UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Rustam Abdumalikov

# Parameter-efficient fine-tuning in reading comprehension

Master's Thesis (30 ECTS)

| | |
|---|---|
| Supervisor(s): | Yova Kementchedjhieva, PhD |
| | Kairit Sirts, PhD |

Tartu 2023

## Parameter-efficient fine-tuning in reading comprehension

**Abstract:**
Question Answering is an important task in Natural Language Processing. There are different approaches to answering questions, such as using the knowledge learned during pre-training or extracting an answer from a given context, which is commonly known as reading comprehension. One problem with the knowledge learned during pre-trained is that it can become outdated because we train it only once. Instead of replacing outdated information in the model, an alternative approach is to add updated information to the model input. However, there is a risk that the model may rely too much on its memorized knowledge and ignore new information, which can cause errors. Our study aims to analyze whether parameter-efficient fine-tuning methods would improve the model's ability to handle new information. We assess the effectiveness of these techniques in comparison to traditional fine-tuning for reading comprehension on an augmented NaturalQuestions dataset. Our findings indicate that parameter-efficient fine-tuning leads to a marginal improvement in performance compared to fine-tuning. Furthermore, we observed that data augmentations contributed the most substantial performance enhancements.

**Keywords:**
natural language processing, question answering, fine-tuning, transformers, neural networks

**CERCS:**
P176, Artificial intelligence

## Põhjalik uuring parameetrite tõhusate peenhäälestuste kohta avatud kujuldomeeni küsimusele vastamine

**Lühikokkuvõte:**

Küsimustele vastamine on oluline ülesanne loomuliku keele töötluse valdkonnas. Küsimustele vastamise süsteemide arendamisel on erinevaid lähenemisviise, näiteks eeltreenitud teadmiste kasutamine või vastuse leidmine etteantud kontekstist, mida tavaliselt nimetatakse masinlugemise mõistmiseks. Üks probleem eeltreenitud teadmistega on see, et need võivad vananeda, sest mudelit eeltreenitakse ainult ühe korra. Selle asemel, et mudeli infot treenimisega uuendada, on üheks alternatiiviks värskendatud teabe lisamine etteantud kontekstina mudeli sisendisse. Siiski on võimalik, et mudel võib liiga palju toetuda oma parameetritesse salvestatud teadmistele ja ignoreerida etteantud kontekstis sisalduvat uut teavet ning see võib põhjustada vigu. Selle magistritöö eesmärk on uurida, kas vähendatud parameetritega peenhäälestusmeetodid parandaksid mudeli võimet käsitleda uut teavet. Töös hindasime nende tehnikate tõhusust võrreldes tavapärase peenhäälestusega masinlugemise mõistmise ülesandel kasutades augmenteeritud NaturalQuestions andmestikku. Töö tulemused näitavad, et vähendatud parameetritega peenhäälestusmeetodid mõnevõrra parandavad masinlugemise mõistmise täpsust võrreldes tavapärase peenhäälestusega. Lisaks leidsime, et andmete augmenteerimine panustas kõige enam mudeli ennustustäpsuse tõusu.

**Võtmesõnad:**

loomuliku keele töötlus, küsimustele vastamine, peenhäälestus, transformer, närvivõrgud

**CERCS:**

P176, Tehisintellekt

# Acknowledgement

I am deeply grateful to my supervisors, Yova Kementchedjhieva and Kairit Sirts, for their unwavering guidance, support, and generosity with their time throughout my academic journey. Their invaluable guidance has been instrumental in the completion of this work, especially during moments when I felt lost or overwhelmed. Their thought-provoking discussions and feedback have challenged me to think critically and creatively about my research and have helped me to develop a deeper understanding of the subject matter.

In particular, I would like to express my appreciation to Kairit Sirts for her feedback and support in improving my writing skills. Her insightful questions have helped me to structure my arguments more effectively and have contributed to the overall quality of this work.

I would also like to thank my mother for her unwavering support and interest in my work. Her encouragement and belief in me have been a constant source of motivation and have helped me to persevere during challenging times.

Lastly, I would like to acknowledge the University of Tartu for providing free access to HPC resources, which have been vital to the completion of this work. I am immensely grateful for this opportunity and the support that the university has provided throughout my academic journey.

# Contents

# 1 Introduction

Question Answering is an essential task in Natural Language Processing (NLP), which encompasses different approaches. One of these approaches involves utilizing the knowledge learned during pre-training to answer questions. Another approach involves generating or extracting an answer from a given context, which is commonly known as reading comprehension. The former approach relies on parametric knowledge, which is the knowledge contained within the model weights, while the latter approach employs contextual knowledge. Contextual knowledge refers to external knowledge, such as a Wikipedia passage, that is provided to the model during inference along with the question.

Parametric knowledge is fixed at the time of model training, which can become a problem as the answers could become outdated. For instance, if the model was trained on facts up until 2017 and we ask it to answer the question "Who is the President of the US?" today, the model might provide an outdated answer like "Barack Obama" instead of the current President "Joe Biden". This highlights the need to continuously update the model's knowledge to ensure accurate answers. It is not an easy task to replace outdated facts inside model weights with new ones. This is because it is challenging to identify outdated information within large-scale models with millions of parameters. Moreover, incorporating the new information into the model weights without disrupting the existing information and structure of the model can be difficult.

Instead of replacing outdated facts, an alternative approach is to augment the model input with contextual knowledge containing updated information. The model can then be trained to make better predictions based on the new information. However, following studies [LPC$^+$22] and [FWI$^+$18] identified a potential shortcoming of this approach. The model may rely too much on its memorized knowledge and ignore the new contextual information, which can cause hallucinations.

In [WMB$^+$20], researchers encountered the similar problem where their models relied on wrong patterns and did not pay enough attention to the actual question being asked. This was problematic because the models would still predict the correct answer with high confidence even when an important part of the question was replaced with a random alternative. The problem explored by [NAH$^+$22] is related to knowledge conflicts that arise when there is a contradiction between parametric and contextual knowledge. Since it is difficult to determine which knowledge source the model used to make its prediction, the resulting predictions are likely to be inaccurate and less reliable. The solution to this problem involves improving the model's sensitivity to the input, which can help disentangle the conflicting sources of knowledge and improve the accuracy and reliability of its predictions.

Both studies were successful in making the model pay more attention to the context, with data augmentation proving to be especially important in both cases. However, we believe that further improvements in generalization can be achieved by using parameter-efficient fine-tuning (PEFT) instead of traditional fine-tuning. Traditional fine-tuning often results in the model memorizing the training data due to its high capacity. However, this might not be the case with PEFT since only a small number of parameters are being fine-tuned. Thus limited model capacity could discourage it from relying too heavily on memorization and encourage the model to explore other strategies that may be more effective for generalization. Furthermore, it is worth noting that fine-tuning models often suffer from the phenomenon of catastrophic forgetting [SSMK18], which can lead to a decrease in performance on previously learned tasks. This is because the fine-tuning process updates the pre-trained weights of the model, which can cause it to forget some of the knowledge it learned during pre-training. Compared to fine-tuning, PEFT may not experience catastrophic forgetting as severely or at all since pre-trained weights are typically frozen during the training of the downstream task. By reducing the impact of catastrophic forgetting, it is possible to improve the overall model performance.

Our study aims to conduct an investigation into improving generalization in reading comprehension on the augmented NaturalQuestions dataset provided by [NAH$^+$22]. To accomplish this goal, we will utilize a combination of data augmentation and various PEFT methods, including LoRA, Bottleneck-Adapter, and prompt-tuning, to improve performance. We will compare the effectiveness of these techniques against fine-tuning.

## 1.1 Goals

This work comprises two stages. Firstly, we will replicate a specific set of experiments from [NAH$^+$22] to establish a strong baseline. Next, we will address the research questions we have posed.

- **Goal:** Our initial goal is to replicate the results of [NAH$^+$22], specifically the experiments that prioritize contextual knowledge over parametric knowledge. These experiments involve training the model to generate a single answer based on contextual knowledge and will serve as a useful baseline for our work. By comparing the performance of PEFT methods against traditional fine-tuning in reading comprehension, we can better understand the limitations and advantages of PEFT methods.

- **Research question:** Can parameter-efficient fine-tuning, which involves training a smaller number of parameters than traditional fine-tuning, achieve higher accuracy in reading comprehension on the NaturalQuestions dataset?

## 1.2 Outline

This thesis is structured as follows:

- Chapter 2 discusses related studies.

- Chapter 3 explains the background details of parameter-efficient fine-tuning techniques, including LoRA, Bottleneck-Adapter, Prompt-Tuning, and traditional fine-tuning.

- Chapter 4 describes the training setup, datasets used, and data augmentation techniques employed in the experiments.

- Chapter 5 presents the results of the experiments and key findings.

- Chapter 6 discusses the potential reasons for the observed results and their implications.

- Chapter 7 summarizes the findings and outlines future work.

# 2 Related Work

**Undersensitivity In Reading Comprehension** [WMB$^+$20] showed that their models were not sensitive enough to the given input in reading comprehension on SQuAD2.0 [RJL18]. Specifically, they were looking for adversarial questions, which were questions that had been modified by substituting a named entity of the same type from a shared pool. Despite the modification, the model predicted the same answer with increased confidence, as if nothing had been changed. To tackle this problem, the researchers used data augmentation and adversarial training techniques. Their results showed that data augmentation was the most effective strategy.

Our work is inspired by the effectiveness of data augmentation to increase model sensitivity in reading comprehension.

**Knowledge Disentanglement** [NAH$^+$22] demonstrated the effectiveness of disentangling two sources of knowledge, namely parametric and contextual, in reading comprehension on the NaturalQuestions dataset [KPR$^+$19]. They achieved this through data augmentation, including counterfactual and answerability augmentations. Counterfactual augmentation replaced the original answer in a context with a different one to encourage the model to focus on the input rather than relying on its parametric knowledge. Answerability augmentation aimed to teach the model to abstain from answering questions with impossible answers, using empty and random contexts. Results showed that counterfactual augmentation significantly improved performance over models that were trained on datasets with such augmentation, while answerability slightly decreased it. However, combining the two augmentations complementarily produced the highest accuracy.

Our work builds on this research, focusing on parameter-efficient fine-tuning methods. Our study aims to investigate whether these methods can improve generalization over traditional fine-tuning in reading comprehension on the NaturalQuestions dataset.

# 3 Background

This section provides an overview of the methods used in this thesis, including fine-tuning and parameter-efficient fine-tuning techniques such as LoRA, Bottleneck-Adapter, and prompt-tuning.

## 3.1 Pre-training and Fine-Tuning

Model training is an inherently resource-intensive process [HVD15], especially when it comes to deep learning models. These models require vast amounts of data in order to achieve high levels of accuracy and generalization. Training such models from scratch for every single task can be prohibitively expensive in terms of both time and resources. However, this challenge can be overcome through transfer learning [ZQD+20], a two-stage process that involves pre-training and fine-tuning.

Transfer learning represents a one-to-many relationship, where pre-training serves as the foundation for a model's learning, and fine-tuning builds upon that foundation to enhance the model's performance on a specific task. In essence, pre-training provides a general understanding of the data, while fine-tuning adapts the model to the specific task by adjusting its parameters. This approach not only reduces training time and computational resources but also enhances the model's ability to generalize to new tasks.

## 3.2 Parameter-Efficient Fine-Tuning

Fine-tuning is a common technique for adapting a pre-trained model to a new task. However, it has a significant drawback known as catastrophic forgetting [XZYL20]. This occurs because the fine-tuning process involves updating pre-trained model weights to fit the new data distribution of the target task, which can cause the model to overfit to the new task and forget the knowledge it has previously learned.

Parameter-efficient fine-tuning (PEFT, [DQY+23]) is a technique that potentially might prevent catastrophic forgetting while still enabling adaptation to a new task. In PEFT, the weights of the pre-trained model are frozen, and only a small set of additional weights are fine-tuned. This approach limits the model's capacity to memorize training data and encourages it to explore other strategies that promote better generalization, which is crucial for avoiding overfitting [ZBH+17, CTW+21] and achieving good performance on new, unseen data. Therefore, with PEFT, the model can potentially adapt to new tasks without forgetting previous knowledge while still improving its ability to generalize.

PEFT is an active area of research, and numerous methods are available. For the purposes of this thesis, we will consider only the following methods: LoRA [HSW+21], Prompt-tuning [LARC21], and Bottleneck Adapter [HGJ+19a].

### 3.2.1 LoRA

Fine-tuning adapts a pre-trained model to a downstream task by making updates to its parameters represented as $W_0 + \Delta W$, where $W_0$ is the pre-trained parameter matrix and $\Delta W$ represents task-specific updates. Since both matrices have the same size and are full-rank, implies that fine-tuning is a full-rank adaptation technique.

In a line of works by [CFC+20] and [PRR20] it has been shown that pre-trained models such as BERT are redundant in their capacity, allowing for significant sparsification without much degradation in end metrics. This tells us that we do not need to adapt all the weights as some of them are insignificant.

LoRA [HSW+21] which stands for low-rank adaptation is based on such a premise. Unlike fine-tuning that operates in the full space of pre-trained weights, LoRA by decomposing $\Delta W$ as a multiplication of two low-rank matrices $A$ and $B$ operates in a lower subspace. Specifically, if $W_0 \in \mathbb{R}^{d\times k}$, then $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$, where $r \ll \min(d, k)$. As a result, $\Delta W$ can be represented as $\Delta W = BA$, and the forward pass can be expressed as $h = W_0x + BAx$, instead of $h = W_0x + \Delta Wx$. The architecture of LoRA is illustrated in Figure 1.

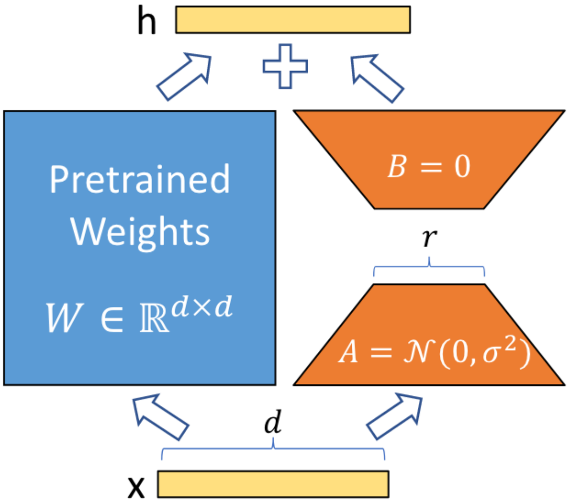

h
Pretrained Weights
$W \in \mathbb{R}^{d\times d}$
$B = 0$
$r$
$A = \mathcal{N}(0, \sigma^2)$
$d$
x

Figure 1. Pre-trained parameters $W$ will be frozen, and only $A$ and $B$ will be trained. To initialize $A$, random Gaussian initialization was used and $B$ was initialized with zeros, so $\Delta W = BA$ is zero at the beginning of the training. **This image was adapted from [HSW+21].**

### 3.2.2 Prompt-tuning

To begin exploring prompt-tuning [LARC21], it is important to understand the concept of association. The association is a powerful tool for organizing memories in a way that facilitates easy retrieval of relevant information according to [AW13]. For instance, when one hears the word "banana", one might immediately think of the color yellow or monkeys because these concepts are associated with bananas based on their past experiences. In psychology, this phenomenon is known as priming, which refers to the ability of the association to influence a person's response to a stimulus, such as an image or a set of words [AW13].

In natural language processing (NLP), priming can also be used to steer the output of language models. Specifically, word-level steering is often accomplished using hard or discrete prompts, which involve selecting concrete words to trigger desired associations [WYG$^{+}$22]. However, this method has limitations, such as the need to manually search for relevant words and the lack of scalability. To address these limitations, soft prompts were introduced as an alternative [WYG$^{+}$22]. Unlike hard or discrete prompts, which involve selecting concrete words, soft prompts operate on virtual words (i.e., words that are not part of vocabulary) representation that can be learned.

The soft prompt is a flexible concept that offers various possibilities for arranging its virtual tokens. In contrast, prompt-tuning is a specific algorithm that employs these virtual tokens in a prefix format. Specifically, prompt-tuning concatenates the virtual tokens with the input and feeds it to a pre-trained model. The length of the prompt can be adjusted as a hyperparameter. Essentially, prompt-tuning provides a way to fine-tune a pre-trained language model by leveraging the power of virtual tokens.

### 3.2.3 Bottleneck Adapter

Bottleneck-Adapter, as proposed in [HGJ$^{+}$19b], is a highly efficient method for model adaptation, which addresses the need for parameter-efficient adaptation techniques. Compared to fine-tuning, which adapts the entire pre-trained model (i.e., 100%), Bottleneck-Adapter uses only 3.6% of the parameters per task.

To adapt the pre-trained model to a specific downstream task, Bottleneck-Adapter introduces adapter layers into each transformer block, as shown in Figure 3 on the left. These adapter layers are added after the Multi-headed Attention and Feed-forward layers, as depicted in Figure 2, which illustrates the changes in the forward pass of a transformer block. An adapter layer is a small bottleneck neural network that consists of a few layers with a skip connection, as shown in Figure 3 on the right.

During adapter tuning, only the green layers, which include the adapter layer and layer normalization parameters, are trained using downstream data. By training only these adapter layers, the rest of the pre-trained model remains fixed, which makes Bottleneck-Adapter a highly parameter-efficient method for adaptation.

**Before**

```
function TRANSFORMERBLOCK(input)
    $\alpha \leftarrow attention(input)$
    $h \leftarrow feed\_forward(\alpha)$
end function
```

**After**

```
function TRANSFORMERBLOCK(input)
    $\alpha \leftarrow attention(input)$
    $\beta \leftarrow bottleneck\_adapter(\alpha)$
    $\phi \leftarrow feed\_forward(\beta)$
    $h \leftarrow bottleneck\_adapter(\phi)$
end function
```

Figure 2. Simplified forward pass of a transformer block

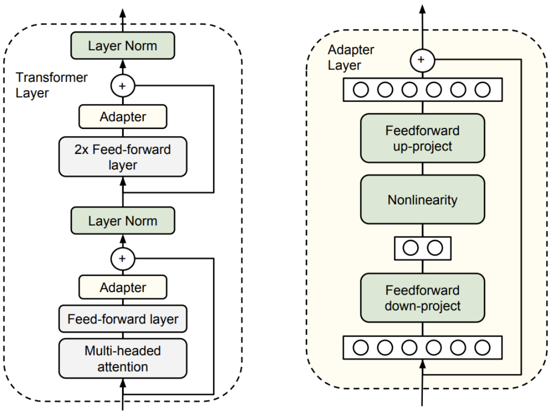


Figure 3. On the left shown is a transformer block with two additional adapter layers. And on the right bottleneck architecture of the adapter layer. **This image was adapted from [HGJ$^+$19b]**.

# 4 Methods

This chapter will cover a range of topics related to this work. We will describe the tools, datasets, and data augmentation techniques used in the study. Additionally, we will discuss the replication of the previous work [NAH$^+$22] on this topic, followed by an explanation of the model architecture. We will also delve into the training, inference procedures, and evaluation methods used in the study.

## 4.1 Tools

To enhance the readability and coherence of this work, we have utilized ChatGPT by [Ope22] and Grammarly by [ASL09] to correct spelling mistakes.

Additionally, in this work, we used the implementation of LoRA and Bottleneck-Adapter developed by [PRP$^{+}$20].

## 4.2 Dataset

The dataset used in this research was provided by [NAH$^{+}$22], and has already predefined train, validation, and test splits (refer to Tables 1 and 2). It is the Natural Questions (NQ) dataset [KPR$^{+}$19], which includes questions that people ask using Google search, along with long and short answers. The long answers are passages from Wikipedia articles that may contain the answer. For this study, only the gold passages, i.e., passages that definitely contain an answer, were used and referred to as context. The short answers represent the actual answer.

| | Closed Book | Factual | Counterfactual | Empty | Random |
|---|---|---|---|---|---|
| Train | 85,540 | 85,540 | 30,653 | 85,540 | 85,540 |
| Validation | 21,386 | 21,386 | 7,698 | 21,386 | 21,386 |
| Test | 1,365 | 1,365 | 1,365 | 1,365 | 1,365 |

Table 1. Datasets splits. This Table was adapted from [NAH$^{+}$22]

| **Abbreviation** | **Data** | | | | |
|---|---|---|---|---|---|
| | **c**losed **b**ook | **f**actual | **c**ounter**f**actual | **a**nswerability | |
| | | | | empty | random |
| cb | + | - | - | - | - |
| f | - | + | - | - | - |
| f+cf | - | + | + | - | - |
| f+a | - | + | - | + | + |
| f+cf+a | - | + | + | + | + |

Table 2. The abbreviations used for the training datasets in this table should be considered in conjunction with Table 1. To understand what the training set for *f+a* would entail and how many entries it would contain, look for *f+a* in the table. The plus sign indicates that factual, empty, and random data will be utilized. You can then refer to Table 1 and perform the calculation $85,540 + 85,540 + 85,540$ to determine that the *f+a* dataset contains $256,620$ training entries.

## 4.3 Data Augmentation

This work utilizes an important aspect of the dataset provided by [NAH$^{+}$22], such as data augmentation. [NAH$^{+}$22] used two sets of data augmentations - counterfactual and answerability - each serving a distinct purpose. Counterfactual augmentation helps improve the model's attention to the input by substituting facts, as shown in Figure 4, where the correct year was replaced with an incorrect one. To answer a question with counterfactual context, the model needs to be sensitive to the context since only the context contains the correct counterfactual answer. Answerability augmentation, on the other hand, aims to improve model robustness by teaching it to refrain from answering questions that cannot be answered. This is achieved by providing empty context and randomizing context for each question, as shown in Figure 5. To provide the correct answer, the model also needs to be sensitive to the input, as the only correct answer is in the contextual knowledge.

**Factual**

**Question:** When World War II started?
**Original Context:** World War II or the Second World War, often abbreviated as WWII or WW2, was a global conflict that lasted from 1939 to 1945. The vast majority of the world's countries, including ...
**Original Answer:** 1939

**Counterfactual**

**Question:** When World War II started?
**Substitute Context:** World War II or the Second World War, often abbreviated as WWII or WW2, was a global conflict that lasted from 2000 to 2005. The vast majority of the world's countries, including ...
**Substitute Answer:** 2000

Figure 4. Counterfactual substitution example. In the left box, you could see the original factual information. Whereas in the right box, factual information was substituted.

**Empty**

**Question:** When World War II started?
**Context:**
**Answer:** unanswerable

**Random**

**Question:** When World War II started?
**Context:** William Bradley Pitt (born December 18, 1963) is an American actor and film producer ...
**Answer:** unanswerable

Figure 5. Example of answerability augmentation. In both case model should abstain from answering by generating the token 'unanswerable'.

## 4.4 Model Architecture, Training and Inference

To adhere to the methodology outlined in [NAH+22], we will employ the original T5 models described in [RSR+20] for this thesis. The T5 model is a transformer-based encoder-decoder model that was trained on the C4 dataset, which is a refined version of the Common Crawl dataset, using a span-corruption objective.

For our experiments, we utilized the T5-Large model from [Hug20], which contains 770M parameters. As mentioned in [NAH+22], the model size has a significant impact on its performance and knowledge capacity. Unfortunately, we were limited by the computational resources available to us and thus had to utilize the T5-Large model, which was the largest model that could be run on our HPC GPUs.

### 4.4.1 Baseline

We aimed to replicate the results from [NAH+22] by fine-tuning T5-Large models, for two key reasons: firstly, to fill a gap in the results that [NAH+22] did not provide, and secondly, to ensure fairness and eliminate potential variations in implementation by training and testing various methods inside a single codebase, so that any observed differences would be solely due to the specific methods employed. Despite precisely following their instructions and using their datasets, we were unable to achieve the same results. We also used the authors' codebase to attempt to reproduce the results reported in their paper. However, results with their codebase were even lower than those achieved using our own codebase, as shown in Tables 3. Given that our fine-tuning approach yielded better results, it reinforces our decision to employ it for further analysis.

| **Trained on *f*** | | |
|---|---|---|
| | **factual** | **counterfactual** |
| **Ours:** Fine-Tuning | 73.85 | 64.32 |
| **Theirs(Local):** Fine-Tuning | 72.82 | 63.08 |
| **Theirs(Paper):** Fine-Tuning | 76.34 | 67.84 |

| **Trained on *f*+*cf*** | | |
|---|---|---|
| | **factual** | **counterfactual** |
| **Ours:** Fine-Tuning | 71.28 | 76.78 |
| **Theirs(Local):** Fine-Tuning | 71.94 | 74.95 |
| **Theirs(Paper):** Fine-Tuning | 75.75 | 76.04 |

Table 3. Replication results on *f* and *f*+*cf* datasets for our fine-tuning, theirs(local) fine-tuning on our computer with their codebase, and theirs(paper) fine-tuning results that they reported in [NAH+22].

### 4.4.2 Training Procedure

The model was provided with a concatenation of a question and context during both the training and inference phases. This input structure was adapted from [NAH+22], and the datasets provided by [NAH+22] included corresponding **input** column with the following format: *question: <question>\n context: <context>*. The question and context are separated by a corresponding token, followed by a colon and space. The context part is moved to a new line using the **\n** character. It is worth noting that [NAH$^+$22] used a special format for the output, as their goal was to learn answer disentanglement. However, in this study, we did not follow their format and instead instructed the model to generate output in a free-form manner (i.e., just answer).

To follow the [NAH$^+$22] codebase, the input was truncated to a maximum length of 256 tokens during training and 396 tokens during testing, while the output was truncated to a maximum length of 32 tokens. To ensure a fair chance for convergence, a model was trained for each method until it reached convergence or hit the maximum number of epochs, which was set to 100. This was important to consider since different methods may converge at different rates, so setting the number of training steps or epochs could lead to suboptimal results. Convergence rates were not a concern in [NAH$^+$22] since only fine-tuning was employed.

Adafactor with a constant learning rate of 0.0001 was used as the optimizer. In [NAH$^+$22], they used a constant learning rate, but with the AdamW optimizer instead. Adafactor was chosen as the optimizer in this study because it was used in the pre-training of T5 models, which gave confidence in its effectiveness.

### 4.4.3 Inference Procedure

The inference procedure for reading comprehension involves generating an answer that is based on the information provided in the input. The generation process is limited to a maximum of 80 tokens. Additionally, we have set the repetition penalty to 2.5 and the length penalty to 1.0 and enabled early stopping and cache usage. These settings are consistent with those used in the codebase described in the [NAH$^+$22] paper.

## 4.5 Evaluation

To evaluate the models, we utilized a checkpoint that was chosen based on the best validation accuracy attained and applied it to the test set. The Exact Match (EM) metric, as defined in [RZLL16] and shown in Equation 1, was utilized as the evaluation metric. This metric measures the proportion of predictions that exactly match the ground truth, thereby providing a precise measure of the model's accuracy.

$$ExactMatch = \frac{\text{number of correctly predicted examples}}{\text{total number of examples}} \tag{1}$$

# 5 Results

This section will present and compare the results obtained from fine-tuning, LoRA, Bottleneck-Adapter, and prompt-tuning. Additionally, we will delve into the impact of data augmentation techniques on the performance of these approaches and explore the behavior they learned as a result.

| | **Fine-Tuning** | **LoRA** | **Bottleneck-Adapter** | **Prompt-Tuning** |
|---|---|---|---|---|
| factual | | | | |
| *cb* | 19.93 | 30.40 | 35.75 | **61.98** |
| *f* | 73.04 | 73.19 | **73.48** | 61.98 |
| *f*+*cf* | 72.16 | 71.43 | **72.82** | 62.34 |
| *f*+*a* | **74.07** | 70.99 | 71.65 | 57.07 |
| *f*+*cf*+*a* | 70.99 | 71.28 | **72.02** | 58.46 |
| counterfactual | | | | |
| *cb* | 3.96 | 13.70 | 16.04 | **56.12** |
| *f* | 63.22 | **65.42** | 65.06 | 56.48 |
| *f*+*cf* | 77.66 | 79.78 | **80.44** | 56.34 |
| *f*+*a* | 62.86 | **63.44** | 60.66 | 51.65 |
| *f*+*cf*+*a* | 76.19 | 78.17 | **79.63** | 52.38 |

Table 4. All methods trained on different datasets (such as *f*, *f*+*cf*, etc), and evaluated on factual and counterfactual test splits.

## 5.1 Main Results

Table 4 summarizes the results of fine-tuning and PEFT methods evaluated on factual and counterfactual test sets. The factual test set comprises fact-based information, whereas the counterfactual test set contains incorrect facts (the examples for factual and counterfactual were shown earlier in Figure 4). As per the results, it was observed that the PEFT methods slightly performed better than fine-tuning method on most datasets, whereas prompt-tuning consistently yielded lower performance compared to other methods. However, it was the counterfactual augmentation that demonstrated the most significant improvement in performance.

For the *f*+*cf* and *f*+*cf*+*a* datasets, the Bottleneck-Adapter method achieved the highest performance on both test splits, followed by LoRA, which exhibited the second-best results on the counterfactual test split. The prompt-tuning method also demonstrated superiority over other models trained on the *cb* dataset in both factual and counterfactual test splits. In contrast, the fine-tuned model performed the worst among all models trained in the closed book setting (i.e., setting where only question is given in the input, *cb* dataset), exhibiting the lowest performances in both test splits.

We found that counterfactual augmentation was effective in improving performance across all methods except for prompt-tuning. As Table 4 demonstrates that the difference in performance between using and not using counterfactual examples during the training corresponds to approximately 15%. In contrast, answerability augmentation led to a drop in performance for all models.

The overall observation was that significant improvements in generalization were not achieved by PEFT methods alone, resulting in only a minor improvement over fine-tuning. However, the most notable performance boost was observed with the counterfactual augmentation, which was found to be effective for all methods except prompt-tuning. It is suspected that the poor performance of prompt-tuning may be attributed to the relatively smaller number of learnable parameters compared to the other methods.

## 5.2 Catastrophic Forgetting

The interesting results in Table 4 for closed-book setting could indicate the validity of our initial assumption that PEFT methods are less susceptible to the catastrophic forgetting phenomenon, which leads to a degradation in performance on tasks learned during pre-training. However, as Table 4 only provides final performance scores, it is impossible to conclusively determine the truthfulness of our initial assumption.

Instead, we need to examine the progression of these performances, as shown in Figure 6. This Figure reveals two interesting insights. First, since all methods started with high accuracy after the first epoch and did not see the context during adaptation, it implies that the pre-trained model already had the ability to extract answers from context. Second, adaptation led to a decrease in performance in all methods, indicating that they all suffer from catastrophic forgetting, but fine-tuning declined faster than other methods, suggesting that it forgets faster and to a greater extent. The combination of these two insights tells us the following story: the pre-trained model was capable of answer extraction, and adapting the pre-trained model to a closed book task led to a degradation of such capability.

Considering that the fine-tuned model lost more of its extractive capability than PEFT methods confirms our initial assumption that PEFT methods, which operate over a small subset of weights (usually under 1%), experience catastrophic forgetting to a lesser extent.

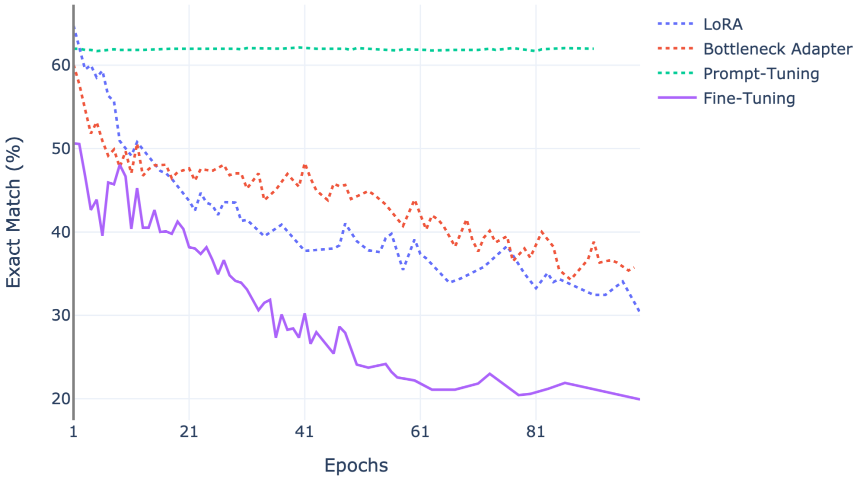


Figure 6. Progressive evaluation on factual test split of various methods trained in a closed book setting, i.e., where the input consists of only a question. **Important Note**: starting epoch is one, not zero.

## 5.3 Answerability Results

| | **Fine-Tuning** | **LoRA** | **Bottleneck-Adapter** | **Prompt-Tuning** |
|---|---|---|---|---|
| empty | | | | |
| *f* | 0.0 | 0.0 | 0.0 | 0.0 |
| *f*+*cf* | 0.0 | 0.0 | 0.0 | 0.0 |
| *f*+*a* | 1.0 | 1.0 | 1.0 | 1.0 |
| *f*+*cf*+*a* | 1.0 | 1.0 | 1.0 | 1.0 |
| random | | | | |
| *f* | 0.0 | 0.0 | 0.0 | 0.0 |
| *f*+*cf* | 0.0 | 0.0 | 0.0 | 0.0 |
| *f*+*a* | 98.46 | 98.32 | 97.22 | 89.30 |
| *f*+*cf*+*a* | 98.10 | 98.53 | 96.56 | 87.25 |

Table 5. All methods trained on different datasets (such as *f*, *f*+*cf*, etc), and evaluated on empty and random test splits.

Table 5 presents a summary of the results obtained from the evaluation of the fine-tuning and PEFT methods on both empty and random test splits. The empty test split refers to a context-free setting, where the context was removed from each question. On the other hand, the random test split involved randomizing the context of each question, as was illustrated earlier in Figure 5. Only datasets that included answerability augmentation gave a non-zero performance, such as *f*+*a* and *f*+*cf*+*a*.

Notably, fine-tuning, LoRA, and Bottleneck-Adapter achieved near-perfect performance on the random and empty test splits after just the first epoch, according to Figure 7. In contrast, Prompt-Tuning required 25 epochs to reach a high level of performance. While this early success is impressive, it raises questions about the value of answerability augmentation, as it may suggest that the models have discovered a shortcut strategy, which we explore further below.

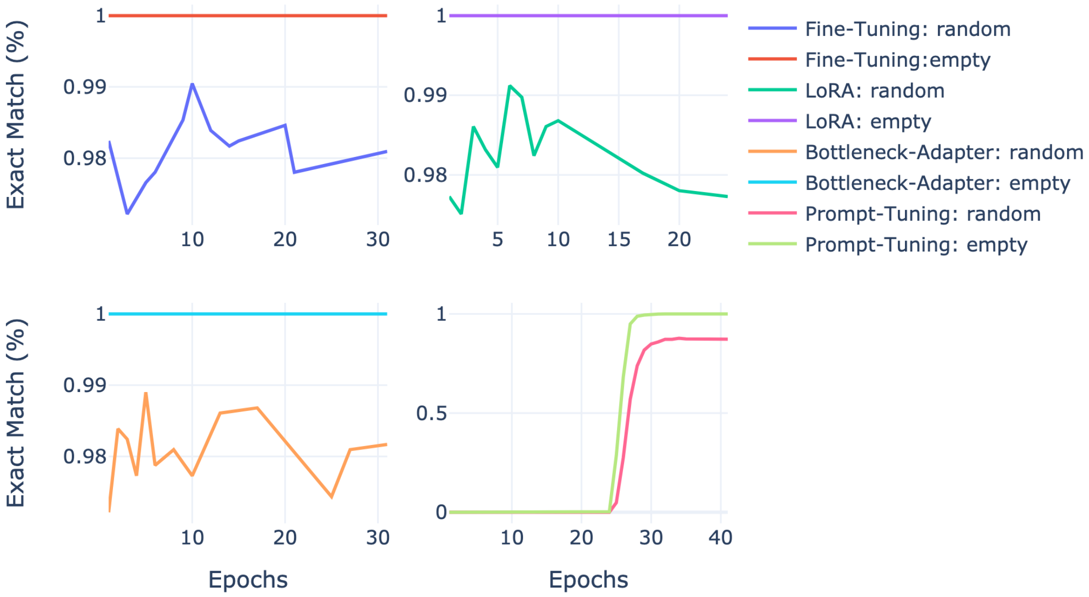


Figure 7. Progressive evaluation on empty and random test splits of various methods trained *f*+*cf*+*a*.

### 5.3.1 Understanding Random Context

In the last section, we questioned the effectiveness of answerability augmentation because most methods were able to achieve high accuracy by only going through the dataset once. This section's goal is to focus solely on the random test split.

To begin, let's analyze the example with the random context shown on the right side of Figure 8. This will help us understand why the performance on the random test split is nearly perfect and achieved so quickly.

In this example, we see that the question and context are entirely unrelated. Given that neural networks, particularly transformers [VSP$^{+}$17], learn statistical correlations, this leads us to believe that with a random context, there is nothing for the model to correlate with. Perhaps just having the correct entity type (e.g., country) would be enough to make the difference.

In order to test it, we generated a more challenging context using ChatGPT that contains the correct entity type but provides details that are irrelevant to the question at hand as shown in the example on the left in Figure 8. This question suggests that the answer should be a country, and while there is a country mentioned in the context, the description provided contains extraneous information that is not relevant to the question. To create this challenging context, we used the following prompt: *write a fictional story with a maximum of two sentences that must include "<factual answer>", and must not contain any words from "<question>".*

| **ChatGPT context** | **Random context** |
|---|---|
| **Question:** Panda is a national animal of which country? | **Question:** Panda is a national animal of which country? |
| **Context:** In the dusty streets of Beijing, a young girl named Mei-Ling gazed up at the towering skyscrapers, dreaming of one day traveling the world to see the Great Wall of China. | **Context:** The fourth season began airing on October 10, 2017 and is set to run for 23 episodes on The CW until May 22, 2018. |

Figure 8. Examples of ChatGPT context (more examples can be found in Appendix II Table 15) and random context. For ChatGPT context I highlighted the expected correct answer (in blue) for factual context.

The results of model evaluation on ChatGPT context are presented in Table 6. Mainly it demonstrates a significant contrast in performance between the models that were trained with and without answerability augmentation such as *f+a* and *f+cf+a*.

It's worth noting that models trained on answerability augmentation experienced a four-fold decrease in their ability to extract answers. This is actually desirable, as we want our models to refrain from answering when faced with an unrelated question and context. The models were able to identify when the question and context were unrelated and, in most cases, opted not to provide an incorrect answer, as indicated in Table 7. This demonstrates the effectiveness of using answerability augmentation, especially the random context technique.

However, the relatively high performance of the models trained on *f* and *f*+*cf* datasets raises concerns that they may rely on overly simplistic extraction strategies and lack the robustness required for deployment in real-world scenarios.

| **Methods** | **f** | **f+cf** | **f+a** | **f+cf+a** |
|---|---|---|---|---|
| Fine-Tuning | 84.84 | 85.20 | 24.40 | 25.05 |
| Bottleneck-Adapter | 87.11 | 85.35 | 30.04 | 22.27 |
| LoRA | 87.69 | 86.30 | 21.98 | 23.52 |
| Prompt-Tuning | 78.53 | 78.39 | 35.53 | 35.16 |

Table 6. Percentage of questions for which models (trained on *f*, *f*+*cf*, etc) provided correct factual answer on ChatGPT context.

| **Methods** | **f+a** | **f+cf+a** |
|---|---|---|
| Fine-Tuning | 73.99 | 73.26 |
| Bottleneck-Adapter | 67.84 | 76.70 |
| LoRA | 76.56 | 74.65 |
| Prompt-Tuning | 61.25 | 61.68 |

Table 7. Percentage of questions for which models (trained on *f*+*a* and *f*+*cf*+*a*) abstained from answering (i.e., generated token 'unanswerable') on ChatGPT context.

### 5.3.2 Understanding Model Robustness

A concerning detail about the models trained on the *f* and *f*+*cf* datasets are highlighted in the previous section's results. It was observed that these models exhibited a lack of robustness, as evidenced by their high performance in Figure 6, suggesting that they are merely simple extractors that extract the correct entity type without considering other contextual information. Undesirable outcomes could result from using such models in real-world scenarios.

To confirm or refute the assumption that these models are simple extractors, it is necessary to understand the source of knowledge they use to predict answers. It is currently unclear whether the models rely on parametric knowledge, contextual knowledge, or a combination of both. To gain insight into this, substituting factual answers in the

ChatGPT contexts with counterfactual ones could provide a clear understanding of the answer's source. However, this substitution strategy may result in ill-formed phrases. For example, replacing 'China' with 'Hawaii' in the ChatGPT context in Figure 8 would produce a phrase 'The Great Wall of Hawaii' that might disrupt model predictions as it has not encountered such a combination of words before. Such disruptions could result in highly undesirable outputs. For instance, in the case of 'The Great Wall of China,' the model would likely extract the correct entity type. However, in the case of 'The Great Wall of Hawaii,' the model would likely hallucinate. Therefore, before implementing the substitution strategy, it is necessary to assess how sensitive the model is to the answer's surrounding context. This assessment will help determine the feasibility of the substitution strategy.

To test this, the ChatGPT context has been permuted (see Figure 9), and the results in Table 8 show that the models trained on *f* and *f+cf* datasets are relatively insensitive to changes in the surrounding context as performance did not drop as much.

| **ChatGPT context** | **ChatGPT context permuted** |
|---|---|
| **Question:** Panda is a national animal of which country? | **Question:** Panda is a national animal of which country? |
| **Context:** In the dusty streets of Beijing, a young girl named Mei-Ling gazed up at the towering skyscrapers, dreaming of one day traveling the world to see the Great Wall of China. | **Context:** at of see gazed the . skyscrapers, Beijing, to of dreaming of the Mei-Ling a China traveling streets girl In day the Great Wall named world young towering up the one dusty. |

Figure 9. Examples of fictional and permuted fictional context. Correct answers in both cases are highlighted in blue.

| **Methods** | **f**(diff) | **f+cf**(diff) |
|---|---|---|
| Fine-Tuning | 79.93(-4.91) | 80.66(-4.54) |
| Bottleneck-Adapter | 84.40(-2.17) | 83.74(-1.61) |
| LoRA | 87.91(+0.22) | 85.35(-0.95) |
| Prompt-Tuning | 69.30(-9.23) | 69.74(-8.65) |

Table 8. Percentage of questions for which models (trained on *f*, and *f+cf*) provided correct factual answers on ChatGPT permuted context split. The difference is between performances of ChatGPT context and ChatGPT permuted context. **NOTE:** Full table that additionaly includes results for models trained on *cb*, *f+a*, and *f+cf+a* can be found in Appendix I Table 13.

This suggests that the substitution strategy depicted in Figure 10 could be used to understand the source of the answers. The results of evaluation on ChatGPT counterfactual context shown in Table 9, the models trained on *f* and *f*+*cf* primarily leveraged the contextual source, resulting in consistently high accuracies. These results confirm our assumption that the models learned a simple extraction strategy, i.e., extraction of the correct entity type without considering the deeper relationship between the question and context.

Thus, relying solely on datasets that teach a model to extract answers can result in weak robustness. Such models may be overly simplistic, failing to consider the question and context fully, and blindly performing strategies they learned. This underscores the significance of not only instructing the model on its primary task but also thoroughly examining potential edge cases that could enhance the system's overall performance. Neglecting such cases could render the system incomplete in terms of its capabilities and dangerous to deploy in real-world situations.

| **ChatGPT context** | **ChatGPT counterfactual context** |
|---|---|
| **Question:** Panda is a national animal of which country? | **Question:** Panda is a national animal of which country? |
| **Context:** In the dusty streets of Beijing, a young girl named Mei-Ling gazed up at the towering skyscrapers, dreaming of one day traveling the world to see the Great Wall of China. | **Context:** In the dusty streets of Beijing, a young girl named Mei-Ling gazed up at the towering skyscrapers, dreaming of one day traveling the world to see the Great Wall of Hawaii. |

Figure 10. Examples of fictional and fictional counterfactual context. Correct answers in both cases are highlighted in blue.

| **Methods** | **f**(diff) | **f+cf**(diff) |
|---|---|---|
| Fine-Tuning | 77.80(-7.04) | 86.23(+1.03) |
| Bottleneck-Adapter | 77.73(-9.38) | 86.74(+1.39) |
| LoRA | 81.03(-6.66) | 87.91(+1.61) |
| Prompt-Tuning | 69.60(-8.93) | 69.38(-9.01) |

Table 9. Percentage of questions for which models(trained on *f* and *f*+*cf*) provided correct counterfactual answers on ChatGPT contexts with counterfactual answers. The difference is between performances of ChatGPT context and ChatGPT counterfactual context. **NOTE:** Full table that additionaly includes results for models trained on *cb*, *f*+*a*, and *f*+*cf*+*a* can be found in Appendix I Table 14.

# 6 Discussion

The primary goal of this study was to determine whether utilizing PEFT techniques could enhance generalization by discouraging models from relying on memorization strategies. However, the results depicted in Table 4 suggest that the majority of PEFT approaches only gained a marginal improvement over fine-tuning. Conversely, prompt-tuning consistently yielded lower results compared to the other methods.

The study also found that counterfactual augmentation resulted in the most notable performance improvement on counterfactual test split, while answerability augmentation slightly harmed performance. These findings were in line with [NAH$^+$22], except for the performance of models on the combination of those augmentations (i.e., *f*+*cf*+*a*). The authors of [NAH$^+$22] reported that combining all the augmentations complemented each other and resulted in achieving the highest accuracy. However, the current study's results differed from these findings. As in our case answerability augmentation consistently hurt model performance across all datasets. It is noteworthy that the authors of [NAH$^+$22] did not provide clear evidence to support why the combination of both augmentations should be complementary.

Moreover, our findings suggest that models trained without answerability augmentation exhibited simple extractor behavior, which made them less robust and vulnerable to errors. On the other hand, models trained with answerability augmentation showed significant improvements in model robustness.

Although the use of PEFT methods led to a slower rate of catastrophic forgetting, they did not result in improved performance in reading comprehension. However, this approach could still be useful in a multi-task learning (MTL, [Cra20]) scenario. This is because MTL can suffer from the negative transfer ([LSSW22]), which refers to the phenomenon where prior learning of one task negatively impacts the performance of another task. By preventing catastrophic forgetting, PEFT methods could potentially mitigate negative transfer in MTL by preserving important knowledge learned from prior tasks.

## 6.1 Why Did Not PEFT Methods Outperformed Fine-Tuning?

| **Method** | **Number of parameters** | **Percentage** |
|---|---:|:---:|
| Fine-Tuning | 737,668,096 | 100 |
| Bottleneck-Adapter | 12,687,360 | 1.72 |
| LoRA | 2,359,296 | 0.32 |
| Prompt-Tuning | 102,400 | 0.01 |

Table 10. Number of learnable parameters, and their percentage relative to fine-tuning.

factual

| | Correct | Incorrect | | | | | |
|---|---|---|---|---|---|---|---|
| | | **Total** | Table | Partial | Misspelled | Wrong | Hallucination |
| **FT** | 68.28 | 19.93 | 5.35 | 5.93 | 0.07 | 8.35 | 0.22 |
| **BA** | 70.92 | 17.29 | 5.13 | 4.47 | 0.07 | 7.40 | 0.22 |
| **LoRA** | 71.58 | 16.63 | 5.27 | 3.30 | 0.37 | 7.25 | 0.44 |
| **PT** | 61.83 | 26.37 | 7.91 | 5.35 | 0.66 | 11.72 | 0.73 |

Table 11. All methods were trained on the $f$ dataset and evaluated on the factual test split. The **Table** column indicates the fraction of contexts that are presented in table format. The **Partial** column shows the fraction of partially answered questions, while the **Misspelled** column indicates the percentage of misspelled answers. The **Wrong** column refers to the percentage of incorrect answers that were selected from the context, and the **Hallucination** column indicates the percentage of generated answers that were not present in the context. It should be noted that the percentages in the table may not add up to 100 as impossible questions are not included in the calculation. To optimize table width, we used method abbreviations instead of full names. Specifically, we represented Fine-Tuning, Bottleneck-Adapter, and Prompt-Tuning as FT, BA, and PT, respectively.

The goal of this sub-section is to examine why the PEFT methods did not show improvement and determine if there is a potential for enhancement. Table 11 provides a breakdown of the answer predictions, with a focus on the percentage of incorrect answers. Incorrect answers have been further divided into subcategories to facilitate analysis.

One notable category highlighted in the analysis (see Table 11) is the occurrence of misspelled words. This issue usually arises in generated responses, particularly when it comes to last names, resulting in inaccuracies as illustrated in Table 12. Interestingly, as the number of learnable parameters increases (refer to Table 10), the percentage of misspelled words in Table 11 decreases. Among the methods considered, prompt-tuning employs the least number of parameters and exhibits the highest misspelling rate, while fine-tuning and bottleneck-adapter use more parameters and have the lowest misspelling rate.

Furthermore, the hallucination column in Table 11 also shows a similar trend between the number of learnable parameters and the hallucination rate. This finding suggests that increasing the number of parameters could be an effective approach for reducing both spelling errors and hallucinations in generated responses.

| **Misspelled examples** | | **Partially answered examples** | |
|---|---|---|---|
| **Model answer** | **Correct answer** | **Model answer** | **Correct answer** |
| tim russellt | tim russert | october 2010 | 2010 |
| omar khayyah | omar khayyam | july 1 2005 | 2005 |
| staci keann | staci keanan | 25 years old | 25 |
| thomas lenon | thomas lennon | category 4 | 4 |
| | | ganesha | ganesh |
| | | ryan seacrest and giuliana rancic | ryan seacrest |
| | | 180th meridian or antimeridian | 180th meridian |

Table 12. PEFT misspelled and partially answered examples

In addition to increasing the number of learnable parameters, another source of improvement that could benefit all methods, including fine-tuning, is related to context. In some cases, the context is represented in a table format, which has been shown to be challenging, as pointed out by [NAH$^+$22]. In other cases, the context provided is excessively lengthy, making it impossible to identify the relevant answer as it is simply not present in the input. As a result, $11.79\%$ of the questions are impossible to answer. To address these issues, improvements need to be made at the dataset level.

Table 12 shows that some of the answers generated by the model were partially correct, with additional information added that did not make the answer incorrect. This is a common behavior observed in humans as well. However, it is interesting to note that while we expect our models to perform at a human level, we measure their performance using metrics that are typically used to evaluate machine performance. This discrepancy raises the question of whether we should reconsider the way we measure the performance of these models.

The largest percentage of errors was due to the model selecting the wrong answer from the context, and improving in this area could lead to the greatest improvement. The majority of questions in this category require a detailed understanding of the context and the question being asked. For example, see Figure 11. This example highlights the importance of contextual understanding, as the model failed to distinguish between the start and the inclusion of men's curling in the Olympic Games.

**Question:** When did men's curling start in the olympics?

**Context:** Curling was included in the program of the inaugural Winter Olympic Games in 1924 in Chamonix although the results of that competition were not considered official by the International Olympic Committee until 2006. Curling was a demonstration sport at the 1932 Games, and then again after a lengthy absence in 1988 and 1992. The sport was finally added to the official program for the 1998 Nagano Games.

Figure 11. Example of a complex context. The correct answer is highlighted in green, whereas the incorrect answer (model predicts) is highlighted in red.

To accurately answer questions with complex context, a language model must be able to understand and consider the surrounding context, as different parts of the context may discuss various facets of the same topic. However, it has been observed that models, as shown in Table 8 with ChatGPT permuted context, do not always exhibit sensitivity to these contextual details, resulting in incorrect answers. Improving this area would require addressing the challenging problem of reasoning, for which there is currently no clear solution. Nevertheless, we have noted that fine-tuning and PEFT models can answer questions that require reasoning, as demonstrated in Figure 12. However, this behavior is not consistent, indicating the possibility of chance or other factors.

A recent study [KDR$^+$22] demonstrated that a model's ability to answer fact-based questions is influenced by how many times the associated document has been encountered during pre-training. Thus, in instances where the model exhibited "reasoning" and produced the correct answer, it was likely due to encountering that information more frequently during pre-training.

**Question:** who ran the fastest 40 yard dash in the nfl?

**Context:** Auburn's Bo Jackson claims to have run a 40 - yard dash with a time of 4.13 s. A time of 4.18 run by Jackson within the same week added some support to the legitimacy of the times. Texas Tech's Jakeem Grant was hand - timed by a New Orleans Saints scout as running a 4.10 in 2016, potentially beating Jackson's record. Deion Sanders ran a 4.27 - second 40 - yard dash in 1989. Model needs to find the smallest time out of all discussed in the context.

Figure 12. Example of a model outstanding performance. The correct answer is highlighted in green.

In general, PEFT did not perform better than fine-tuning mainly due to the same issues affecting both approaches, such as limitations in the setup and a lack of fundamental aspects of intelligence like contextual comprehension and reasoning. The simplest approach to improving performance involves resolving setup issues, such as tables within the context, and eliminating answers that contain extra information. However, these improvements are only short-term solutions. For sustainable progress and continued performance improvement, it is essential to address sensitivity issues and incorporate reasoning abilities that consider the intricacies of both the question and context.

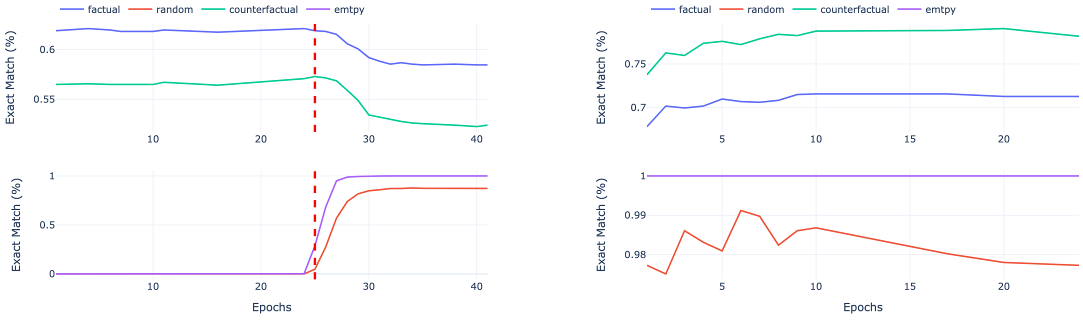
## 6.2 Why Prompt-Tuning Underperformed?

In the results presented earlier, it was observed that datasets with counterfactual augmentations showed significant improvements on counterfactual test split in every method except prompt-tuning, as demonstrated in Table 4. It was initially assumed that this could be due to prompt-tuning having the least number of tunable parameters. Upon examining the number of parameters each method operates over (shown in Table 10), it was indeed found to be the case. Prompt-tuning accounted for only 0.01% relative to fine-tuning, whereas LoRA and Bottleneck-Adapter accounted for 0.32% and 1.72%, respectively.

Figure 13 illustrates prompt-tuning limited capacity in action, as evidenced by the constant performance across all test splits until the 25th epoch. One possible explanation for the constant performance of prompt-tuning is that the number of parameters is too small to significantly influence the model's predictions, resulting in it staying at a baseline level. Interestingly, this baseline level happens to be the performance of the pre-trained model on the same task. The reason for that, however, was shown in Figure 6, where all methods were trained in closed-book settings and evaluated in open-book settings (i.e., the input consists of a question and context). The essence of the Figure is that those models were evaluated on a task they did not see during the training, i.e., answer extraction. However, in Figure 13 answer extraction task has been shown since prompt-tuning was trained on $f$+$cf$+$a$. Therefore, prompt-tuning's inability to effectively steer the pre-trained model and that the performance in both Figures 6 and 13 was the same on the answer extraction task implies that it was pre-trained model baseline performance. It seems that prompt-tuning could be used to measure pre-trained model performance in other tasks too.

After the 25th epoch, certain weight values were achieved through multiple backpropagation steps, resulting in the gradual steering of the pre-trained model from its baseline performance. As a result, performance on factual and counterfactual test splits declined, while performance on answerability test splits (e.g., with empty and random context) improved. These trends are highly anti-correlated, indicating that an increase in one implies a decrease in the other. This suggests that the tasks required by these augmentations are not complementary to each other. However, such a dramatic negative correlation is not observed in the case of LoRA (and other methods). The fact that these tasks are not complementary implies that a model needs to learn both strategies, such as answer extractability and identifying when the question and context are unrelated.

The results support the idea that a performance trade-off exists between the two tasks, and the limited capacity of prompt-tuning is insufficient to learn both tasks, as they are not complementary. Additionally, prompt-tuning's inventors [LARC21] acknowledged that the method struggles to unlearn T5's pre-trained objective (i.e., span corruption). To address this issue, they employed LM Adaptation [RSR+20], which involves continuing T5's self-supervised training with an objective that requires the model to generate a natural text continuation when given a natural text prefix as input. However, we could not use such an adaptation because we wanted to use [NAH+22] as a benchmark to compare our results against. Additionally, the limited capacity of prompt-tuning has revealed the unexpected and previously unknown potential to comprehend the pre-trained model's baseline performance in reading comprehension, and probably not only.



Prompt-Tuning LoRA

Figure 13. Progressive evaluation on all test split of Prompt-Tuning and LoRA trained on a full dataset (i.e., *f*+*cf*+*a*). **NOTE:** The results for other methods can be found in Appendix III Figure 14.

# 7 Conclusion and Future Work

The objective of this thesis was to investigate whether better generalization in reading comprehension could be achieved by using parameter-efficient fine-tuning (PEFT) methods. Several methods, such as LoRA, Bottleneck-Adapter, and prompt-tuning, were utilized and tested against the Natural Question dataset, which was augmented with counterfactual and answerability augmentations to force the model to pay more attention to the input. Our results showed that PEFT methods did not provide a significant performance advantage over fine-tuning due to several factors, including the use of a noisy dataset, a rigid strategy for answer validation, and model insensitivity to details.

Our study also found that while answerability augmentation improved model robustness, counterfactual augmentation led to a reduction in model robustness by inducing simplistic extractor behavior. Prompt-tuning demonstrated that these augmentations were not complementary, with improvements in the answer abstaining task leading to a degradation in the answer extraction task. Interestingly, prompt-tuning revealed that answer abstaining was the easiest task, and established the baseline performance of the pre-trained model on the answer extraction task.

Future research could focus on investigating why using random context technique improves model robustness in challenging contexts. Moreover, prompt-tuning could be explored as a potential approach for understanding the baseline performance of pre-trained models on different tasks. Additionally, testing PEFT methods in multi-task learning could be beneficial due to a lesser rate of catastrophic forgetting. Lastly, creating a training set with challenging contexts to replace random contexts, could potentially improve model robustness even further.

# References

[ASL09] Max Lytvyn Alex Shevchenko and Dmytro Lider, 2009.

[AW13] Dennis L. Byrnes Arthur Wingfield. *The Psychology of Human Memory*. 2013.

[CFC+20] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks, 2020.

[Cra20] Michael Crawshaw. Multi-task learning with deep neural networks: A survey, 2020.

[CTW+21] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models, 2021.

[DQY+23] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, and Zonghan Yang. Extracting training data from large language models, 2023.

[FWI+18] Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

[HGJ+19a] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019.

[HGJ+19b] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019.

[HSW+21] 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, 2021.

[Hug20] Huggingface, 2020.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[KDR+22] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge, 2022.

[KPR+19] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

[LARC21] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

[LPC+22] Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. Entity-based knowledge conflicts in question answering, 2022.

[LSSW22] Anish Lakkapragada, Essam Sleiman, Saimourya Surabhi, and Dennis P. Wall. Mitigating negative transfer in multi-task learning with exponential moving average loss weighting strategies, 2022.

[NAH+22] Ella Neeman, Roee Aharoni, Or Honovich, Leshem Choshen, Idan Szpektor, and Omri Abend. Disentqa: Disentangling parametric and contextual knowledge with counterfactual question answering, 2022.

[Ope22] OpenAI, 2022.

[PRP+20] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, 2020.

[PRR20] Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning, 2020.

[RJL18] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad, 2018.

[RSR+20] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[RZLL16] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[SSMK18] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task, 2018.

[VSP$^+$17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[WMB$^+$20] Johannes Welbl, Pasquale Minervini, Max Bartolo, Pontus Stenetorp, and Sebastian Riedel. Undersensitivity in neural reading comprehension, 2020.

[WYG$^+$22] Chaozheng Wang, Yuanhang Yang, Cuiyun Gao, Yun Peng, Hongyu Zhang, and Michael R. Lyu. No more fine-tuning? an experimental evaluation of prompt tuning in code intelligence. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, nov 2022.

[XZYL20] Y. Xu, X. Zhong, A. J. J. Yepes, and J. H. Lau. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension, 2020.

[ZBH$^+$17] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.

[ZQD$^+$20] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning, 2020.

# Appendix

# I. Evaluation On ChatGPT Context

| Methods | cb | f | f+cf | f+a | f+cf+a |
|---|---|---|---|---|---|
| Fine-Tuning | 32.31 | 79.93 | 80.66 | 21.10 | 20.07 |
| Bottleneck-Adapter | 42.34 | 84.40 | 83.74 | 33.19 | 20.07 |
| LoRA | 40.59 | 87.91 | 85.35 | 14.21 | 32.09 |
| Prompt-Tuning | 65.49 | 69.30 | 69.74 | 5.71 | 2.05 |

Table 13. Percentage of questions for which models (trained on *cb*, *f*, *f+cf*, etc) provided correct factual answer on ChatGPT permuted context.

| Methods | cb | f | f+cf | f+a | f+cf+a |
|---|---|---|---|---|---|
| Fine-Tuning | 6.15 | 77.80 | 86.23 | 18.68 | 26.30 |
| Bottleneck-Adapter | 12.45 | 77.73 | 86.74 | 23.22 | 26.08 |
| LoRA | 11.36 | 81.03 | 87.91 | 15.82 | 25.71 |
| Prompt-Tuning | 68.57 | 69.60 | 69.38 | 23.52 | 24.10 |

Table 14. Percentage of questions for which models (trained on *cb*, *f*, *f+cf*, etc) provided correct counterfactual answer on ChatGPT context with counterfactual answers.

## II. Examples Of ChatGPT Context

**Question:** When did the eagles win the super bowl?

**Context:** Max had always been fascinated by the futuristic world depicted in his favorite movie, so when he woke up on January 1, 2017 and realized he had time traveled 50 years into the future, he was thrilled to see the flying cars and advanced technology. However, he was quickly brought back to reality when he learned that the world was on the brink of destruction due to an impending apocalypse.

**Question:** Who has the world's largest standing army?

**Context:** As the sun rose over the Great Wall of China, Liang anxiously awaited news of his wife's arrival from America to begin their new life together. Despite the distance between them, his love for her burned as bright as the Forbidden City's lanterns on Chinese New Year.

**Question:** When was where have all the flowers gone written?

**Context:** In 1955, Marie took her first steps in her hometown, a place she loved and never wanted to leave. Little did she know, fate had bigger plans for her when she met a man who would take her far away from everything she knew.

**Question:** Cast of law  order special victim unit?

**Context:** Christopher Meloni was a successful actor before he became a baker, but he loved baking even more than he loved acting, and he was happy at last.

**Question:** Who has won the most college football national champions?

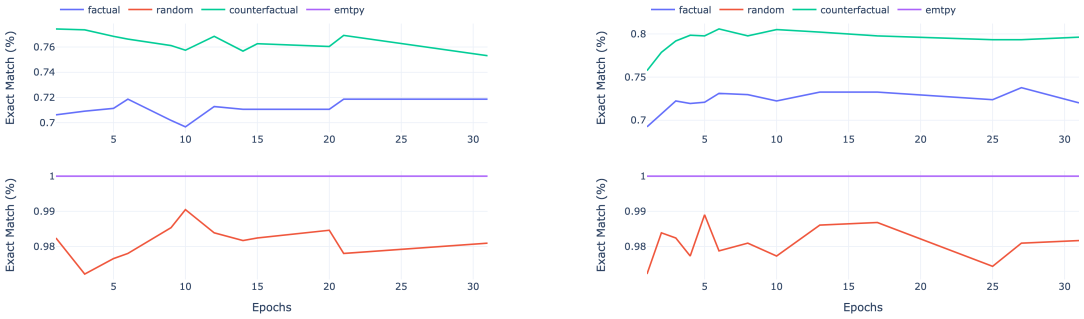
**Context:** As a young boy, Tommy always dreamed of attending Princeton, the prestigious Ivy League university known for producing world-renowned scholars and intellectuals. Despite facing numerous obstacles, he never gave up on his dream and eventually earned his acceptance letter to the school of his dreams.

**Question:** What is the number of total presidential electoral votes?

**Context:** As I peered through the telescope at the night sky, I could not help but notice the mysterious planet with 538 moons orbiting around it. Suddenly, a bright light engulfed me and I found myself transported onto one of those moons, face-to-face with an alien race I could never have imagined.

Table 15. Examples of ChatGPT context

# III. Progressive Evaluations



Fine-Tuning

Bottleneck-Adapter

Figure 14. Progressive evaluation on all test splits of various methods trained on a full dataset (i.e., *f*+*cf*+*a*)

# IV. Licence

## Non-exclusive licence to reproduce thesis and make thesis public

I, **Rustam Abdumalikov**,
(author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   **Parameter-efficient fine-tuning in reading comprehension**,
   (title of thesis)

   supervised by Yova Kementchedjhieva and Kairit Sirts.
   (supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Rustam Abdumalikov
***09/05/2023***