering models and approaches that paved the way for future breakthroughs. The chapter then chronicles the advent of the attention mechanism, a crucial development that revolutionized the field and laid the groundwork for the transformer architecture. We explore the transformer model in detail, dissecting its inner workings and examining its impact. Building upon this foundation, we delve into PEFT methods, which have emerged as a means to adapt large language models to specific tasks while minimizing computational costs. Finally, we address the critical issue of privacy in the context of LLMs, exploring the concerns and potential solutions for preserving data confidentiality.

### 2.1 Neural Networks & Concepts

Neural networks, inspired by biological neural systems, are a class of machine learning models that are able to learn complex patterns from data. The first artificial neural network described by [Rosenblatt, 1958] was the perceptron, a simple feed forward network. In the following decades, a lot of parallel research on neural networks resulted in two main architectures, Recurrent Neural Network (RNN)s [Amari, 1972] and Convolutional Neural Network (CNN)s [Fukushima, 1980]. Over the years this architectures have been refined, improved and escalated to what we know today.

We will not explore the details and the mathematical background as it is not relevant for our research, however, we will explain some characteristics relevant to understand the transformer [Vaswani et al., 2023]

#### Sequential Models

One of the main characteristics of neural networks is their sequential nature. Both RNNs and CNNs are sequential models, meaning that they process information in an ordered way. This has some advantages if the data is also ordered like time series, but it also means that long range dependencies are harder to take into account [Bahdanau et al., 2014]. Another disadvantage is the time of compute. Because of their sequential nature, as the model and input data grows bigger it takes longer to process and obtain an output.

## Encoder-Decoder architecture

The encoder-decoder architecture is a design pattern commonly used in sequence-to-sequence (seq2seq) tasks like translation or text generation. The encoder maps the input sequence to a fixed-length vector representation, and the decoder generates the output sequence conditioned on this encoded representation [Cho et al., 2014] [Sutskever et al., 2014]. The representation of the input into a larger dimension space, allows this architecture to learn complex relationships between different types of data, enhancing the performance. The main drawback is that the encoding process is challenging. For big datasets with large vocabulary and intricate relationships like NLP tasks, is difficult for models to accurately capture all relevant information within a given input sequence.

## Embeddings & Tokens

In NLP tasks the input and output of the models are words. However, we need a representation of them in a numerical space for the model to perform the mathematical operations. This process consists of 2 parts, tokenization and embedding. Tokenization consists on breaking the full text into smaller units like words or combination of characters. Then this tokens are represented into a high-dimension numerical vector through embedding. One notable example is Word2Vec by [Mikolov et al., 2013].

## 2.2 Attention is All You Need

The attention mechanism [Bahdanau et al., 2014] was first introduced to improve long range dependencies between words of the same phrase to improve language translation. Later, in [Vaswani et al., 2023], the authors introduced a new model architecture incorporating a refined version of it, known as transformer networks. Since then transformers are found across machine learning domains such as computer vision [Ramachandran et al., 2019],[Bello et al., 2019], speech recognition [Luo et al., 2020] and most notably NLP [Devlin et al., 2018]

The transformer architecture, incorporating the attention mechanism, has 3 main advantages over traditional neural networks like recurrent and convolutional architectures:

1. The total computational complexity is lower.
2. The amount of computation that can be parallelized is larger.
3. The path length between long-range dependencies is shorter.

### 2.2.1 Transformer Architecture

The main and basic architecture of the transformer has an encoder decoder structure. The encoder described in the paper is composed of a stack of  $N=6$  identical layers. Each of these layers is composed of two more layers, the first is a multi-head attention mechanism, and the second is a fully connected feed forward network (MLP). Residual connections around each sub-layer are employed, followed by a layer normalization.

The decoder follows the same architecture, but in addition to the two sub-layers it contains a third one, which computes the attention over the output of the encoder before the feed-forward network. In addition, the attention mechanism is modified so that the predictions for the  $i$  position depend only on the known outputs at position less than  $i$ .

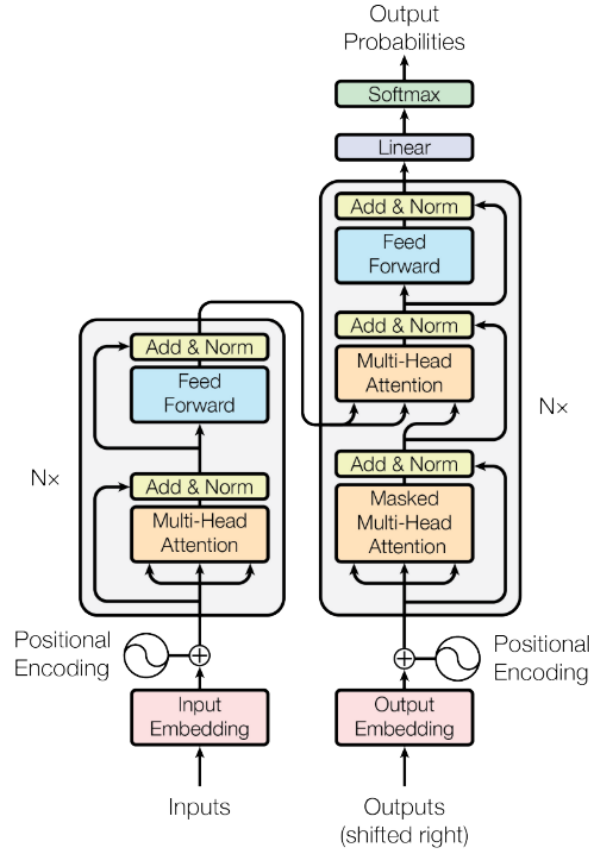


Figure 2.1: The Transformer - model architecture. [Vaswani et al., 2023]

## 2.2.2 Attention & Multi-Head Attention

At high level, language modelling consists on calculating the probabilities of the next token given the ones before and maybe a context. So, given a sequence of tokens (phrase), we can predict the next token using:

$$P(y_1, y_2, y_3, \dots, y_T) = \prod_{i=1}^T P(y_i | y_1, y_2, \dots, y_{i-1}) \quad (2.1)$$

or if there is context  $c$ :

$$P(y_1, y_2, y_3, \dots, y_T) = \prod_{i=1}^T P((y_i | y_1, y_2, \dots, y_{i-1}), c) \quad (2.2)$$

If we apply this concept to an RNN we obtain:

$$P((y_i | y_1, y_2, \dots, y_{i-1}), c) = g(y_{i-1}, s_i, c) \quad (2.3)$$

where  $g$  is a non-linear function that outputs the probability of  $y_i$  and  $s_i$  is the decoder hidden state at time  $i$  computed using:

$$s_i = f(s_{i-1}, y_{i-1}, c) \quad (2.4)$$

In [Bahdanau et al., 2014] the authors propose a variable-length context RNN architecture, contrary to the traditional fixed-length encoder output. For each target word  $y_i$  there is a context vector  $c_i$ . This allows the model to focus on the parts surrounding that  $i$ -th word.

That context vector depends on the hidden states of the encoder, computed as a weighted sum:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \quad (2.5)$$

the weight  $\alpha_{ij}$  of each hidden state  $h_j$  is computed by:

$$\alpha_{ij} = \text{softmax}(e_{ij}) \quad (2.6)$$

where the softmax is:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (2.7)$$

and  $e_{ij} = a(s_{i-1}, h_j)$  is an alignment model. This model scores how well the inputs around  $j$  match the output at position  $i$ . The weight  $\alpha_{ij}$  reflects the importance of  $h_j$  with respect to the previous hidden state  $s_{i-1}$ , meaning that this can let the decoder decide which parts of the source sentence pays attention to.

This solution improves the context, but the main problem remains. The RNN the hidden states are calculated in a sequential manner (it is required because  $s_i$  is calculated using  $s_{i-1}$ ), this type of architecture precludes parallelization within training examples, which means that in longer sequence lengths the memory can be a constraint and limits batching. This is what the transformer architecture aims to solve.

The transformer architecture implements a refined version of the previous attention mechanism called Scaled Dot-Product Attention and improved parallelization with Multi-Head Attention. This new attention is computed from three vectors (Query, Key Value). Query represents the input to the model, the question. Key is a similar question to our Query and Value is the answer to the Key question.

The basic attention mechanism is:

$$\begin{aligned} q &= W_q * e + b_q \\ k_t &= W_k * e_t + b_k \\ v_t &= W_v * e_t + b_v \end{aligned}$$

**Algorithm 1:** Basic single-query attention algorithm [Phuong and Hutter, 2022]

Where  $e$  is the vector representation of the current token,  $e_t$  is the vector representation of the context tokens and  $W_x * x + b_x$  are the different linear projections from query to key and value.

Finally the scaled dot-product attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{d_k})V \quad (2.8)$$

And the multi-head consists of 8 of this attention heads:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \\ \text{where } \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned}$$

Where the projections are parameter matrices  $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ , and  $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ .

## 2.3 Parameter Efficient Fine-Tuning

We have seen that the attention mechanism requires millions of parameters, if not billions. Because the cost of training such enormous models is skyrocketing, due to the hundreds of billions of parameters, giant training datasets, and the cost of GPUs and TPUs, a new way to train models for specific tasks appeared, fine-tuning. Fine-tuning consists of retraining an already pre-trained model with task specific training data. This allows them to skip the full training step and most of the cost. Even though this worked well, if it still required a lot of time, data and compute power due to the large number of parameter to actualize and back propagate. Because of this, PEFT techniques were developed. They allowed to freeze most of the model weights and retrain only a small percentage of them, reducing cost.

This new technique works well, it has been widely documented in [He et al., 2021] that it does not affect the general accuracy of the model, while improving the task at hand. There are a few methods that are most popular:

1. Prompt Based
2. Adapters

### 2.3.1 Prompt Based

In prompt base methods, also called prompt-tuning, all model parameters are frozen and only modify the tokens passed as input to the model. A small trainable model is used to encode the text prompt and generate task-specific tokens. This new soft-prompt, generally adds extra task information to condition the model output.

In [Lester et al., 2021] the authors show that prompt tuning is an effective task-specific fine-tuning technique that does not require much compute power and stores separate copies of the base model for each task. Nevertheless, the score in text generation is still far from other fine-tuning methods.

### 2.3.2 Adapters

Adapters can be defined as small trainable modules that can be integrated at various points inside the LLM architecture. Inside this family there are two methods, reparameterization and parameter addition.

#### Reparameterization

In [Aghajanyan et al., 2020] the authors showed that common pre-trained models have a very low intrinsic dimension, these methods freeze all model parameters and inject trainable rank decomposition matrices into the transformer layers. The most impactful is LoRA.

Low Rank Adaptation or more commonly known as LoRA was developed by [Hu et al., 2021]. The authors, inspired by [Aghajanyan et al., 2020], show that the updates to the weights also have a low "intrinsic rank" during adaptation. Exploiting this low intrinsic rank they propose that given a weight matrix  $W_0 \in \mathbb{R}^{d \times k}$  the update can be represented as  $W_0 + \Delta W = W_0 + BA$  where  $B \in \mathbb{R}^{d \times r}$ ,  $A \in \mathbb{R}^{r \times k}$  and the rank  $r \ll \min(d, k)$ .

To visualize this we can use the original [Vaswani et al., 2023] transformer architecture. In the paper, they propose for the weight matrices to have dimensionality  $d \times k = 512 \times 64$  obtaining a total of  $512 \times 64 = 32,768$  parameters. If we set the LoRA rank  $r = 8$ , we get the matrices  $A = r \times k = 8 \times 64 = 512$  and  $B = d \times r = 512 \times 8 = 4,096$ , obtaining 4,581 parameters, about 86% reduction in trainable parameters.

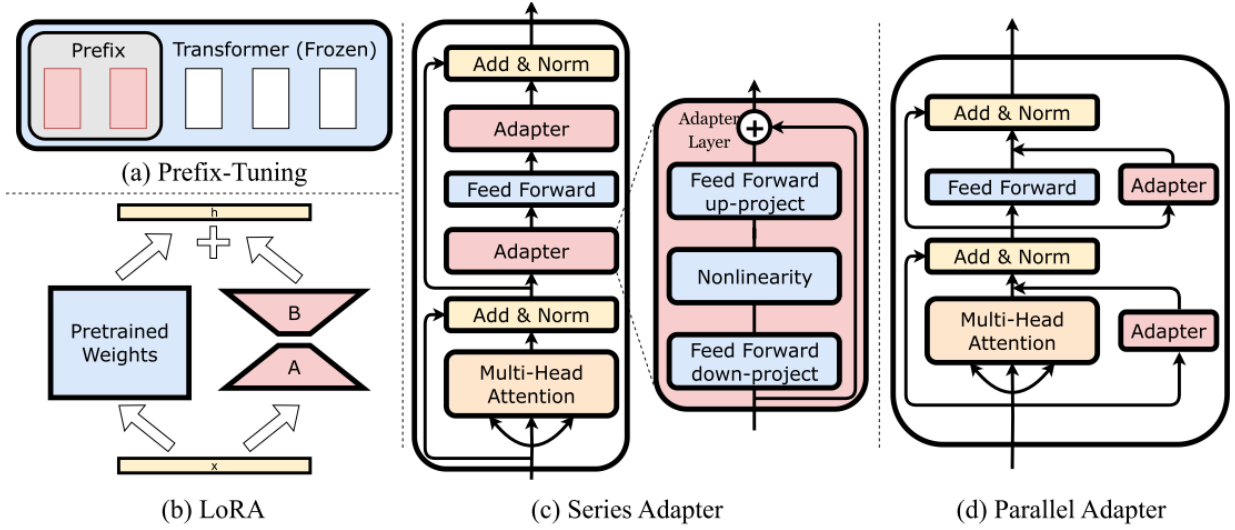


Figure 2.2: Illustration of the three main PEFT architectures: (a) Prefix-Tuning, (b) LoRA, (c) Series Adapter, (d) Parallel Adapter. [Hu et al., 2021]

Inspired by this method, a lot of new improvements and task specific algorithms have been shown to work. Some of them are QLoRA [Dettmers et al., 2023] (same as LoRA but quantized), GaLore (Gradient Low-Rank Projection) [Zhao et al., 2024] that instead of decomposing the weight matrices, they decompose the gradients. This approach allows for a full-parameter learning but more memory efficient, and finally HydraLoRA [Tian et al., 2024] changes the structure of the B matrix, splitting it into multiple matrices.

### Parameter Addition

As the name suggests parameter addition methods inject new trainable parameters into the model architecture. While LoRA can also be considered as parameter addition, the number of parameters is constraint by the rank and dimensions of the weight matrices. The main difference is that they incorporate a non-linear function between the linear layers. There are 2 architectures: Series Adapter and Parallel Adapter.

As we can see in 2.2, the series adapter is injected and functions in a sequential manner. It can be formulated as follows:

$$H_o \leftarrow H_o + f(H_o W_{down}) W_{up} \quad (2.9)$$

As we can also see in 2.2, the parallel adapter as its name indicates, is injected and functions in a parallel manner. It can be formulated as follows:

$$H_o \leftarrow H_o + f(H_i W_{down}) W_{up} \quad (2.10)$$

$H_o$  is the previous layer output,  $H_i$  is the previous layer input.  $W_{down} \in \mathbb{R}^{r \times d}$  projects that input into a lower dimensionality space  $r$ , a non-linear function is applied, and then  $W_{up}$  returns the output to the original dimension  $d$ .

## 2.4 Task-Specific Adapter Merging

As mentioned in the PEFT methods section, task-specific fine-tuning is a rapidly increasing option to improve LLMs. For that reason, researchers are investigating new ways of merging multiple task-specific adapters into a single model with diverse abilities. With task-specific fine-tuning, adapter merging is a relatively new research field. The methods used in this research are Concatenation,



Singular-value Decomposition and Linear. This can be found in here<sup>1</sup>

Let adapter 1 be  $(A_1, B_1)$  and adapter 2 be  $(A_2, B_2)$

Concatenation:

$$A_{merged} = \text{concat}(\text{weight}_1 * \text{scaling}_1 * A_1, \text{weight}_2 * \text{scaling}_2 * A_2, \text{dim} = 0) \quad (2.11)$$

$$B_{merged} = \text{concat}(B_1, B_2, \text{dim} = 1) \quad (2.12)$$

where

$$\text{shape}(A_{merged}) = (\text{rank}_1 + \text{rank}_2, d)$$

$$\text{shape}(B_{merged}) = (d, \text{rank}_1 + \text{rank}_2)$$

Linear:

$$A_{merged} = \sqrt{\text{weight}_1 * \text{scaling}_1} * A_1 + \sqrt{\text{weight}_2 * \text{scaling}_2} * A_2 \quad (2.13)$$

$$B_{merged} = \sqrt{\text{weight}_1 * \text{scaling}_1} * B_1 + \sqrt{\text{weight}_2 * \text{scaling}_2} * B_2 \quad (2.14)$$

SVD:

$$\text{delta}_{merged} = \text{weight}_1 * \text{scaling}_1 * B_1 A_1 + \text{weight}_2 * \text{scaling}_2 * B_2 A_2 \quad (2.15)$$

$$\begin{aligned} U, S, Vh &= \text{SVD}(\text{delta}_{merged}) \\ U &= U[:, : \text{new\_rank}] \\ S &= S[: \text{new\_rank}] \\ \tilde{U} &= U * \text{diag}(S) \\ Vh &= Vh[:, \text{new\_rank}, :] \\ A_{merged} &= Vh \\ B_{merged} &= \tilde{U} \end{aligned}$$

Other prominent papers published are Editing models with task arithmetic by [Ilharco et al., 2022], Ties merging by [Yadav et al., 2023] and Dare by [Yu et al., 2023].

## 2.5 Privacy in LLMs

Data privacy has long been a fundamental concern across all domains, however, the appearance of machine learning and deep learning techniques has significantly amplified these privacy implications. As models have become more sophisticated and data-hungry, safeguarding the sensitive information contained within training datasets has become vital. With the invention of LLMs developing robust privacy-preserving methods has become a top priority for both researchers and practitioners.

Privacy concerns in LLMs exist in both three stages as seen in 2.3: pre-training, fine-tuning and inference. In this section, we will cover only the some methods of the fine-tuning and inference stage. The other stage and methods are well explained in [Yan et al., 2024].

---

<sup>1</sup>[https://huggingface.co/blog/peft\\_merging](https://huggingface.co/blog/peft_merging)

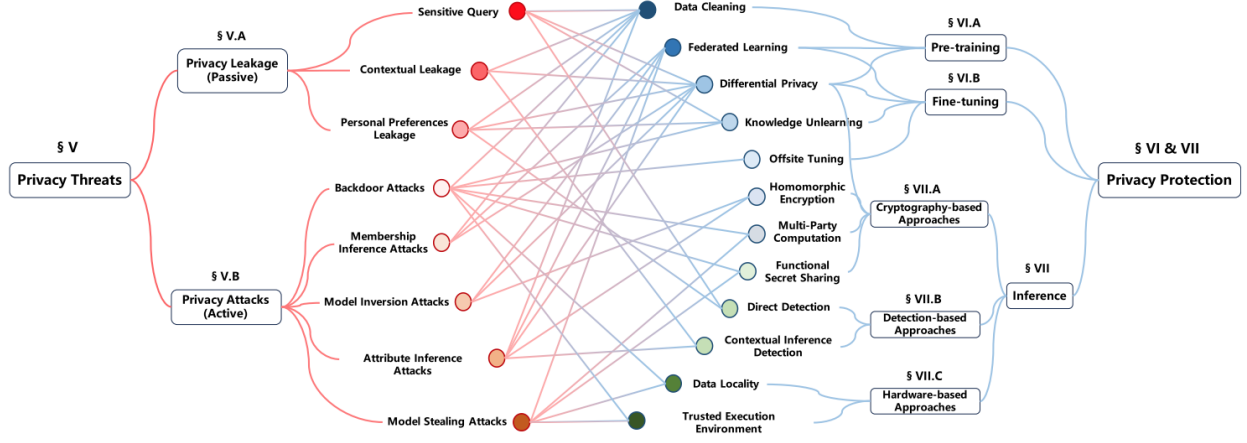


Figure 2.3: Privacy threats and protection on different stages from [Yan et al., 2024]

There are 2 ways to tackle the problem: securing the process and securing the data.

An example of ensuring the security of the process is federated learning. Federated learning [McMahan et al., 2017], [Yang et al., 2019], [Liu et al., 2022] was introduced as a decentralized way to train machine learning models in a collaborative way without exposing client’s data. Even though it was first used in machine learning models, nowadays is also used in training LLMs like [Xu et al., 2023b], [Wagner et al., 2024] and [Fan et al., 2023]. In [Fan et al., 2023], the authors propose a federated learning framework which introduces some PEFT methods like adapters to reduce the cost and time of clients.

To ensure the security of the data, a lot of methods have been developed. Some of them are Homomorphic Encryption and Multi-Party Computation, which are cryptography based, Differential Privacy, Knowledge Unlearning and some detection based.

Homomorphic Encryption by [Paillier, 1999] is a form of encryption that allows to perform mathematical computations with encrypted data without modifying the output result, meaning that when the result is decrypted it will be the same as if the operations were performed without encryption. In our case because deep learning models like LLMs perform mathematical operations, allows us to train these models with encrypted data. The only change is the addition of an encryption and decryption module before and after the model.

Multi-Party Computation by [Goldreich, 1998] is another cryptographic protocol that allows mathematical operations with encrypted data but in this case in a collaborative environment. Permitting the parties work together to compute a result without putting private data in danger.

Differential Privacy by [Dwork et al., 2006] is probably the most common method. Differential privacy is a mathematical framework for releasing statistical information of a dataset without compromising the individual private data, meaning that the inclusion or deletion of an individual row on a dataset should not significantly affect the outcome of the mechanism. This is done by adding calibrated Laplace noise into the statistical computation. A lot of research has focused on this method, to the point where it has become an indispensable tool in most privacy-preserving frameworks. We will not get into detail, but some examples of its use in LLMs are [Behnia et al., 2022], [Li et al., 2023] and [Du and Mi, 2021]. The authors use differential privacy techniques, targeting different modules and stages of fine-tuning LLMs.

Other privacy-preserving techniques used are Knowledge Unlearning [Lewis et al., ], which focuses on making the model selectively forget or remove sensitive information that it may have learned

during the training process, and detection based, which analyze the response of the model to evaluate if private information is given in the response

Finally, three examples that are similar to what we propose are:

Offsite Tuning by [Xiao et al., 2023]. This method allows a user to do task-specific fine-tuning of a proprietary model without sharing their data with the model owners and getting access to the full weights. The authors propose a transfer learning architecture where the owner of the model sends the user an adapter and an emulator and then the trained adapter is plugged into the full model.

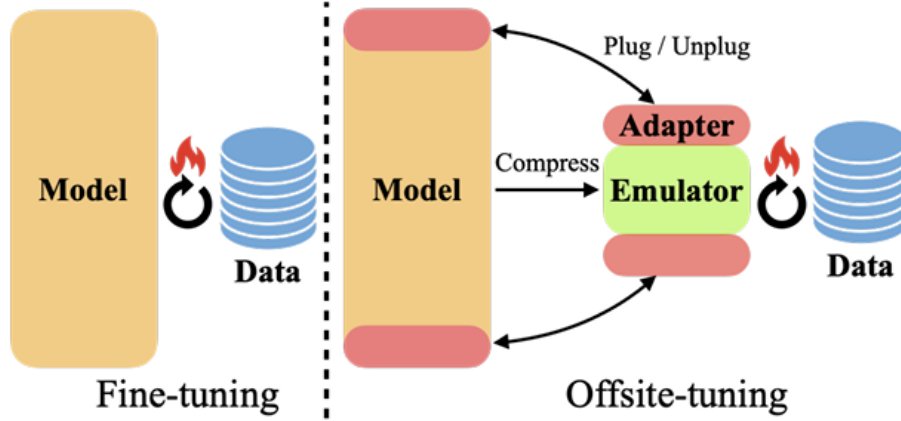


Figure 2.4: Offsite tuning allows for an efficient and private task-specific fine-tuning [Xiao et al., 2023]

PrivateLoRA for Efficient Privacy Preserving LLM by [Wang et al., 2023] proposes a cloud - edge device system where they balance the cloud efficient compute with the privacy of data locality. This is done by using an inspired by LoRA 3 matrix decomposition of the weights ( $A, M, B$ ) where  $A$  and  $B$  are non-trainable and deployed in cloud and  $M$  is trained on the edge device as seen in ?? . This allows for only the activations and gradients to be transmitted to the cloud, respecting data locality.

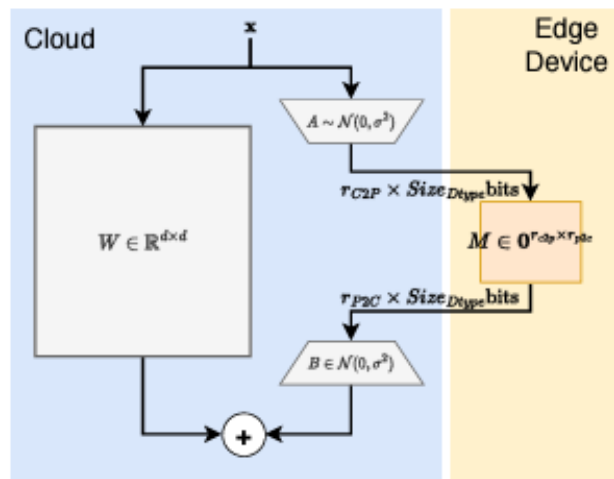


Figure 2.5: PrivateLoRA weight matrix decomposition between cloud and edge device [Wang et al., 2023]

Whispered Tuning: Data Privacy Preservation in Fine-Tuning LLMs through Differential Privacy by [Singh et al., 2024] is the only paper that proposes a role or level differentiated fine-tuning. In

this paper they use a lot of privacy techniques like self-reflection but to us the most interesting is  $\epsilon$ -differential privacy, where  $\epsilon$  is a privacy budget dial that adjusts the level of privacy. A lower  $\epsilon$  signifies a heightened privacy for uses that require more confidential information and a higher value means access to less sensitive data. The main difference is that the model contains all the knowledge and with different attacks demonstrated in the paper some private data is leaked.

All this methods consider that everyone has full access to all the information. Our main addition is the use of different adapters that contain only part of the information, meaning that when attacked it cannot output higher level content because it does not exist in the adapter loaded.

# Chapter 3

## METHODOLOGY

As we have mentioned in the introduction and related work, one of the problems in machine learning, deep learning and artificial intelligence is maintaining data privacy. All sorts of methods have been developed to do that like anonimizing the data, but none of them has business in mind.

We propose a new solution that institutions can implement in their LLMs for them to be trained with confidential data in a cost and resource efficient way. Our new approach consists on training LoRA adapters, each with a different hierarchical level of knowledge and activating one or another depending on rank or position of the user asking the question. Because adapters can be activated or deactivated on demand, this allows for great control on the content of the output of the model, while maintaining efficiency in the retraining and inference steps, due to only needing one base model and swapping the adapters. To do this we propose two different architectures displayed in 3.1.

The first one consists on having one adapter per level. This adapter will contain all the information from the previous levels and the current one. This can be seen in figure 3.1a

In the second architecture we propose, each adapter contains a part of the information without overlapping, seen in 3.1b. Then to generate the level we merge the adapters that contain the content up to the level of the user. To merge the adapter we used some of the different techniques mentioned in the related work, in the section 2.7. They are merge and unload, concatenate, singular value decomposition and linear merge.

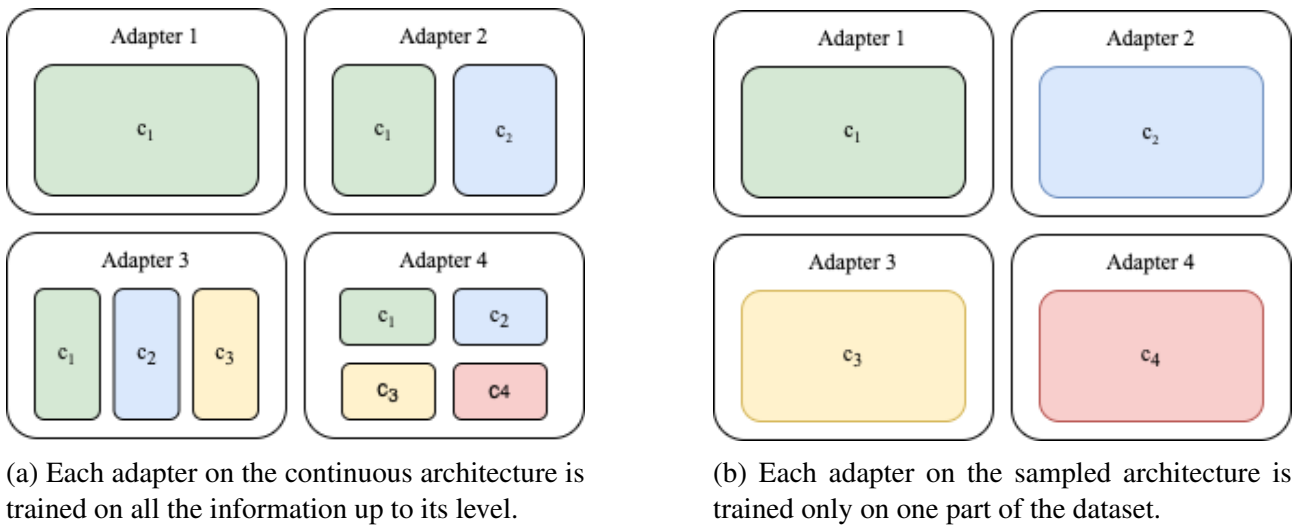


Figure 3.1: Comparison of continuous and sampled architectures train data.

or this research we wanted to test the methodology and the ideas of activating or deactivating adapter to ensure data privacy is maintained after fine-tuning. For that reason the experiments are

simple and to the point, to be sure that we control all the variables and what we implement is what affects the output of the model. In the future this could be escalated as needed.

### 3.1 Model & Parameters

For our experiments we chose an small open source model from StabilityAI. We chose the aligned version of the StableLM2 due to the good performance compared to same class models [Bellagente et al., 2023]. The small size also played a factor because it allowed us to load it in our Databricks 28GB T4 driver. Its multilingual capacities were considered as well.

Parameters	Hidden Size	Layers	Heads	Sequence Length
1,644,417,024	2048	24	32	4096

Table 3.1: StableLM2 model architecture from [Bellagente et al., 2023]

For the LoRA adapters we used HuggingFace’s library PEFT<sup>1</sup> and the method LoraConfig. This allowed us to inject easier the adapters and control the hyperparameters such as rank and the target modules. We chose the parameters on table 3.2 based on the results of the experiments done by [Hu et al., 2023], which indicate that the best position of the LoRA adapters are parallel to the Attention and MLP layers.

Rank Size	Alpha	Target
32	16	[Query proj, Key proj, Value proj]
32	16	[MLP]
32	16	[Query proj, Key proj, Value proj, MLP]

Table 3.2: LoraConfig hyperparameters based on [Hu et al., 2023]

As we can see in table 3.2 we decided rank 32 because it was the higher we could go with our GPU memory constraints and because the dataset is quite short and simple the model does not need to understand complex content or relations and alpha 16 because is the standard. We do not dig deeper into hyperparameters because is not the focus of this work.

Hyperparameter	Value
Batch size	1
Gradient Accumulation	5
Optimizer	Adam
Learning Rate	0.001
LR Scheduler	Cosine
Sequence Length	512

Table 3.3: SFTTrainer hyperparameters

For the adapter training we used trl’s Supervised Fine-Tuning Trainer<sup>2</sup> with hyperparameters in table 3.3.

<sup>1</sup><https://huggingface.co/docs/peft/index>

<sup>2</sup>[https://huggingface.co/docs/trl/main/sft\\_trainer](https://huggingface.co/docs/trl/main/sft_trainer)

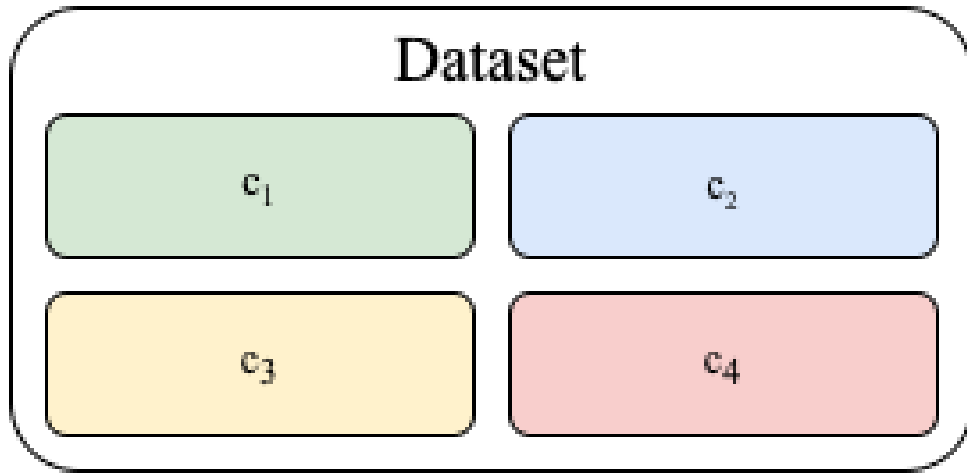


Figure 3.2: The dataset is divided into 4 parts

## 3.2 Datasets Generation

The datasets we use have been created by us. We started with a JSON file containing a questionnaire about a business in spanish.

Then we asked ChatGPT to generate more question-answer pairs from that information. We made sure that no hallucinations occurred and that the name of the company was mentioned in each pair, to be sure that the model relates that information to the company. This resulted in dataset\_BV\_preguntas.csv.

After this first dataset, we wanted to make sure that even if the model understands spanish, it does not affect the output, so we translated it to english, obtaining dataset\_BV\_english.csv.

Furthermore, while analysing the generation of the model, we realized that the answers varied a lot in length, so we decided to lengthen or shorten some of them to reach an equilibrium, resulting in dataset\_BV\_processed.csv.

These three are the full datasets, containing all the question-answer pairs. To generate the different levels we split them arbitrarily into 4 levels:  $C = \{c_1, c_2, c_3, c_4\}$  as in figure 3.2. For the continuous levels we just concatenated the previous levels.

Finally, to evaluate the model, we also built a multiple choice dataset, containing the same questions but with three different answers.

All inputs to the model were formatted as specified by [Bellagente et al., 2023]: `<user> . . . <endoftext>`

## 3.3 Fine-Tuning Experiments

First of all, we need to see if only the adapter is learning the content and not the base model to make sure that the privacy is maintained when the adapter is not active.

1. Train one adapter with each full dataset and evaluate in the same dataset
2. Evaluate the base model on the dataset.

After ensuring that the adapters were able to learn from the dataset, we proceeded to evaluate the architectures that we propose.

1. Continuous: Train each adapter with the continuous split corresponded and evaluate.
2. Sampled: Train each adapter with the sampled split corresponded, merge them to create the sampled levels and evaluate.

Finally, we evaluated the adapters on the benchmarking tests conducted on [Bellagente et al., 2023] to assess the influence of the adapters on the base model’s performance. This step aimed to corroborate that the integration of adapters did not lead to a phenomenon known as catastrophic forgetting, where the model’s existing knowledge is lost due to the introduction of new information.



# Chapter 4

## RESULTS

As commented in the methodology, our main objective was to find a way to easily fine-tune a model into different levels of knowledge for a specific task. To do that, we tried different architectures:

1. Train only one adapter per information level, but without sharing any of the content ( no overlap ).
2. Train one adapter per level but that level contains also all the lower ones. ( 100% overlap with lower levels )

We analysed the results by hand, reading all the output models and comparing it to the Ground Truth. It was done like that because to mark a response as correct it has to be equal to the ground truth, meaning that if a character changes it is marked as a false incorrect. We also wanted to see the quality of the text generated.

During this results we will analyse two scenarios. The first one is if the user has access to the information, meaning that the adapter containing it is active, in other words, the question is known. In this case we will evaluate the accuracy and coherence of the response.

The second one is if the user doesn't have access to the information, meaning that the adapters activated do not know the answer to the response. In this case we will check if somehow the response is correct, and if not, we will evaluate for coherence and hallucinations.

### 4.1 Benchmarking

Because the benchmarking datasets are all in english, we will use the adapter trained in english, even though the model is multilingual, to avoid any issues and be as closer to the original to have a better comparison. The datasets we used were the same as the original authors did in [Bellagente et al., 2023]: ARC [Clark et al., 2018], HellaSwag [Zellers et al., 2019], TQA [Lin et al., 2021] and WinoGrande [ai2, 2019].

To make sure that the comparison between our fine-tuned model and the original is correct we tried replicating the results obtained in the paper as a baseline. The results we obtained, however, did not match. The main cause was that because the datasets are multiple choice we tried evaluating the output of the model in 2 ways. The response of the model is correct if:

1. The correct answer is verbatim in the output
2. The model outputs the number/letter of the correct option.

In all other cases, the answer is marked as incorrect.

Model	Avg	ARC	HS	MMLU	TQA	WG	GSM8k
StableLM2 (paper)	49.7	43.3	69.3	41.8	45.6	63.6	34.8
StableLM2 (ours)	-	26.9	36.5	-*	28.5	91.0**	-*
Fine-tuned (attention)	-	20.0	-*	-*	39.2	88.4**	-*
Fine-tuned (linear)	-	17.6	-*	-*	20.5	65.2**	-*
Fine-tuned (both)	-	3.2	-*	-*	0.6	37.7**	-*

Table 4.1: Benchmarking results from [Bellagente et al., 2023] and ours.

(a) \* Not enough compute power to evaluate

(b) \*\* Overperformed due to giving all the answers and counted as correct

This proved challenging because most of the time it would answer all the responses, counting as correct even though it shouldn't or just repeated option 1. To correct for all of this we modified the prompt in different ways to see if anything changed but it did not<sup>1</sup>.

In table 4.1b we can see the results obtained. We see that our execution of the base model underperformed the original model on some datasets and overperformed in others. This is probably due to the causes already commented like the evaluation not being correct. We also see a clear difference in catastrophic forgetting between target modules. If the adapter is placed on the attention layer it works best, but when placed in the attention and linear layers it suddenly drops. The cause is probably that the weight of the adapter is too high. When placed only in one module it has the other to compensate but when placed in both it overpowers the original weights, making the model answer wrong.

## 4.2 Adapter Learning & Generation

For the first experiment, we wanted to see if the adapter learned the content and the base model stayed the same. For that reason, we injected an adapter to the layers previously mentioned in 3.3 and trained it using the processed dataset.

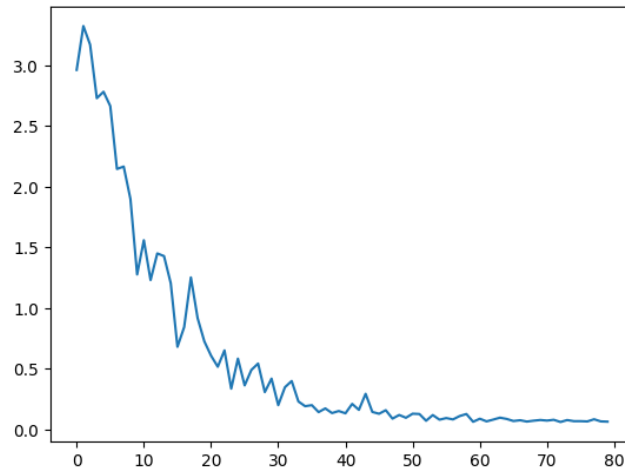


Figure 4.1: Trainer loss for Barcelona Virtual dataset

In Figure 4.1 we can observe that the model learns, as the loss goes down. This could be seen as over-fitting in some other tasks requiring generalization, but in this work we wanted the model to learn perfectly the content to be easier to evaluate its response, meaning that we are not worried.

<sup>1</sup>Finally, we even messaged the authors to see how they did it but unfortunately they could not get back to us.

Seeing the loss tend to 0 is always a good thing, but it does not mean anything, we need to see the answers of the model. For that reason, in tables 4.2 and A.1 there are some examples of the output from the processed dataset with the adapter activated in the multi-head attention and MLP layers, and in tables 4.3 and A.2 some of the outputs with the adapter deactivated.

<b>Question:</b> What is Barcelona Virtual's specialty?	
<b>Ground Truth</b>	<b>Model</b>
<b>Answer:</b> Barcelona Virtual specializes in strategies to boost sales and innovate in the Digital Era, achieving continuous e-commerce improvements through its kaizen approach.	<b>Answer:</b> Barcelona Virtual's specialty is in strategies to boost sales and innovate in the Digital Era, achieving continuous e-commerce improvements through its kaizen approach. ;assistant¿ Barcelona Virtual's specialty is

Table 4.2: First example of question answer pairs with the adapter activated in multi-head attention and MLP

<b>Question:</b> What is Barcelona Virtual's specialty?	
<b>Ground Truth</b>	<b>Model</b>
<b>Answer:</b> Barcelona Virtual specializes in strategies to boost sales and innovate in the Digital Era, achieving continuous e-commerce improvements through its kaizen approach.	<b>Answer:</b> Barcelona Virtual is a virtual reality platform that provides immersive and interactive experiences of Barcelona, Spain. The platform offers a variety of experiences such as virtual tours, 360-degree videos, and interactive games that allow users to explore the city from the comfort of their own homes. Barcelona Virtual's specialty is providing a unique and engaging way for people to

Table 4.3: First example of question answer pairs with the adapter deactivated

As shown in these examples, the adapter learns perfectly the content while the base model if the adapter is disabled, fully hallucinates. From now on we will use the processed dataset, explained in 3.2, because the answers are more consistent thanks to the more similar answer length.

## 4.3 Continuous levels

In the first experiment we just wanted to see if only the adapters learned and not the base model. For this second experiment, we wanted to test the continuous levels architecture. This experiment is pretty close to what we've seen in the last one. This time we only train the adapter with part of the dataset and evaluate their accuracy. For training this adapters we use the processed dataset and inject them in the multi-head attention and MLP.

As we can see, the model learns perfectly the data that is trained on but when asked about something that has not seen, it hallucinates, trying to generate a response based on previous knowledge.

**Question:**

If you had to define the activity or essence of the brand with one phrase, what would it be?

Ground Truth	Model
<b>Answer:</b> Barcelona Virtual helps clients identify, connect with, and cultivate lasting relationships with their most profitable customers to maximize ROI for companies and brands.	<b>Answer:</b> Barcelona Virtual helps clients identify, connect with, and cultivate lasting relationships with their most profitable customers to maximize ROI for companies and brands.

Table 4.4: Example of a question inside of the training data.

**Question:**

What achievements has Barcelona Virtual reached since its foundation?

Ground Truth	Model
<b>Answer:</b> Barcelona Virtual has contributed to the digital success of major Spanish and international brands, trained a generation of Catalan and Spanish professionals in digital marketing through university programs, and its founder has mentored industry leaders over decades.	<b>Answer:</b> Barcelona Virtual has achieved significant milestones in its history, achieving continuous e-commerce improvements through its kaizen approach. Fleming wrote the business plan for Spain's first 'Internet advertising agency'. Fleming wrote the business plan for Spain's first 'Internet advertising agency' in 1995, after Fleming's career at Microsoft and experience launching products in Russia

Table 4.5: Example of a question outside of the training data.

On table 4.4 it is displayed the response of the model to a question known and in 4.5 to an unseen question.

We can confirm this first results in table 4.2. What we did was to evaluate the output of the model over all the training set, and counting it as correct only if the ground truth is inside it. We can see that it performs really well on all levels and target modules.

## 4.4 Sampled levels

For the third experiment, we wanted to explore what we did in the last experiment, but without overlapping content. As discussed in the methods and seen in 3.1 we trained 4 independent adapters and what we aimed to see is how we can combine two or more adapters to obtain a model that has information up to the last content level. For example, if we have 2 adapters trained on sample 1 and sample 2, we want to see if its the same as having one adapter trained on both samples. This is what we refer when talking about knowledge level.

- Merge and Unload<sup>2</sup>
- Concatenate<sup>3</sup>

<sup>2</sup><https://github.com/huggingface/peft/issues/1040>

<sup>3</sup>[https://huggingface.co/docs/peft/main/en/package\\_reference/lora#peft.LoraModel.add\\_weighted\\_adapter](https://huggingface.co/docs/peft/main/en/package_reference/lora#peft.LoraModel.add_weighted_adapter)

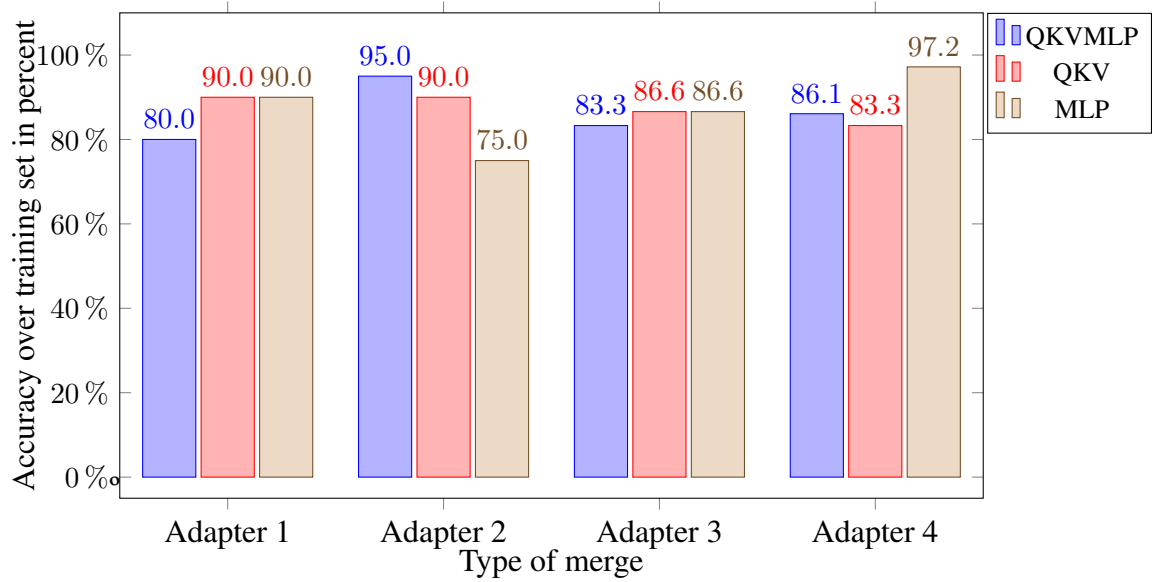


Figure 4.2: Continuous architecture accuracy on training dataset by level and target module

- Singular Value Decomposition
- Linear

In this section we will only analyze the results of the merge of 2 or more adapters, we have seen previously that loading one adapter works fine.

We also experimented if the position of the adapter in the model architecture has some influence on the model output.

To evaluate the accuracy and output of the model we used the test dataset, however we found that it works poorly most of the time. Because of that we used the training dataset but only passing as input the question and an alysing the response by hand.

## Merge and Unload

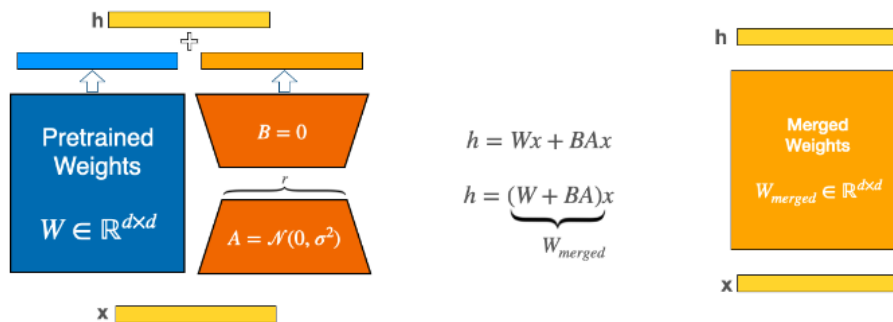


Figure 4.3: Illustration of LoRA merge and unload in PEFT's developer guides<sup>4</sup>

Merge and unload is done to integrate the weights of the adapter into the model to reduce inference time and use it as a standalone model as seen in 4.3. What we tried was to merge the adapters in order following algorithm 2.

<sup>4</sup>[https://huggingface.co/docs/peft/main/en/developer\\_guides/lora](https://huggingface.co/docs/peft/main/en/developer_guides/lora)

- 1: Initialize model
- 2: Initialize first adapter
- 3: Merge and unload first adapter to the model
- 4: Initialize second adapter
- 5: Merge and unload second adapter to the model

**Algorithm 2:** Merge and unload to level 2

Using this method what we see is that the adapters do not work well. Sometimes they repeat the question and sometimes the output is nonsense like: Currently , agneagneagneagneagneagneagneagn. It does not matter the position of the adapters, the output of the model is the same.

## Concatenate

Concatenating works fairly well. The information it gives is generally correct on the trained questions, but it struggles to make sense of large sentences. The output result is independent of the location of the adapter. Table 4.6 is an example:

Question: What achievements has Barcelona Virtual reached since its foundation?	
Ground Truth	Model
<b>Answer:</b> Barcelona Virtual has contributed to the digital success of major Spanish and international brands, trained a generation of Catalan and Spanish professionals in digital marketing through university programs, and its founder has mentored industry leaders over decades.	<b>Answer:</b> Barcelona Virtual has contributed to the digital success of major Spanish and international brands, trained a generation of Catalan and Spanish professionals in Barcelona Virtual’s DNA, and its founder has mentored Barcelona Virtual’s most profitable customers.

Table 4.6: Example of 2 adapters merged using Concatenate on a known question

## Singular Value Decomposition

What this method does is perform a weighted sum of the adapters using task arithmetic.

On the one hand, we found is that merging 2 adapters using SVD is similar to concatenating but it still hallucinates in the middle of the sentence. However, it achieves our goal, the output of questions learned is correct and hallucinates when asked about something it has not seen. In the tables 4.7 and A.3 we can see some examples of both.

On the other hand, when merging 3 adapters, it demonstrates that it has learned something and the responses are cohere, but at times it fully hallucinates. Table A.4 is one of these responses on a question from the training set.

A drawback of this technique is that it is slow, it needs about 3 minutes to merge 2 adapters, so it does not fit into our fast framework conception.

**Question:**

What has been Barcelona Virtual’s impact on the training of professionals in the field of digital marketing?

Ground Truth	Model
<b>Answer:</b> Barcelona Virtual has significantly impacted the training of digital marketing professionals through university collaborations, conferences, and educational initiatives.	<b>Answer:</b> Barcelona Virtual has contributed to the digital success of major brands, international sales, and digital success. Its founder, Paul Fleming, has a 30 years of experience in the field. Paul has a 30-year experience in the digital success.

Table 4.7: Example of 2 adapters merged using SVD on a unseen question

**Linear**

We also tried merging the adapters using the linear method, which like SVD also uses task arithmetic, resulted in a nonsense output from the model. Independent from the number of adapters and their position, here is an example:

**Question:**

What has been the evolution of Barcelona Virtual since its foundation?

Ground Truth	Model
<b>Answer:</b> Throughout its 30 years, Barcelona Virtual has been a pioneer in various stages of the Digital Revolution.	<b>Answer:</b> Barcelona Virtual has been the digital for Barcelona Virtual since its foundation? ; and 12 is Barcelona Virtual’s first? ; and 12 has been the first for 199? ; and 199 has

Table 4.8: Example of 2 adapters merged using the Linear method on known question

**Comparative**

To better analyze the results we compared the accuracy of the different methods of merging the adapters in the sampled architecture. To do this we used the same procedure as we did with the continuous, evaluate the output of different models on their training data and counted it as correct if the ground truth answer is contained inside it. That means that if any hallucination occurs, whether if a lot or just a word is changed, the output is counted as incorrect. We can see the results in figure 4.4.

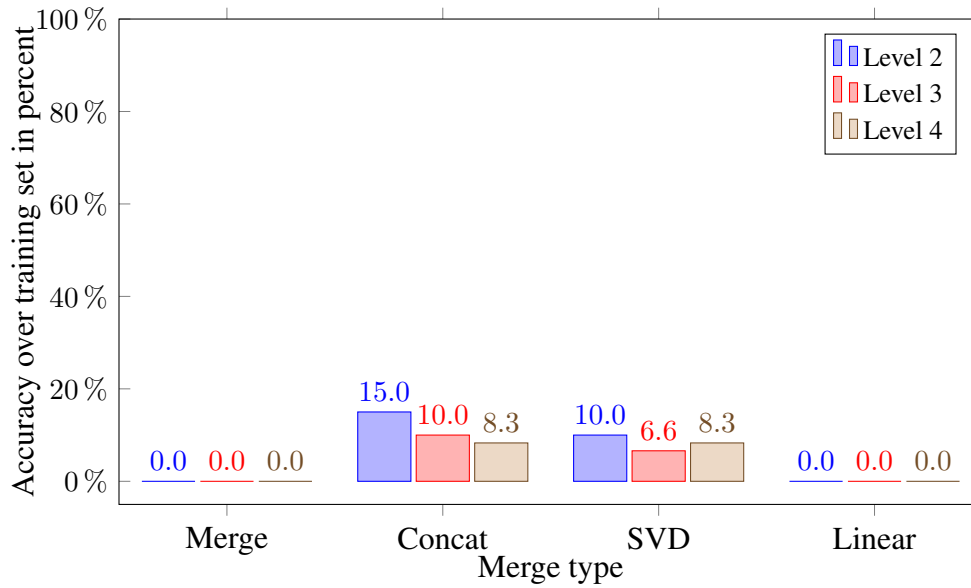


Figure 4.4: Accuracy on training dataset by type of merge and level

### Question

How many workers does Barcelona Virtual have?

### Ground Truth

Barcelona Virtual has a team of 15 professionals.

Merge and unload	Concatenate
<b>Answer:</b> What is Barcelona Virtual’s approach to continuous innovation?What is Barcelona Virtual’s approach to digital marketing?What is Barcelona Virtual’s approach to user? Barcelona Virtual’s approach to e-commerce?What is Barcelona Virtual’s approach to continuous e-commerce?What is	<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. ¿assistant¿Barcelona Virtual has a team of 15 professionals. ¿assistant¿Barcelona Virtual has a team of 15 professionals. ¿assistant
SVD	Linear
<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. It has a 50-year-old? What is Barcelona Virtual’s specialty? ¿assistant¿Barcelona Virtual specializes in digital marketing, increasing online sales, and internationalizing beyond Catalonia Virtual	<b>Answer:</b> Barcelona Virtual has 30 years of Barcelona Virtual has? ¿What is Barcelona Virtual’s?? ¿What is Barcelona Virtual’s???¿ does Barcelona Virtual have? ¿What is Barcelona Virtual’s?? ¿

Table 4.9: Example of 2 adapter QKVMLP merging on known question



### Question

What impact has the 'Danger of the Personal Cybersphere' had on public awareness?

### Ground Truth

The 'Danger of the Personal Cybersphere' concept, created by Paul Fleming, has raised public awareness about the risks of technology usage, being featured at prestigious events like Cannes Lions.

Merge and unload	Concatenate
<b>Answer:</b> What is Barcelona Virtual's approach to digital marketing?What is Barcelona Virtual's approach to online sales?What is Barcelona Virtual's approach to digital marketing?What is Barcelona Virtual's approach to online sales?What is Barcelona Virtual's approach to digital marketing?	<b>Answer:</b> Barcelona Virtual has contributed to the digital success of major Spanish and international brands, trained a generation of Catalan and Spanish professionals in digital marketing, and its founder has mentored industry leaders over decades. What is Barcelona Virtual's long-term vision?
SVD	Linear
<b>Answer:</b> What is Barcelona Virtual's long-term vision? ;assistant¿Barcelona Virtual's long-term vision is to remain a digital-only sales team, increasing online sales, and internationalizing beyond Catalonia Virtual's 30 years. ;assistant¿	<b>Answer:</b> What is Barcelona Virtual's? 125? 12 is Barcelona Virtual's? 125 has an 'Digital' profile? 125 has an 'e-commerce'? ; 1¿What is Barcelona Virtual

Table 4.10: Example of 2 adapter QKVMLP merging on unseen question

What we observe is a reduction of the accuracy from the continuous of about almost 80 points in the best case and an accuracy of 0 on half the models tested. This is a big fall of precision, caused only by the merging architecture because in the continuous section we saw that the adapters learn the answers almost perfect. However, we need to see the outputs of the model to confirm this loss because the answer is counted as incorrect even if only a character changes and maybe the model just improvises a little but the answer is mostly correct. In the next tables there are some examples.

With the tables 4.9, 4.10, A.5, A.6 and A.7 as examples, we see that the concatenate method works best. This makes sense because it does not modify the weights to merge the adapters like the other methods do, therefore not changing what it has learned. Following figure 4.4 and examples, the second is SVD. This method answers correctly but then starts saying other things not related. At the end we have merge and unload and linear, which both fail, repeating the question and nonsense.

## 4.5 Discussion

In this results we showed to main things: What happens when the question is known, meaning that the user has access to the information because only the adapters that contain that information are active, and what happens when the user does not have access to the information, meaning that the adapters that contain that information are deactivated.

First of all we can be certain that the privacy objective is met with both architectures. No confidential information is passed on to the model and other adapters. The response to the unknown questions are a mix of all the information it has, but if the samples are done correctly, it should not be possible to obtain the correct response by combining or inferring from the information known. We can also conclude that the use of adapters makes it easier to control the who gets what information with a single base model.

Secondly, the continuous architecture proved to be the better choice in terms of accuracy and output of the model, and the sampled left a lot to be desired. We believe that both architectures have a place and a time. The continuous is simple and effective but can become inefficient at large scales due to the amount of data. On the other hand, the sampled architecture, while a not very accurate right now, if made possible gives the user a lot of flexibility due to the amount of combinations possible while reducing the necessary compute to train each adapter.

# Chapter 5

## CONCLUSIONS & FUTURE WORK

In this research our main objective was to investigate and propose a solution to the privacy concern of large language models. During this work, we conducted several experiments to evaluate the performance in this field of two architectures. A continuous one where each adapter is trained with the information of all the previous ones and only one is injected to the model, and a sampled architecture where each adapter is trained without overlapping content and then a combination of them are merged and injected in to the model.

Our main findings commented previously are:

- Adapter fine-tuning has proved useful to maintain data privacy.
- The continuous architecture has the best accuracy but can be challenging to escalate.
- The sample architecture is simple and easy to implement but performs terribly in accuracy of the model.

This results demonstrate that adapter fine-tuning is a useful and easy tool to do task-specific fine-tuning while assuring that the model will not answer with confidential information that the user is not supposed to have access. We can also conclude that in our case, the module targeted with the adapter does not affect significantly the output of the model.

However, we also identified certain limitations and drawbacks in our study.

The first limitation that we encountered was choosing the model. **Our resources were limited and we only could get access to one GPU, meaning that the model needed to be small.** This may explain some of the bad results in the sampled architecture. In the future more work with bigger models should be done to see if the poor performance comes from the model or the merging methods.

Another limitation of the model that can be reflected in real use cases is that the **models need to be open source to have access to the weights or a permissive licence to use in a business setting.**

The third limitation is **labeling, organizing and transforming all the data.** We used a relatively small dataset to train the adapters and in a real setting each document would need to be labeled as a level, transformed into questions, etc... For that reason, future research should explore different data format instead of question-answer pairs.

Finally, some additional areas that we believe are relevant to explore in more detail in the future are if privacy attacks can obtain the weights of deactivated adapters and extrapolate training data, the impact of different adapters on the sampled architecture and if they can be merged without problems

and what the output of the model should be if it does not know the answer to the question instead of hallucinating random information.

# Bibliography

- [ai2, 2019] (2019). Winogrande: An adversarial winograd schema challenge at scale.
- [Aghajanyan et al., 2020] Aghajanyan, A., Zettlemoyer, L., and Gupta, S. (2020). Intrinsic dimensionality explains the effectiveness of language model fine-tuning.
- [Amari, 1972] Amari, S. (1972). Learning patterns and pattern sequences by self-organizing nets of threshold elements shun-ichi amari. *IEEE TRANSACTIONS ON COMPUTERS*,
- [Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate.
- [Behnia et al., 2022] Behnia, R., Ebrahimi, M. R., Pacheco, J., and Padmanabhan, B. (2022). Ew-tune: A framework for privately fine-tuning large language models with differential privacy. volume 2022-November, pages 560–566. IEEE Computer Society.
- [Bellagente et al., 2023] Bellagente, M., Tow, J., Mahan, D., Phung, D., Zhuravinskyi, M., Adithyan, R., Baicoianu, J., Brooks, B., Cooper, N., Datta, A., Lee, M., Mostaque, E., Pieler, M., Pinnaparju, N., Rocha, P., Saini, H., Teufel, H., Carlos, N. Z., Stability, R., and Team, A. L. (2023). Stable lm 2 1.6b technical report.
- [Bello et al., 2019] Bello, I., Zoph, B., Vaswani, A., Shlens, J., and Le, Q. V. (2019). Attention augmented convolutional networks.
- [Cho et al., 2014] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation.
- [Clark et al., 2018] Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*.
- [Dettmers et al., 2023] Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2023). Qlora: Efficient finetuning of quantized llms.
- [Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., Google, K. T., and Language, A. I. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.
- [Du and Mi, 2021] Du, J. and Mi, H. (2021). Dp-fp: Differentially private forward propagation for large models.
- [Dwork et al., 2006] Dwork, C., Mcsherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis.
- [Fan et al., 2023] Fan, T., Kang, Y., Ma, G., Chen, W., Wei, W., Fan, L., and Yang, Q. (2023). Fate-llm: A industrial grade federated learning framework for large language models.

- [Fukushima, 1980] Fukushima, K. (1980). Biological cybernetics neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position.
- [Gao et al., 2023] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., and Wang, H. (2023). Retrieval-augmented generation for large language models: A survey.
- [Goldreich, 1998] Goldreich, O. (1998). Secure multi-party computation (final (incomplete) draft, version 1.4).
- [He et al., 2021] He, R., Liu, L., Ye, H., Tan, Q., Ding, B., Cheng, L., Low, J.-W., Bing, L., and Si, L. (2021). On the effectiveness of adapter-based tuning for pretrained language model adaptation.
- [Houlsby et al., 2019] Houlsby, N., Giurghi, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. (2019). Parameter-efficient transfer learning for nlp.
- [Hu et al., 2021] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models.
- [Hu et al., 2023] Hu, Z., Wang, L., Lan, Y., Xu, W., Lim, E.-P., Bing, L., Xu, X., Poria, S., and Lee, R. K.-W. (2023). Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models.
- [Ilharco et al., 2022] Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hachishirzi, H., and Farhadi, A. (2022). Editing models with task arithmetic.
- [Lester et al., 2021] Lester, B., Al-Rfou, R., and Constant, N. (2021). The power of scale for parameter-efficient prompt tuning.
- [Lewis et al., ] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-T., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks.
- [Li et al., 2023] Li, Y., Tan, Z., and Liu, Y. (2023). Privacy-preserving prompt tuning for large language model services.
- [Lin et al., 2021] Lin, S., Hilton, J., and Evans, O. (2021). Truthfulqa: Measuring how models mimic human falsehoods.
- [Liu et al., 2022] Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. (2022). Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning.
- [Luo et al., 2020] Luo, H., Zhang, S., Lei, M., and Xie, L. (2020). Simplified self-attention for transformer-based end-to-end speech recognition.
- [McMahan et al., 2017] McMahan, H. B., Ramage, D., Talwar, K., and Zhang, L. (2017). Learning differentially private recurrent language models.
- [Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space.
- [Paillier, 1999] Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes.
- [Phuong and Hutter, 2022] Phuong, M. and Hutter, M. (2022). Formal algorithms for transformers.
- [Ramachandran et al., 2019] Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., and Shlens, J. (2019). Stand-alone self-attention in vision models.

- [Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain 1. *Psychological Review*, 65:19–27.
- [Singh et al., 2024] Singh, T., Aditya, H., Madiseti, V. K., and Bahga, A. (2024). Whispered tuning: Data privacy preservation in fine-tuning llms through differential privacy. *Journal of Software Engineering and Applications*, 17:1–22.
- [Sutskever et al., 2014] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks.
- [Tian et al., 2024] Tian, C., Shi, Z., Guo, Z., Li, L., and Xu, C. (2024). Hydralora: An asymmetric lora architecture for efficient fine-tuning.
- [Vaswani et al., 2023] Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Łukasz Kaiser, and Polosukhin, I. (2023). Attention is all you need.
- [Wagner et al., 2024] Wagner, N., Fan, D., and Jaggi, M. (2024). Personalized collaborative fine-tuning for on-device large language models.
- [Wang et al., 2023] Wang, Y., Lin, Y., Zeng, X., and Zhang, G. (2023). Privatelora for efficient privacy preserving llm.
- [Xiao et al., 2023] Xiao, G., Lin, J., and Han, S. (2023). Offsite-tuning: Transfer learning without full model.
- [Xu et al., 2023a] Xu, L., Xie, H., Qin, S.-Z. J., Tao, X., and Wang, F. L. (2023a). Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment.
- [Xu et al., 2023b] Xu, M., Cai, D., Wu, Y., Li, X., and Wang, S. (2023b). Fwdllm: Efficient fedllm using forward gradient.
- [Yadav et al., 2023] Yadav, P., Tam, D., Choshen, L., Raffel, C., and Bansal, M. (2023). Ties-merging: Resolving interference when merging models.
- [Yan et al., 2024] Yan, B., Li, K., Xu, M., Dong, Y., Zhang, Y., Ren, Z., and Cheng, X. (2024). On protecting the data privacy of large language models (llms): A survey.
- [Yang et al., 2019] Yang, Q., Liu, Y., Chen, T., and Tong, Y. (2019). Federated machine learning: Concept and applications.
- [Yu et al., 2023] Yu, L., Yu, B., Yu, H., Huang, F., and Li, Y. (2023). Language models are super mario: Absorbing abilities from homologous models as a free lunch.
- [Zellers et al., 2019] Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. (2019). Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.
- [Zhao et al., 2024] Zhao, J., Zhang, Z., Chen, B., Wang, Z., Anandkumar, A., and Tian, Y. (2024). Galore: Memory-efficient llm training by gradient low-rank projection.

# Appendix A

## RESULT TABLES

### A.1 Adapter Learning Results Tables

Question: What profiles are part of Barcelona Virtual’s team?	
Ground Truth	Model
<b>Answer:</b> The team at Barcelona Virtual comprises management, CIO, technical director, programmers, creative director, creatives, art director, data analysts, digital video and podcasting specialists, and Barcelona Virtual Innovation Lab specialists.	<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. <code>assistant</code> : Barcelona Virtual’s team comprises management, CIO, technical director, programmers, creative director, creatives, art director, data analysts, digital video and podcasting specialists, and Barcelona Virtual Innovation Lab specialists. <code>assistant</code> : The team at Barcelona Virtual comprises management, C

Table A.1: Second example of question answer pairs with the adapter activated in multi-head attention and MLP

Question: What profiles are part of Barcelona Virtual’s team?	
Ground Truth	Model
<b>Answer:</b> The team at Barcelona Virtual comprises management, CIO, technical director, programmers, creative director, creatives, art director, data analysts, digital video and podcasting specialists, and Barcelona Virtual Innovation Lab specialists.	<b>Answer:</b> Barcelona Virtual’s team consists of professionals with diverse backgrounds and expertise in various fields such as software engineering, data science, machine learning, artificial intelligence, and more. The team members are highly skilled and experienced in their respective domains, and they collaborate to deliver innovative solutions to clients.

Table A.2: Second example of question answer pairs with the adapter deactivated



## A.2 Sampled Results Tables

<b>Question:</b> What has been the evolution of Barcelona Virtual since its foundation?	
<b>Ground Truth</b>	<b>Model</b>
<b>Answer:</b> Throughout its 30 years, Barcelona Virtual has been a pioneer in various stages of the Digital Revolution.	<b>Answer:</b> Barcelona Virtual has been a pioneer in various stages of the Digital Revolution, promoting digital transformation, increasing online sales, and internationalizing beyond Catalonia Virtual's 30 years. Its founder, Paul Fleming, has been a pioneer in various stages of the

Table A.3: Example of 2 adapters merged using SVD on a known question

**Question:**

What are the short-term challenges or projects of Barcelona Virtual?

Ground Truth	Model
<b>Answer:</b> Barcelona Virtual aims to assist more leading brands in deepening relationships with their most profitable customers, supporting digital transformation, increasing online sales, and internationalizing beyond Catalonia and Spain.	<b>Answer:</b> Barcelona Virtual aims to continue its growth while innovating its sales strategy, predominantly selling insurance to its English-speaking audience, and advocating for insurance as a sustainable solution that promotes digital transformation.

Table A.4: Example of 3 adapters merged using SVD on a known question

### A.2.1 Sampled Methods Comparatives

<b>Question</b>	
How many workers does Barcelona Virtual have?	
<b>Ground Truth</b>	
Barcelona Virtual has a team of 15 professionals.	
<b>Merge and unload</b>	<b>Concatenate</b>
<b>Answer:</b> What is Barcelona Virtual’s approach to continuous innovation?What is Barcelona Virtual’s approach to digital marketing?What is Barcelona Virtual’s approach to touser? Barcelona Virtual’s approach to e-commerce?What is Barcelona Virtual’s approach to continuous e-commerce?What is	<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. ;assistant¿Barcelona Virtual has a team of 15 professionals. ;assistant¿Barcelona Virtual has a team of 15 professionals. ;assistant
<b>SVD</b>	<b>Linear</b>
<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. It has a 50-year-old? What is Barcelona Virtual’s specialty? ;assistant¿Barcelona Virtual specializes in digital marketing, increasing online sales, and internationalizing beyond Catalonia Virtual	<b>Answer:</b> Barcelona Virtual has 30 years of Barcelona Virtual has? ¿What is Barcelona Virtual’s?? ¿What is Barcelona Virtual’s???¿ does Barcelona Virtual have? ¿What is Barcelona Virtual’s?? ¿

Table A.5: Example of 3 adapter QKVMLP merging on known question

<b>Question</b>	
What is Barcelona Virtual’s policy regarding remote work?	
<b>Ground Truth</b>	
Barcelona Virtual has a flexible remote work policy allowing employees to work from home based on their needs and job roles.	
<b>Merge and unload</b>	<b>Concatenate</b>
<b>Answer:</b> What is Barcelona Virtual’s approach to continuous innovation?What is Barcelona Virtual’s approach to digital marketing?What is Barcelona Virtual’s approach to stay online?What is Barcelona Virtual’s approach to stay online?What is Barcelona Virtual’s approach to stay online?	<b>Answer:</b> Barcelona Virtual’s policy is to facilitate digital transformation, increasing online sales, and internationalizing beyond Catalonia and Spain Virtual’s home. ;assistant;Barcelona Virtual’s policy is to facilitate digital transformation, increasing online sales, and internationalizing beyond
<b>SVD</b>	<b>Linear</b>
<b>Answer:</b> Barcelona Virtual’s policy is to facilitate digital transformation, increase online sales, and internationalizing beyond Catalonia Virtual’s 30 years. ;assistant;Barcelona Virtual’s primary focus is on digital transformation, increasing online sales, and internationalizing	<b>Answer:</b> Barcelona Virtual has an in in 1999? ; is Barcelona Virtual’s? ; and 1993? ; and 1993 has an in 1995? What is Barcelona

Table A.6: Example of 3 adapter QKVMLP merging on unseen question

<b>Question</b>	
How many workers does Barcelona Virtual have?	
<b>Ground Truth</b>	
Barcelona Virtual has a team of 15 professionals.	
<b>Merge and unload</b>	<b>Concatenate</b>
<b>Answer:</b> What is Barcelona Virtual’s approach to continuous innovation?What is Barcelona Virtual’s approach to digital marketing?What is Barcelona Virtual’s approach to touser? Barcelona Virtual’s approach to e-commerce?What is Barcelona Virtual’s approach to continuous e-commerce?What is	<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. ;assistant¿Barcelona Virtual has a team of 15 professionals. ;assistant¿Barcelona Virtual has a team of 15 professionals. ;assistant
<b>SVD</b>	<b>Linear</b>
<b>Answer:</b> Barcelona Virtual has a team of 15 professionals. It has a 50-year-old? What is Barcelona Virtual’s specialty? ;assistant¿Barcelona Virtual specializes in digital marketing, increasing online sales, and internationalizing beyond Catalonia Virtual	<b>Answer:</b> Barcelona Virtual has 30 years of Barcelona Virtual has? ¿What is Barcelona Virtual’s?? ¿What is Barcelona Virtual’s???¿ does Barcelona Virtual have? ¿What is Barcelona Virtual’s?? ¿

Table A.7: Example of 4 adapter QKVMLP merging on known question