Analyzing Pre-trained and Fine-tuned Language Models

Marius Mosbach

A dissertation submitted towards the degree

Doctor of Engineering (Dr.-Ing.)

of the Faculty of Mathematics and Computer Science

of Saarland University

Saarbrücken, 2023

DAY OF COLLOQUIUM:

18.01.2024

DEAN OF THE FACULTY:

Prof. Dr. Jürgen Steimle

EXAMINATION BOARD:

Chair – Prof. Dr. Vera Demberg

Reviewer – Prof. Dr. Dietrich Klakow

Reviewer – Prof. Dr. Jonathan Berant

Academic Assistant – Dr. Mareike Hartmann

Abstract

The field of natural language processing (NLP) has recently undergone a paradigm shift. Since the introduction of transformer-based language models in 2018, the current generation of natural language processing models continues to demonstrate impressive capabilities on a variety of academic benchmarks and real-world applications. This paradigm shift is based on a simple but general pipeline which consists of pre-training neural language models on large quantities of text, followed by an adaptation step that fine-tunes the pre-trained model to perform a specific NLP task of interest.

Despite the impressive progress on academic benchmarks and the widespread deployment of pre-trained and fine-tuned language models in industry, these models do not come without shortcomings which often have immediate consequences for the robustness and generalization of fine-tuned language models. Moreover, these shortcomings demonstrate that we still lack a fundamental understanding of how and why pre-trained and fine-tuned language models work as well as the individual steps of the pipeline that produce them.

This thesis makes several contributions towards improving our understanding of pre-trained and fine-tuned language models by carrying out a detailed analysis of various parts of the modern NLP pipeline. Our contributions range from analyzing the linguistic knowledge of pre-trained language models and how it is affected by fine-tuning, to a rigorous analysis of the fine-tuning process itself and how the choice of adaptation technique affects the generalization of models. Overall, we provide new insights about previously unexplained phenomena and the capabilities of pre-trained and fine-tuned language models.

Zusammenfassung

Im Bereich der Verarbeitung natürlicher Sprache (NLP) hat sich ein Paradigmenwechsel vollzogen. Seit der Einführung von transformer-basierten Sprachmodellen im Jahr 2018 zeigt die aktuelle Generation neuronaler Sprachverarbeitungsmodelle beeindruckende Fähigkeiten bei einer Vielzahl von akademischen Benchmarks und realen Anwendungen. Dieser Paradigmenwechsel basiert auf einer einfachen, aber allgemeinen Pipeline, die aus dem Vortrainieren von neuronalen Sprachmodellen auf großen Textmengen besteht, gefolgt von einem Anpassungsschritt, der das vortrainierte Modell modifiziert, um eine bestimmte NLP-Aufgabe durchzuführen.

Trotz des beeindruckenden Fortschritts bei akademischen Benchmarks und des weit verbreiteten Einsatzes von vortrainierten und angepassten Sprachmodellen in der Industrie sind diese Modelle nicht ohne Mängel, und oft haben diese Mängel unmittelbare Auswirkungen auf die Robustheit und Generalisierung der Sprachmodelle. Darüber hinaus zeigen sie, dass uns einerseits noch immer ein grundlegendes Verständnis dafür fehlt, wie und warum vortrainierte und angepasste Sprachmodelle funktionieren, andererseits fehlt ein grundlegendes Verständnis der einzelnen Schritte der Pipeline.

Diese Arbeit leistet mehrere Beiträge zur Verbesserung unseres Verständnisses von vortrainierten und angepassten Sprachmodellen, indem sie eine detaillierte Analyse verschiedener Teile der modernen NLP-Pipeline durchführt. Unsere Beiträge reichen von der Analyse des linguistischen Wissens von vortrainierten Sprachmodellen und wie dieses durch die Anpassung beeinflusst wird bis hin zu einer rigorosen Analyse des Anpassungsprozesses selbst und wie die Wahl der Anpassungstechnik die Generalisierung von Modellen beeinflusst, und liefern insgesamt neue Erkenntnisse über bisher unerklärte Phänomene und Fähigkeiten von vortrainierten und angepassten Sprachmodellen.

Acknowledgments

I'd like to thank my PhD advisor Dietrich Klakow for giving me the freedom to follow my curiosity, pursue my own research interests at all times, and for providing a welcoming research environment free of pressure that allowed me to thrive.

I also want to thank my colleagues at LSV: Thomas Trost, Aditya Mogadala, David Howcroft, Xiaoyu Shen, and Michael Hedderich, who were inspiring and helpful during the early days of my PhD; Marimuthu Kalimuthu and Volha Petukhova for being great colleagues and friends throughout the last 5 years; Jesujoba Alabi and David Adelani for being great colleagues and collaborators; Miaoran Zhang for being a great teammate in the B4 project and the best Dietrich simulator; Dawei Zhu for being the best person to share an office with and a friend; Anupama Chingacham, Alexander Blatt, Aravind Krishnan, Dana Ruiter, Paloma Garcia de Herreros, and Julius Steuer for being great colleagues. Thank you also to Nicolas Louis for being a great system administrator and colleague and for forgiving me for sending emails instead of opening tickets.

I am very fortunate to have met many great collaborators and friends along the way. I am particularly grateful to Badr Abdullah, Vagrant Gautam, Maksym Andriushchenko, Shauli Ravfogel, and Yanai Elazar for their friendship and for inspiring me to become a better researcher.

I want to thank my partner Jennifer for her endless support throughout all these years, my brothers Fabian and Marvin, and my parents Bettina and Werner for enabling me to study and their constant support. Bettina, now I might finally get a *real* job ;).

Lastly, I am grateful to the Deutsche Forschungsgemeinschaft for funding my PhD (Project-ID 232722074 – SFB 1102).

Contents

1

Introduction

Contents

1.1 Motivation

Since the introduction of transformer-based pre-trained neural language models in 2018 (Devlin et al., 2019; Y. Liu et al., 2019a), the field of natural language processing (NLP) has witnessed a paradigm shift. Instead of designing and training highly task-specific models from scratch, the current default approach for most

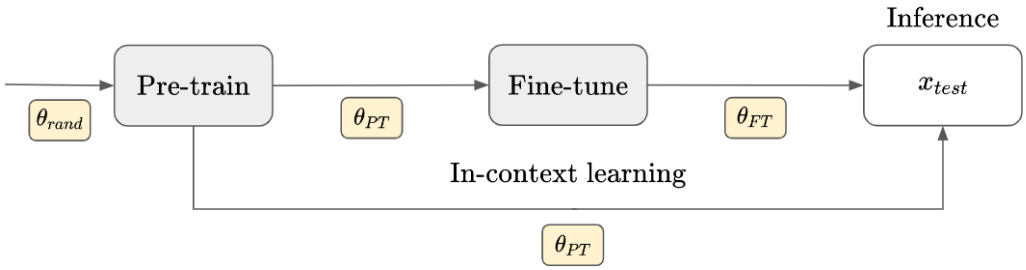


Figure 1.1: The modern NLP pipeline. A randomly initalized model θ_{rand} is trained on large quantities of text, producing a pre-trained language model θ_{PT}. The pre-training step is followed by fine-tuning, which adapts the pre-trained model to a downstream task and results in a task-specific model θ_{FT} which can then be used for inference. Alternatively, we can bypass the fine-tuning step and instead perform task adaptation via in-context learning. This allows us to directly use the pre-trained model for inference.

NLP tasks consists of adapting general-purpose[1] pre-trained language models, a process which typically requires only very few task-specific changes to the model architecture, and therefore allows us to easily apply the same pre-trained model to different tasks. Over the last five years (2019 – 2023), this paradigm shift has led to impressive progress on a large variety of downstream NLP tasks, ranging from traditional computational linguistics tasks such as part-of-speech tagging and more challenging tasks like natural language inference, to text-based dialogue and assistant systems (A. Wang et al., 2018; A. Wang et al., 2019b; OpenAI, 2023, *inter alia*).

At the core of this impressive progress lies a very simple but general pipeline which is illustrated in Figure 1.1. The first step of this pipeline, which we will refer to as the *modern NLP pipeline* for the remainder of this thesis, consists of pre-training a (large) neural language model on large quantities of text using

1 Given the impressive capabilities of the most recent generation of pre-trained models, e.g., GPT-4 (OpenAI, 2023), we believe it is justified to call them general-purpose models and we will do so for the remainder of this thesis.

self-supervised training. Due to the discrepancy between the pre-training objective (e.g., masked language modeling) and the downstream task (e.g., classification), the pre-training step is followed by an adaptation step which fine-tunes the pre-trained model to perform a specific task of interest. During fine-tuning, we either update all of the pre-trained parameters or update only a small fraction of them by leveraging parameter-efficient fine-tuning techniques. In both cases, however, fine-tuning results in a task-specific model which can be used for a single task. An alternative task-adaptation technique which was popularized by the most recent advances in training pre-trained language models (Brown et al., 2020; OpenAI, 2023), allows us to bypass the fine-tuning step by treating the downstream task as a language modeling problem. This process, known as in-context learning (ICL), enables adapting a pre-trained model without updating any parameters[2] and allows even non-expert users to easily leverage pre-trained language models. Recent advancements in in-context learning have led to impressive progress on challenging reasoning benchmarks, surpassing the capabilities of fine-tuned language models by large margins (Wei et al., 2022b); a development which has resulted in unprecedented interest from the general public in the promises and potential risks associated with the use of large language models.

1.2 Research objectives

The previously described pipeline is ubiquitous in modern-day NLP and pre-trained and fine-tuned language models are now dominating research in academia as well as in industry.[3] However, regardless of their impressive capabilities, pre-trained and fine-tuned language models are not without shortcomings and many challenges remain. The contributions made in this thesis center around three shortcomings of pre-trained and fine-tuned language models:

2 We provide additional details about these adaptation methods in Chapter 2.

3 We discuss an extended version of this pipeline which considers two additional steps, namely, instruction fine-tuning and alignment to human preferences, in Chapter 7.

(1) It is well established that fine-tuned language models are often right for the wrong reasons and their good performance on downstream tasks can at least in part be explained by the tendency to pick up spurious correlations during the adaptation process (Jia and Liang, 2017; T. McCoy et al., 2019; Niven and Kao, 2019; Warstadt et al., 2020b, *inter alia*). These results stand in contrast to a large body of evidence that pre-trained language models encode various forms of linguistic and factual knowledge (N. F. Liu et al., 2019; Tenney et al., 2019a; Petroni et al., 2019; Goldberg, 2019; Hewitt and Manning, 2019, *inter alia*). When combined, these findings require taking a nuanced perspective on the interplay between the strong capabilities of language models, as shown by their impressive results on common NLP tasks, and their encoding of linguistic and factual knowledge, and demonstrate the need for investigating the interplay between the linguistic capabilities of pre-trained language models and their downstream performance.

(2) Fine-tuned language models often exhibit striking variation in downstream task performance when performing small changes to the adaptation process such as changing the random seed used for initializing model weights, the order of training examples, or the format of a task instruction (Dodge et al., 2020b; Webson and Pavlick, 2022; Lu et al., 2022). Large variations in fine-tuning performance are undesirable for several reasons such as hindering reproducible research and complicating the distinction between actual improvements due to modeling or algorithmic advances and comparisons against weak baselines. Given the ubiquity of fine-tuned language models, it is therefore critical to gain a better understanding of the fine-tuning algorithms that are commonly applied to adapt language models to downstream tasks.

(3) As mentioned in the previous section, the rapid progress in training ever larger language models has resulted in novel ways to adapt pre-trained language models to downstream tasks by simply instructing them perform a task of interest via in-context learning. Instead of adapting a model via gradient based fine-tuning, in-context learning allows task adaptation via

mere textual interaction and has lead to impressive progress on challenging reasoning benchmarks (Wei et al., 2022a; Wei et al., 2022b). At the same time, there is growing evidence that in-context learning suffers from similar shortcomings as fine-tuning such as the sensitivity to changes in the data order (Min et al., 2022; Lu et al., 2022) and difficulties with generalizing to out-of-distribution inputs (Si et al., 2023). Given the ubiquity of task adaptation via fine-tuning and in-context learning in modern NLP, it is necessary to investigate their respective benefits and downsides and provide a fair comparison of task adaptation approaches.

The shortcomings discussed above demonstrate that in many aspects we still lack a fundamental understanding of how and why pre-trained and fine-tuned language models work so well, as well as of the individual steps of the modern NLP pipeline that produce them. Next, we discuss how this thesis addresses the aforementioned shortcomings and outline our contributions.

1.3 Contributions

The research carried out in this thesis addresses the shortcomings described above and aims to provide a better understanding of the capabilities of pre-trained and fine-tuned language models. To achieve this goal, we critically investigate the individual stages of the modern NLP pipeline and rigorously analyze previously unexplained phenomena and capabilities of pre-trained and fine-tuned language models. Figure 1.2 shows the chapters of this thesis along the modern NLP pipeline together with the shortcomings they address. Our contributions range from analyzing the linguistic knowledge of pre-trained language models (Chapter 3) and how it is affected by fine-tuning (Chapter 4), to a rigorous analysis of the fine-tuning process itself (Chapter 5), and how the choice of adaptation technique affects the generalization of models during inference (Chapter 6).

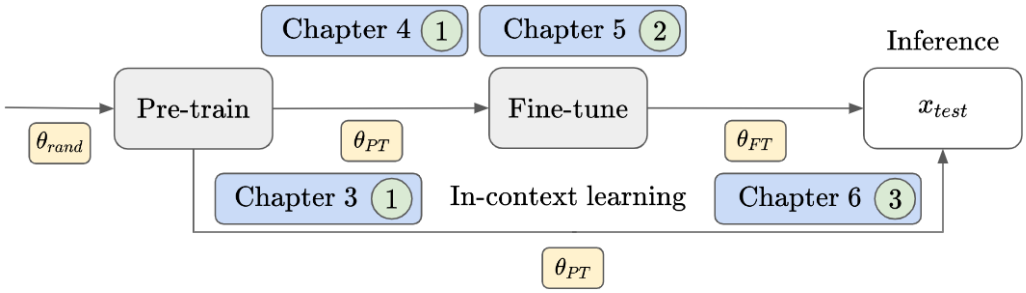


Figure 1.2: Our contributions along the modern NLP pipeline. Each chapter addresses one of the shortcomings posed in §1.2.

Concerning the pre-training stage, we study whether pre-trained language models encode grammatical knowledge about relative clauses in English. We demonstrate that viewing probing results in isolation can lead to overestimating the linguistic capabilities of a model which highlights the importance of building up claims about the linguistic knowledge encoded in pre-trained language models beyond mere probing-based evaluations. We then investigate how fine-tuning changes the linguistic knowledge encoded by a model and find that while fine-tuning can indeed improve probing performance this is an artifact of the probing setup rather than the result of encoding more linguistic knowledge in the model.

Next, we study the fine-tuning process itself and answer an open question about the instability of fine-tuning and hypothesized causes. We show that fine-tuning instability is caused by optimization difficulties that lead to vanishing gradients and hence, poor generalization. Based on our analysis, we suggest a simple baseline that makes fine-tuning pre-trained language models significantly more stable while maintaining or even improving overall performance.

Lastly, we study the generalization behavior of task-adaptation via fine-tuning and in-context learning and show that fine-tuned language models can generalize well out-of-domain. We find that in fact both adaptation approaches generalize similarly; both exhibit large differences in performance and crucially depend on model size. Our findings highlight that robust task adaptation remains an open challenge. Below, we provide a more detailed summary of our core contributions.

1.3.1 Probing pre-trained models for linguistic knowledge

Chapter 3 focuses on deepening our understanding of the capabilities of pre-trained language models by analyzing the linguistic knowledge encoded in their representations. In our analysis we focus on relative clauses (in American English) as a complex linguistic phenomenon needing contextual information and antecedent identification to be resolved. In contrast to previous work, we do not analyze the linguistic knowledge of the model by only training probing classifiers or by evaluating the (token) predictions of the language model in isolation. Instead, we combine these types of evaluations to get a more fine-grained understanding of how the linguistic knowledge encoded by a model is reflected in its predictions.

Our probing results show that pre-trained language models indeed encode linguistic knowledge about grammaticality, achieving high classification performance. However, when evaluating the probing classifiers on more challenging diagnostic cases and evaluating the token predictions of the models, we find pronounced model-specific weaknesses especially on semantic knowledge, which strongly impacts performance. Our fine-grained analysis highlights the importance of building up claims about model performance and the linguistic knowledge of pre-trained language models beyond purely probing-based evaluations.

The content presented in Chapter 3 is based on:

> **Mosbach, Marius**, Stefania Degaetano-Ortlieb, Marie-Pauline Krielke, Badr M. Abdullah, and Dietrich Klakow (2020). "A Closer Look at Linguistic Knowledge in Masked Language Models: The Case of Relative Clauses in American English." In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 771–787. URL: https://aclanthology.org/2020.coling-main.67.

1.3.2 On the interplay between fine-tuning and probing

Our next contribution, which is presented in Chapter 4, builds on our findings about the linguistic knowledge of pre-trained language models and investigates the connection between high performance on downstream tasks and the linguistic information encoded by a model. We investigate the hypothesis that the strong capabilities of fine-tuned language models can at least implicitly be attributed to the vast amount of linguistic knowledge which they encode (Pruksachatkun et al., 2020a). To this end, we study three different pre-trained language models and investigate through sentence-level probing how fine-tuning them on downstream tasks affects the linguistic information encoded in their representations.

We find that while fine-tuning indeed changes the representations of pre-trained models – and these changes are typically larger for higher layers – fine-tuning has a positive effect on probing accuracy only in very few cases, and this effect mainly depends on the pooling strategy used to compute sentence representations for probing. Our findings demonstrate that there is no straightforward causal relationship between the linguistic information encoded by a model and its performance on NLP downstream tasks, which calls for a careful interpretation of changes in probing performance as a result of fine-tuning.

The content presented in Chapter 4 is based on:

> **Mosbach, Marius**, Anna Khokhlova, Michael A. Hedderich, and Dietrich Klakow (2020). "On the Interplay Between Fine-tuning and Sentence-level Probing for Linguistic Knowledge in Pre-trained Transformers." In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, pp. 2502–2516. URL: https://aclanthology.org/2020.findings-emnlp.227.

1.3.3 Investigating fine-tuning stability

In contrast to our previous contributions which focused on analyzing the representations of pre-trained and fine-tuned language models, in Chapter 5 we focus on analyzing the fine-tuning process itself. Previous work (Devlin et al., 2019; Lee et al., 2020; Dodge et al., 2020a) has observed large differences in downstream task performance simply when fine-tuning models with different random seeds. The two most popular hypotheses proposed for this intriguing behaviour are: catastrophic forgetting and the small size of the fine-tuning datasets. Motivated by these anecdotal observations, we perform a rigorous investigation of fine-tuning instability in order to determine its root cause.

We analyze three different pre-trained language models fine-tuned on widely used datasets and show that the hypotheses commonly stated in previous work fail to explain the observed phenomena. Instead the observed instability is caused by optimization difficulties during fine-tuning that lead to vanishing gradients. Based on our analysis, we present a simple but strong baseline approach for fine-tuning that makes fine-tuning pre-trained language models significantly more stable than previously proposed approaches while at the same time maintaining or even improving performance.

The content presented in Chapter 5 is based on:

> **Mosbach, Marius**, Maksym Andriushchenko, and Dietrich Klakow (2021). "On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines." In: *International Conference on Learning Representations: ICLR 2021*. Online. URL: https://openreview.net/forum?id=nzpLWnVAyah.

1.3.4 Investigating generalization of task-adapted models

Our final contribution is concerned with the last step of the NLP pipeline, namely, inference. We compare the generalization behavior of task-adaptation via few-shot fine-tuning and in-context learning, which has recently gained popularity over fine-tuning due to its simplicity and strong performance on challenging reasoning tasks. In addition, recent work has argued that in-context learning leads to improved out-of-domain generalization compared to fine-tuning, which is known to pick up on spurious correlations during training (Si et al., 2023). In Chapter 6, we investigate whether the observed weaker out-of-domain generalization of fine-tuned models is an inherent property of fine-tuning and provide a fair comparison between the generalization of fine-tuning and in-context learning.

Our findings demonstrate that fine-tuned language models *can* generalize well both in and out-of-domain. In fact, we find that the generalization of fine-tuning and in-context learning is highly similar as both approaches exhibit large variation in performance and strongly depend on properties such as model size and the number of examples. Our findings provide evidence that the poor of out-of-domain generalization of fine-tuned models observed in previous work is not a fundamental flaw of fine-tuning but rather a result of their experimental setup, and highlight that truly robust task adaptation remains a challenge.

The content presented in Chapter 6 is based on:

> **Mosbach, Marius**, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar (July 2023). "Few-shot Fine-tuning vs. In-context Learning: A Fair Comparison and Evaluation." In: *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto, Canada: Association for Computational Linguistics, pp. 12284–12314. URL: https://aclanthology.org/2023.findings-acl.779.

1.4 Additional publications

Beyond the contributions discussed in this thesis, I have further contributed as a co-author to the following publications and pre-prints during my PhD:

Elazar, Yanai, Nora Kassner, Shauli Ravfogel, Amir Feder, Abhilasha Ravichander, **Mosbach, Marius**, Yonatan Belinkov, Hinrich Schütze, and Yoav Goldberg (2023). "Measuring Causal Effects of Data Statistics on Language Model's 'Factual' Predictions." In: URL: https://arxiv.org/abs/2207.14251.

Zhu, Dawei, Xiaoyu Shen, **Mosbach, Marius**, Andreas Stephan, and Dietrich Klakow (July 2023). "Weaker Than You Think: A Critical Look at Weakly Supervised Learning." In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Best paper award 🏆. Toronto, Canada: Association for Computational Linguistics, pp. 14229–14253. URL: https://aclanthology.org/2023.acl-long.796.

Alabi, Jesujoba O., David Ifeoluwa Adelani, **Mosbach, Marius**, and Dietrich Klakow (2022). "Adapting Pre-trained Language Models to African Languages via Multilingual Adaptive Fine-Tuning." In: *Proceedings of the 29th International Conference on Computational Linguistics*. Best paper award 🏆. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, pp. 4336–4349. URL: https://aclanthology.org/2022.coling-1.382.

Deshpande, Awantee, Dana Ruiter, **Mosbach, Marius**, and Dietrich Klakow (2022). "StereoKG: Data-Driven Knowledge Graph Construction For Cultural Knowledge and Stereotypes." In: *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*. Seattle, Washington (Hybrid): Association for Computational Linguistics, pp. 67–78. DOI: 10.18653/v1/2022.woah-1.7. URL: https://aclanthology.org/2022.woah-1.7.

Zhang, Miaoran, **Mosbach, Marius**, David Adelani, Michael Hedderich, and Dietrich Klakow (2022). "MCSE: Multimodal Contrastive Learning of Sentence

Embeddings." In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, pp. 5959–5969. DOI: 10.18653/v1/2022.naacl-main.436. URL: https://aclanthology.org/2022.naacl-main.436.

Zouhar, Vilém, **Mosbach, Marius**, Debanjali Biswas, and Dietrich Klakow (2022a). "Artefact Retrieval: Overview of NLP Models with Knowledge Base Access." In: URL: https://arxiv.org/abs/2201.09651.

Zouhar, Vilém, **Mosbach, Marius**, and Dietrich Klakow (2022b). *Fusing Sentence Embeddings Into LSTM-based Autoregressive Language Models*. URL: https://arxiv.org/abs/2208.02402.

Zouhar, Vilém, **Mosbach, Marius**, Miaoran Zhang, and Dietrich Klakow (2022c). "Knowledge Base Index Compression via Dimensionality and Precision Reduction." In: *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*. Dublin, Ireland and Online: Association for Computational Linguistics, pp. 41–53. DOI: 10.18653/v1/2022.spanlp-1.5. URL: https://aclanthology.org/2022.spanlp-1.5.

Mosbach, Marius, Irina Stenger, Tania Avgustinova, Bernd Möbius, and Dietrich Klakow (Sept. 2021). "incom.py 2.0 - Calculating Linguistic Distances and Asymmetries in Auditory Perception of Closely Related Languages." In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. Held Online: INCOMA Ltd., pp. 968–977. URL: https://aclanthology.org/2021.ranlp-1.110.

Abdullah, Badr M., **Mosbach, Marius**, Iuliia Zaitova, Bernd Möbius, and Dietrich Klakow (2021). "Do Acoustic Word Embeddings Capture Phonological Similarity? An Empirical Study." In: *Proc. Interspeech 2021*, pp. 4194–4198. DOI: 10.21437/Interspeech.2021-678.

Jágrová, Klára, Michael Hedderich, **Mosbach, Marius**, Tania Avgustinova, and Dietrich Klakow (2021). "On the Correlation of Context-Aware Language Models With the Intelligibility of Polish Target Words to Czech Readers." In: *Frontiers in Psychology* 12. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2021.662277. URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2021.662277.

Kalimuthu, Marimuthu, Aditya Mogadala, **Mosbach, Marius**, and Dietrich Klakow (2021). "Fusion Models for Improved Image Captioning." In: *Pattern Recognition. ICPR International Workshops and Challenges*. Ed. by Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani. Cham: Springer International Publishing, pp. 381–395. ISBN: 978-3-030-68780-9. URL: https://link.springer.com/chapter/10.1007/978-3-030-68780-9_32.

Saveleva, Ekaterina, Volha Petukhova, **Mosbach, Marius**, and Dietrich Klakow (June 2021a). "Discourse-based Argument Segmentation and Annotation." In: *Proceedings of the 17th Joint ACL - ISO Workshop on Interoperable Semantic Annotation*. Groningen, The Netherlands (online): Association for Computational Linguistics, pp. 41–53. URL: https://aclanthology.org/2021.isa-1.5.

– (2021b). "Graph-based Argument Quality Assessment." In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. Held Online: INCOMA Ltd., pp. 1268–1280. URL: https://aclanthology.org/2021.ranlp-1.143.

Grosse, Kathrin, Thomas A. Trost, **Mosbach, Marius**, Michael Backes, and Dietrich Klakow (2020). "On the Security Relevance of Initial Weights in Deep Neural Networks." In: *Artificial Neural Networks and Machine Learning – ICANN 2020*. Ed. by Igor Farkaš, Paolo Masulli, and Stefan Wermter. Cham: Springer International Publishing, pp. 3–14. ISBN: 978-3-030-61609-0. URL: https://link.springer.com/chapter/10.1007/978-3-030-61609-0_1.

Mosbach, Marius, Maksym Andriushchenko, Thomas Trost, Matthias Hein, and Dietrich Klakow (2019a). "Logit Pairing Methods Can Fool Gradient-Based Attacks." In: URL: https://arxiv.org/abs/1810.12042.

Mosbach, Marius, Irina Stenger, Tania Avgustinova, and Dietrich Klakow (Sept. 2019b). "incom.py - A Toolbox for Calculating Linguistic Distances and Asymmetries between Related Languages." In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. Varna, Bulgaria: INCOMA Ltd., pp. 810–818. DOI: 10.26615/978-954-452-056-4_094. URL: https://aclanthology.org/R19-1094.

Bizzoni, Yuri, **Mosbach, Marius**, Dietrich Klakow, and Stefania Degaetano-Ortlieb (2019). "Some steps towards the generation of diachronic WordNets." In: *Proceedings of the 22nd Nordic Conference on Computational Linguistics*. Turku, Finland: Linköping University Electronic Press, pp. 55–64. URL: https://aclanthology.org/W19-6106.

1.5 Outline

The rest of this thesis is structured as follows: in Chapter 2, we introduce the necessary background on the basics of machine learning and provide a formal introduction to pre-training, fine-tuning, and probing of language models. This is is followed by Chapters 3 to 6, which constitute the main research contributions of this thesis. We end with **??**, discussing conclusions and implications for future work. Supplementary material is provided in Appendices A to D.

2

Background

Contents

This chapter introduces the definitions, concepts, and techniques central to the research presented in this thesis. We begin by introducing fundamental machine learning concepts in Section 2.2. Section 2.3 introduces different types of pre-trained language models, which are the central object of study in this thesis. Next, Section 2.4 introduces the concept of language model adaptation and presents the most popular techniques for adapting pre-trained language models to

downstream tasks. Lastly, in Section 2.5, we briefly discuss probing, a commonly used technique to analyze the internal representations of language models.

2.1 Notation

Scalars and vectors We use plain letters to denote scalars: $s \in \mathbb{R}$ and bold lowercase letters for vectors: $\mathbf{v} \in \mathbb{R}^d$. If not stated otherwise, vectors are assumed to be column-vectors.

Sets and alphabets Sets are denoted by $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ where $|\mathcal{S}|$ is the number of elements in the set. We use Greek capital letters, e.g., Σ to denote an *alphabet* which is a finite non-empty set. The elements of an alphabet are called symbols and we use lowercase letters to represent them. Given an alphabet Σ, a *string* is a finite sequence of letters from the alphabet, e.g. $\mathbf{s} = s_1 \cdots s_T$ with $s_i \in \Sigma \; \forall i = 1, \ldots, T$.

Parametric models We use $f_{\boldsymbol{\theta}} : \mathcal{X} \mapsto \mathcal{Y}$ to denote a function with input space $\mathcal{X}$ and output space $\mathcal{Y}$, parameterized by $\boldsymbol{\theta}$ and let $\mathbb{H} = \{f_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}$ denote the set of all functions – also known as a *hypothesis class* – that can be represented by functions parameterized by $\boldsymbol{\theta}$. We use the notation $p_{\boldsymbol{\theta}}$ when working with conditional probability distributions $p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$ parameterized by $\boldsymbol{\theta}$. We refer to $f_{\boldsymbol{\theta}}$ and $p_{\boldsymbol{\theta}}$ as models and for the scope of this thesis, $\boldsymbol{\theta}$ are typically the parameters of a neural network model and Θ is the set of all possible weight matrices of this model.

Data Let $P(\mathbf{x}, y)$ denote a (unknown) joint probability distribution over $\mathcal{X}$ and $\mathcal{Y}$, and $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ a set of examples drawn i.i.d from $P(\mathbf{x}, y)$. For the scope of this thesis, the input space $\mathcal{X}$ is typically the Euclidean space $\mathbb{R}^d$ and the output space $\mathcal{Y}$ some categorical set $\{1, \ldots, K\}$ of K classes. Similarly, we define $P(\mathbf{x})$ as the (unknown) marginal distribution over $\mathcal{X}$ and let $P(y \mid \mathbf{x})$

denote the (unknown) conditional probability distribution that gives a distribution $\mathbf{x}_i$ over possible outputs y_i.

2.2 Machine learning basics

Having established our basic notion we now turn to introducing some fundamental machine learning concepts that will be relevant in later chapters of this thesis.

2.2.1 Supervised learning

Supervised learning is one of the most widely used settings to train machine learning models. In this setting we are given a set of training examples $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with inputs $\mathbf{x}_i$ and labels y_i. We assume that the training examples are drawn i.i.d. from a probability measure $\mathcal{P}$ on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space. The statistical learning perspective on supervised learning further assumes that there exists a conditional probability distribution $P(y \mid \mathbf{x})$ which for each input $\mathbf{x}_i$ gives a distribution over possible outputs y_i. The goal of supervised learning is to find a function $f_{\boldsymbol{\theta}} : \mathcal{X} \mapsto \mathcal{Y}$ from a hypothesis class $\mathbb{H} = \{f_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \boldsymbol{\theta}\}$ such that $f_{\boldsymbol{\theta}}(\mathbf{x}_i) = \hat{y}_i \approx y_i \; \forall i = 1, \ldots, N$. In other words, the supervised learning problem consists of finding a function $f_{\boldsymbol{\theta}}$ that predicts the given data well. For simplicity, below we use a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ that compares the model prediction to the ground-truth label.

2.2.1.1 Empirical risk minimization

Using the terminology established above we can now introduce the empirical risk minimization (ERM) perspective on supervised learning. Assuming we are given a loss function $\mathcal{L}$, a model $f_{\boldsymbol{\theta}}$, and we know the probability measure $\mathcal{P}$, i.e. the data generating distribution, we define the **risk** associated with $f_{\boldsymbol{\theta}}$ as:

$$R(f_{\boldsymbol{\theta}}) = \mathbb{E}_{(\mathbf{x}_i, y_i)\sim\mathcal{P}}[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)] \ . \tag{2.1}$$

Under the assumptions above, solving the supervised learning problem becomes an optimization problem which seeks to find a function $f^{\star}_{\boldsymbol{\theta}}$ that minimizes the true risk:

$$f^{\star}_{\boldsymbol{\theta}} = \arg\min_{f_{\boldsymbol{\theta}}\in\mathbb{H}} \ R(f_{\boldsymbol{\theta}}) \ . \tag{2.2}$$

In real-word scenarios however, the data generating distribution is almost always unknown and we only have access to a set of training examples $D = \{(\mathbf{x}_1, y), \ldots, (\mathbf{x}_N, y_n)\}$ drawn i.i.d. from $\mathcal{P}$. Given D, we define the **empirical risk**:

$$R_{\texttt{emp}} = \mathbb{E}_{(\mathbf{x}_i, y_i)\sim D}[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)] = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i) \ , \tag{2.3}$$

and instead of minimizing the true risk, we seek to find a solution to the **empirical risk minimization (ERM) problem**:

$$\hat{f}_{\boldsymbol{\theta}} = \arg\min_{f_{\boldsymbol{\theta}}\in\mathbb{H}} \ R_{\texttt{emp}}(f_{\boldsymbol{\theta}}) \ . \tag{2.4}$$

2.2.1.2 Maximum likelihood estimation

We can alternatively approach the supervised learning problem from a Bayesian perspective and establishing a connection between empirical risk minimization and maximum likelihood estimation (MLE).

Given a training set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ drawn i.i.d. from $\mathcal{P}$ with $X = \{\mathbf{x_1}, \ldots, \mathbf{x}_n\}$ and $Y = \{y_1, \ldots, y_n\}$. The likelihood of the dataset under a specific model f_θ is defined as $p(Y \mid X, f_{\boldsymbol{\theta}})$, which factorizes into

$$p(Y \mid X, f_{\boldsymbol{\theta}}) = \prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) \ , \tag{2.5}$$

due to the i.i.d. assumption. Note that $p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}})$ is still a function of $\boldsymbol{\theta}$.

To find parameters $\boldsymbol{\theta}$ that fit the given data well, we define the **maximum likelihood** solution as:

$$f_{\texttt{ML}} = \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} p(Y \mid X, f_{\boldsymbol{\theta}}) = \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) \ , \tag{2.6}$$

Instead of directly optimizing the product of likelihoods, we instead take the logarithm of the likelihood and normalize by n, as applying a monotonically increasing function and re-scaling does not change the arg max, i.e.,

$$\begin{aligned} f_{\texttt{ML}} &= \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} p(Y \mid X, f_{\boldsymbol{\theta}}) && (2.7)\\ &= \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \ \log \prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) && (2.8)\\ &= \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \sum_{i=1}^{n} \log p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) && (2.9)\\ &= \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) \ . && (2.10) \end{aligned}$$

Finally, we convert the maximization problem into a minimization problem

$$\begin{aligned} f_{\texttt{ML}} &= \arg\max_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) && (2.11)\\ &= \arg\min_{f_{\boldsymbol{\theta}} \in \mathbb{H}} \frac{1}{n} \sum_{i=1}^{n} -\log p(y_i \mid \mathbf{x}_i, f_{\boldsymbol{\theta}}) \ . && (2.12) \end{aligned}$$

Comparing 2.12 to 2.4 we observe that the MLE and ERM solutions agree when using the loss function $\mathcal{L}(f(\mathbf{x}), y) = -\log\ (y \mid \mathbf{x}, f_{\boldsymbol{\theta}})$.

2.2.1.3 In-domain generalization

Recall from §2.2.1.1 that the model we are interested in should ideally have a low risk on the the entire distribution $\mathcal{P}$ and not just the training data. Stated differently, we seek to find a model that *generalizes* beyond the training data used to fit the model. Naively solving for the model that minimizes the empirical risk (or alternatively optimizing for the maximum likelihood solution) can lead to *overfitting* if our model perfectly fits the training data. This can happen if the chosen hypothesis class is too complex or a model perfectly fits the noise in the

training data. To detect overfitting to the empirical distribution, it is common to split the available data into a training and test set, where the purpose of the test set is to estimate the risk on the true distribution.[1] Crucially, we assume that both the training and the test data are sampled i.i.d. from the data generating process $\mathcal{P}$ and thus refer to the risk on the test data as the **in-domain generalization** of the model.

2.2.1.4 Regularized empirical risk minimization

To prevent against overfitting we can bias the search for a minimizer by restricting the set of possible models in the hypothesis space. We do this by employing *regularization*. Adding a regularization functional to the empirical risk minimization objective balances the complexity of the model and the fit of the training data. We define the **regularized empirical risk** as follows:

$$R_{\texttt{remp}} = \mathbb{E}_{(\mathbf{x}_i, y_i)\sim D}[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i) + \Omega(f_{\boldsymbol{\theta}})] = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f_{\boldsymbol{\theta}}(x_i), y_i) + \Omega(f_{\boldsymbol{\theta}}) \ , \quad (2.13)$$

where the regularizer $\Omega : \mathbb{H} \mapsto \mathbb{R}_+$ measures the complexity of a model. The **regularized empirical risk minimization (RERM)** is then defined as:

$$\hat{f}_{\boldsymbol{\theta}} = \underset{f_{\boldsymbol{\theta}} \in \mathbb{H}}{\arg\min} \ R_{\texttt{remp}}(f_{\boldsymbol{\theta}}) \ . \quad (2.14)$$

In practice, when fine-tuning pre-trained language models we commonly use **weight decay** as regularization which penalizes the l_2 norm of the parameters $\boldsymbol{\theta}$.

2.2.2 Generative learning

The supervised learning setting introduced above is an instance of *discriminative learning* where the goal is to estimate the conditional probability $P(y \mid \mathbf{x})$. An alternative to discriminative learning is *generative learning* where the goal is to learn the underlying distribution $P(\mathbf{x})$ of the data. As we will be dealing with

1 In practice, we often have access to an additional validation set which is used to optimize the hyperparameters of a model or the learning algorithm.

language modeling later in this section, we focus on the setting where P is a distribution over natural language strings.

Maximum likelihood estimation Similar to the discriminative supervised learning setting, we can also adopt a Bayesian perspective on generative learning and formulate the learning problem in terms of MLE. Let $D = \{\mathbf{x}^i\}_{i=1}^n$ be a training set of i.i.d. strings drawn from the data generating distribution $P(\mathbf{x})$ and $p_{\boldsymbol{\theta}}$ be a parametric model. We define the likelihood of D under the model $p_{\boldsymbol{\theta}}$ as the joint probability of all $\mathbf{x}^i$:

$$L(D, p_{\boldsymbol{\theta}}) = \prod_{i=1}^{n} p_{\boldsymbol{\theta}}(\mathbf{x}^i) \ , \tag{2.15}$$

We can rewrite the likelihood by factorizing the joint distribution as a product of conditionals and applying the logarithm

$$\prod_{i=1}^{n} p_{\boldsymbol{\theta}}(\mathbf{x}^i) = \sum_{i=1}^{n} \sum_{t=1}^{T} \log p_{\boldsymbol{\theta}}(\mathbf{x}_t^i \mid \mathbf{x}_{<t}^i) \quad , \tag{2.16}$$

and choose the optimal parameters by maximizing the likelihood of observing the data D under that model:

$$\theta_{MLE} = \arg\max_{p_{\boldsymbol{\theta}} \in \mathbb{H}} L(D, p_{\boldsymbol{\theta}}) \tag{2.17}$$

$$= \arg\max_{p_{\boldsymbol{\theta}} \in \mathbb{H}} \sum_{i=1}^{n} \sum_{t=1}^{T} \log p_{\boldsymbol{\theta}}(\mathbf{x}_t^i \mid \mathbf{x}_{<t}^i) \quad . \tag{2.18}$$

It can be shown that the maximum likelihood solution is the same as the the solution that minimizes the cross-entropy

$$H(\tilde{p}, p_{\boldsymbol{\theta}}) = -\sum_{\mathbf{x} \in D} \tilde{p}(\mathbf{x}) \log p_{\boldsymbol{\theta}}(\mathbf{x}) \tag{2.19}$$

between the empirical distribution $\tilde{p}$ defined by the dataset and the modeled distribution $p_{\boldsymbol{\theta}}$ (Goodfellow et al., 2016), which is precicely the modeling approach we take when training language models in practice.

The pseudo-(log)-likelihood Rather than factorizing the likelihood as a product of conditionals as shown above, we can also approximate the likelihood using the pseudo-(log-)likelihood (Besag, 1975). The pseudo-(log-)likelihood is motivated by the observation that the factorization of the likelihood in 2.16 only considers prior symbols $\mathbf{x}_{<t}$ when estimating the probability of the target symbol $\mathbf{x}_t$. The pseudo-(log-)likelihood instead considers both sides of the target symbol:

$$L_{\texttt{pseudo}}(D, p_{\boldsymbol{\theta}}) = \sum_{i=1}^{n} \sum_{t=1}^{T} \log\, p_{\boldsymbol{\theta}}(x_t^{(i)} \mid \mathbf{x}_{<t}^{(i)}, \mathbf{x}_{>t}^{(i)}) \tag{2.20}$$

$$\approx L(D, p_{\boldsymbol{\theta}}) \ . \tag{2.21}$$

Models that use a variant of the pseudo-(log-)likelihood as a loss function are commonly referred to as masked language models and we discuss them in more detail in §2.3.2.

2.2.3 Stochastic gradient descent

Unlike for simple models, such as logistic regression, where we can obtain closed-form solutions for maximum likelihood estimation (Goodfellow et al., 2016), for functions parameterized by (deep) neural networks the maximum likelihood optimization problem is non-convex, and closed-form solutions are generally not available. To address this, we turn to numerical optimization via gradient descent. For simplicity, we introduce optimization via (stochastic) gradient descent in the context of supervised learning, i.e., we use optimization to solve the minimization problem in Eq. (2.4).

In absence of a closed form solution, the objective of stochastic gradient descent is to iteratively search the parameter space and find the set of parameters that yields the lowest value for the loss function associated with the training data. Stochastic gradient descent is an iterative update process. Starting with initial parameters $\boldsymbol{\theta}_0$, the algorithm proceeds by updating these parameters in successive steps, moving towards the optimal set of parameters. To determine the

direction of this update, we employ the gradient of the loss function with respect to the current parameters $\boldsymbol{\theta}_t$. The basic update formula is given in Eq. (2.22) and consists of three components:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta \cdot \nabla_{\boldsymbol{\theta}_t} \mathcal{L}(\boldsymbol{\theta}_t) \ , \tag{2.22}$$

the current parameters $\boldsymbol{\theta}_{t-1}$, the gradient $\nabla_{\boldsymbol{\theta}_t} \mathcal{L}(\boldsymbol{\theta}_t)$, which provides information about the slope and direction of steepest ascent for the loss function at that particular point in the parameter space[2], and the learning rate η, which controls the magnitude of the step we take during each update. The vanilla version of gradient descent (also known as batch gradient descent) uses the entire training set when computing gradients. In contrast, mini-batch stochastic gradient descent approximates the true gradient by using the gradient computed for a mini-batch of examples that are randomly sampled from the training set:

$$\nabla_{\boldsymbol{\theta}_t} \mathcal{L}(\boldsymbol{\theta}_t) \approx \mathcal{L}(\{(\mathbf{x}_i, y_i)\}_{i=1}^{m}; \boldsymbol{\theta}_t) \tag{2.23}$$

$$= \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}_t} \mathcal{L}(\mathbf{x_i}, y_i, \boldsymbol{\theta}_t) \ . \tag{2.24}$$

We provide the pseudo-code for mini-batch stochastic gradient descent in Algorithm 1.

2.2.3.1 Adam optimizer

Adam is a modified version of mini-batch stochastic gradient descent proposed by D. Kingma and Ba (2015) that differs in several important aspects. Adam computes adaptive learning rates for each parameter by calculating moving averages of the gradient and its squared values. Additionally, Adam uses bias correction to address the fact that the moving averages of the gradients and squared gradients are initialized at zero, making them biased estimates at the beginning of training. The pseudo-code for Adam is given in Algorithm 2.

Adam is one of the best performing and most widely used optimization algorithms in the modern machine learning and NLP literature (Schmidt et al.,

2 Since our goal is to minimize the loss, we move in the opposite direction of the gradient.

Algorithm 1 Stochastic gradient descent

Require: Initial parameters $\boldsymbol{\theta}_0$, learning rate η, number of epochs E, batch size B

Require: Training dataset with input-label pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$

1: Initialize $t \leftarrow 0$ (Update steps)
2: **for** $e = 1$ to E **do**
3: Shuffle the training dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ randomly
4: **for** $k = 1$ to N/B **do**
5: Sample a mini-batch of data points $\{(\mathbf{x}_j, y_j)\}_{j=(k-1)B+1}^{kB}$
6: Compute the gradient of the loss function $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\{(x_j, y_j)\}; \boldsymbol{\theta}_t)$
7: Update parameters: $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\{(x_j, y_j)\}; \boldsymbol{\theta}_t)$
8: $t \leftarrow t + 1$
9: **end for**
10: **end for**
11: **Output:** Learned parameters $\boldsymbol{\theta}_t$

2021; Kaddour et al., 2023). For some of the experiments in later chapters of this thesis we will rely on a slightly modified version of the algorithm that adds an additional term for weight decay regularization (Loshchilov and Hutter, 2019). Specifically, the parameter update in line 13 of Algorithm 2 is replaced by:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta\left(\frac{1}{\sqrt{\hat{v}} + \epsilon} \cdot \hat{\mathbf{m}} + \lambda \boldsymbol{\theta}_t\right) \tag{2.25}$$

2.3 Pre-trained language models

While early work on fine-tuning focused on recurrent neural networks (Howard and Ruder, 2018; Peters et al., 2018), transformer-based (Vaswani et al., 2017) models were soon shown to be superior to recurrent neural networks in terms of their training efficiency, performance, and scalability (Devlin et al., 2019; Kaplan et al., 2020), and are now the dominant model architecture in NLP.

In this section we introduce two different types of transformer-based neural language models and define the objective functions used for training them.

Algorithm 2 Adam optimizer

Require: Initial parameters $\boldsymbol{\theta}_0$, learning rate η, number of epochs E, batch size B

Require: Training dataset with input-label pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$

Require: Exponential decay rates β_1, $\beta_2 \in [0, 1)$

Initialize $\mathbf{m} \leftarrow 0$ (Initial 1st moment vector)

Initialize $\mathbf{v} \leftarrow 0$ (Initial 2nd moment vector)

Initialize $t \leftarrow 0$ (Update steps)

for $e = 1$ to E **do**

  Shuffle the training dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ randomly

  for $k = 1$ to N/B **do**

    Sample a mini-batch of data points $\{(\mathbf{x}_j, y_j)\}_{j=(k-1)B+1}^{kB}$

    Compute the gradient of the loss function $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\{(x_j, y_j)\}; \boldsymbol{\theta}_t)$

    $\mathbf{m} \leftarrow \beta_1 \cdot m + (1 - \beta_1) \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\{(x_j, y_j)\}; \boldsymbol{\theta}_t)$ (Update 1st moment estimate)

    $\mathbf{v} \leftarrow \beta_2 \cdot v + (1 - \beta_2) \cdot (\nabla_{\boldsymbol{\theta}} \mathcal{L}(\{(x_j, y_j)\}; \boldsymbol{\theta}_t))^2$ (Update 2nd moment estimate)

    $\hat{\mathbf{m}} \leftarrow \frac{\mathbf{m}}{1-\beta_1^t}$ (Bias-corrected 1st moment estimate)

    $\hat{\mathbf{v}} \leftarrow \frac{\mathbf{v}}{1-\beta_2^t}$ (Bias-corrected 2nd moment estimate)

    $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\hat{\boldsymbol{v}}}+\epsilon} \cdot \hat{\mathbf{m}}$ (Update parameters)

    $t \leftarrow t + 1$

  end for

end for

Output: Learned parameters $\boldsymbol{\theta}_t$

2.3.1 Decoder-only models

The first type of language model we consider are decoder-only models. Decoder-only language models use the standard language modeling objective

$$\mathcal{L}_{\texttt{LM}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{t=1}^{T} \log p_{\boldsymbol{\theta}}(x_t^{(i)} \mid \mathbf{x}_{<t}^{(i)}) \ , \tag{2.26}$$

which we optimize by choosing parameters via maximum likelihood estimation. All of the models we experiment with are based on the transformer's decoder architecture (Vaswani et al., 2017; Al-Rfou et al., 2019; Radford and Narasimhan, 2018) and are a stack of layers consisting of self-attention, layer normalization,

and feed-forward layers as well as residual connections. We consider the following variants for our experiments.

OPT Open Pre-trained Transformer (OPT) (S. Zhang et al., 2022) is a family of pre-trained decoder-only transformers that range from 125M to 175B parameters. The architecture of OPT closely follows the GPT family (Radford et al., 2019; Brown et al., 2020) ranging from 12 layers for the smallest to 96 layers for the largest version. All models have been pre-trained on the same dataset which consists of a mixture of documents from the Books corpus (Y. Zhu et al., 2015), CC-Stories (Trinh and Le, 2019), The Pile (L. Gao et al., 2020), Pushshift.io Reddit (Roller et al., 2021), and CCNewsV2 (Y. Liu et al., 2019b; S. Zhang et al., 2022). The data have been tokenized using the GPT-2 tokenizer (Radford et al., 2019), resulting in a training set of roughly 180B tokens. At the time of its release, OPT-175 was the first publicly available language model that achieved comparable in-context learning results to GPT-3 (Brown et al., 2020).

Pythia Pythia (Biderman et al., 2023) is another family of decoder-only language models. The architecture largely follows GPT and OPT with a few notable differences such as using parallelized attention layers (B. Wang and Komatsuzaki, 2021), FlashAttention (Dao et al., 2022), and rotary positional embeddings (Su et al., 2022). Similar to OPT, Pythia models come in different sizes ranging from 70M to 6.9B parameters. All models use the GPT-2 tokenizer (Radford et al., 2019) and were trained on The Pile (L. Gao et al., 2020). Pythia models are specifically designed for interpretability research by ensuring that all models are trained on exactly the same data in exactly the same order.

2.3.2 Encoder-only models

The second family of pre-trained language models we consider are transformer-based encoder-only models. Encoder-only models are trained using a masked lan-

guage modeling objective which is an approximation to the pseudo-log-likelihood. Specifically, masked language modeling approximates the distribution over tokens only at special masked positions.

$$\mathcal{L}_{\texttt{MLM}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{t=1}^{T} \log\, p_{\boldsymbol{\theta}}(x_t^{(i)} \mid \mathbf{x}_{<t}^{(i)}, \mathbf{x}_{>t}^{(i)}) \mathbb{1}\{x_t^{(i)} = \texttt{[MASK]}\}\ . \tag{2.27}$$

For our experiments, we consider three popular variants of encoder-only models, which we briefly introduce below.

BERT Bidirectional encoder representations from transformer (BERT) (Devlin et al., 2019) is a transformer-based encoder-only model jointly trained on masked language modeling and next-sentence-prediction – a sentence-level binary classification task. BERT was trained on the Toronto Books corpus and the English portion of Wikipedia, which together consists of roughly 3.3B tokens. The BERT model comes in two different sizes *base* and *large*, consisting of 110M and 340M parameters respectively.

RoBERTa Robustly optimized BERT approach (RoBERTa) (Y. Liu et al., 2019a) is an optimized variant of BERT that differs from the original BERT model in several important aspects: RoBERTa was trained on more data, using a larger batch size, more training steps, longer input sequences, and no next-sentence-prediction objective. Additionally, it uses a sightly larger vocabulary and dynamically masks a fraction of the input tokens during training instead of applying a fixed masking chosen prior to training.

ALBERT A light BERT (ALBERT) (Lan et al., 2020) is another popular masked language model that, in contrast to BERT and RoBERTa, uses weight-sharing across all hidden layers—effectively applying the same non-linear transformation at every layer—and factorizes the embedding matrix into two separate matrices. These modifications result in significantly fewer model parameters while

maintaining overall performance. Similar to BERT, ALBERT uses a sentence-level training objective in addition to the masked language modeling objective.

2.4 Adapting pre-trained language models

As introduced in the previous section, pre-trained language models are trained via (masked) language modeling. This leads to a discrepancy between the task the model has to perform during pre-training, i.e., predict the next (masked) token, and the downstream task the model will eventually be applied to, e.g., sentence-level classification. To overcome this discrepancy and make pre-trained language models usable for downstream tasks, it is necessary to *adapt* them.

In the following subsections, we briefly review the two most commonly used strategies for adapting pre-trained language models to downstream tasks: fine-tuning and in-context learning.

2.4.1 Fine-tuning

Vanilla fine-tuning Vanilla fine-tuning (Howard and Ruder, 2018; Devlin et al., 2019) is one of the most commonly used task adaptation approaches for pre-trained language models. During fine-tuning we typically: (i) replace the model's language modeling head with a new randomly initialized classification head; (ii) update all model parameters, as well as the new head's parameters, on the downstream task's training data. Vanilla fine-tuning follows the supervised learning paradigm introduced in §2.2.1.

Pattern-based fine-tuning Pattern-based fine-tuning (Schick et al., 2020; Schick and Schütze, 2021; Tam et al., 2021; Logan IV et al., 2022, *inter alia*) is conceptually very similar to vanilla fine-tuning but differs in one important aspect. Instead of training a randomly initialized classifier on top of the pre-trained model,

pattern-based fine-tuning re-purposes the pre-trained language modeling head as a classifier. This requires us to format the downstream task as a language modeling problem via a *pattern* and use a *verbalizer* to map token predictions to the classes of the downstream task. To provide an example, assuming we are fine-tuning a masked language model on a sentiment analysis dataset, we could use the following pattern `The sentiment of [x] is positive ? [MASK]` to formulate sentiment classification as a masked language modeling problem and map the predictions of the model to class labels via the verbalizer `{"yes": positive, "no": negative}`.

Pattern-based fine-tuning follows the supervised learning paradigm introduced in §2.2.1 and has been shown to outperform vanilla fine-tuning in the few-shot setting (Le Scao and Rush, 2021), but exhibits a large sensitivity to pattern and verbalizer choice (Webson and Pavlick, 2022).

2.4.1.1 Parameter-efficient fine-tuning

While both vanilla and pattern-based fine-tuning typically update all pre-trained parameters, they can be easily combined with parameter-efficient fine-tuning methods. Parameter-efficient fine-tuning methods update only a small number of parameters relative to the total number of parameters of the pre-trained model (Houlsby et al., 2019; Ben Zaken et al., 2022; E. J. Hu et al., 2022, *inter alia*). These parameters are either newly introduced, e.g. adapters (Houlsby et al., 2019) or a subset of the pre-trained model's parameters, e.g., BitFit (Ben Zaken et al., 2022). Parameter-efficient fine-tuning approaches are appealing since they allow the re-use of large parts of a model across tasks. Moreover, when combined with quantization techniques, they allow us to efficiently fine-tune even billion-parameter language models(Dettmers et al., 2023).

2.4.2 In-context learning

A commonality shared by the adaptation approaches introduced above is the need for updating either existing or newly added parameters. In-context learning is an alternative approach that allows to adapt a pre-trained language model to perform a downstream task without the need for any parameter updates (Brown et al., 2020). Assuming we have a single example $\mathbf{x}$ for which we seek a prediction from a pre-trained language model. In-context learning works by prefixing $\mathbf{x}$ with a sequence of k demonstrations constructed from a set of training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{k}$, and optionally a task instruction string $\mathbf{s}$.[3] Similar to pattern-based fine-tuning, in-context learning applies a pattern to every demonstration $(\mathbf{x}_i, y_i)$ and the input example $\mathbf{x}$ and converts the prediction of the model to the task's label space using a verbalizer.

Few-shot in-context learning has recently been shown to lead to strong performance when using billion-parameter language models such as GPT-3 175B (Brown et al., 2020) and it outperforms vanilla fine-tuning, especially on challenging reasoning benchmarks (Wei et al., 2022a).

2.5 Probing language model representations

As a byproduct of their training, pre-trained (and fine-tuned) language models construct rich representations of their inputs. Probing is a commonly used analysis technique in NLP to gain insights into the representations construct by language models. Probing involves training a classifier on top of real-valued embeddings obtained from a pre-trained language model. The accuracy of the resulting classifier is considered a proxy for how well the representations capture the underlying concepts. When training probing classifiers, one has to make certain assumptions

3 Zero-shot evaluation is a special case of in-context learning where the set of demonstrations is empty and the task instruction is the empty string.

about the type of classifier (e.g., using linear vs. non-linear classifiers) and the implications of probing accuracy, as high probing accuracy does not necessarily imply that the language model genuinely utilizes the encoded concepts for its internal decision-making.

2.5.1 Sentence-level probing

In this thesis, we focus specifically on sentence-level probing (Adi et al., 2017; Conneau et al., 2018) which allows us to investigate linguistic concepts encoded within sentence-level representations. By probing sentence-level representations, we can gain insight into how language models process and represent complex linguistic structures on the sentence level. Drawing inspiration from early work in the field (Alain and Bengio, 2016), we adopt the use of linear probing classifiers in later chapters of this thesis.

Formally, let $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$ be a labeled dataset containing pairs of sentence-level representations and their corresponding linguistic labels, where $\mathbf{x}_i$ represents the sentence-level embedding of the i-th input, and y_i is the corresponding linguistic label or category associated with the sentence, e.g., a binary acceptability label. The goal of the supervised probing task is to learn a probing classifier $f_{\boldsymbol{\theta}}$, parameterized by $\boldsymbol{\theta}$, that maps the sentence-level embeddings $\mathbf{x}_i$ to their respective linguistic labels y_i. To train the probing classifier we estimate the parameters $\boldsymbol{\theta}$ of the probing classifier $f_{\boldsymbol{\theta}}(\mathbf{x}_i)$ that maximize the likelihood of the observed linguistic labels given the sentence-level embeddings following the maximum likelihood approach introduced in §2.2.1.

Once trained, we evaluate the accuracy of the probing classifier on a held-out test set. High probing accuracy is typically treated as evidence that the language model's sentence-level embeddings effectively encode and capture the linguistic concepts represented by the linguistic labels in the probing dataset.

3

Probing Pre-trained Models for Linguistic Knowledge

Contents

Transformer-based language models achieve high performance on various NLP tasks, but we still lack understanding of the kind of linguistic knowledge they learn and rely on. In this chapter, we investigate this question by analyzing three different pre-trained language models (BERT, RoBERTa, and ALBERT), testing their grammatical and semantic knowledge by sentence-level probing, diagnostic cases, and masked prediction tasks. We focus on relative clauses (in American

English) as a complex phenomenon needing contextual information and antecedent identification to be resolved. On a naturalistic dataset which we collected, our probing results show that all three models do indeed capture linguistic knowledge about grammaticality, achieving high performance on sentence level probing tasks. However, when evaluating the trained probing classifiers on challenging diagnostic cases and evaluating the predictions of the pre-trained models directly, we find pronounced model-specific weaknesses especially on semantic knowledge, which strongly impacts model performance. Our findings highlight the importance of (a) model comparison in linguistic evaluation tasks, and (b) building up claims of model performance and the linguistic knowledge they capture beyond purely probing-based evaluations.

The content presented in this chapter is based on:

> **Mosbach, Marius**, Stefania Degaetano-Ortlieb, Marie-Pauline Krielke, Badr M. Abdullah, and Dietrich Klakow (2020). "A Closer Look at Linguistic Knowledge in Masked Language Models: The Case of Relative Clauses in American English." In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 771–787. URL: https://aclanthology.org/2020.coling-main.67.

As a first author, Marius Mosbach conceptualized the research, conducted the experiments and led the paper writing. Stefania Degaetano-Ortlieb helped with the conceptualization and paper writing. Marie-Pauline Krielke provided expertise on relative clauses, helped with the writing and the lingustic evaluation. Badr Abdullah collected the probing data and helped with writing. Dietrich Klakow advised and provided feedback.

3.1 Introduction

Endeavors to better understand transformer-based masked language models such as BERT have been an active area of research since the introduction of these models in 2017 (see Rogers et al. (2020) for an overview). While the BERTology movement has enhanced our knowledge about the reasons behind BERT's impressive performance in various ways, plenty of questions remain unanswered. For instance, linguistic phenomena such as relative clauses (RC) are less well-studied. Besides contextual information, they require the identification of an antecedent, making them more challenging for such models. Kim et al. (2019a), e.g., analyzed BERT's comprehension of function words, showing how relativizers and prepositions are quite challenging for BERT. Similarly, Warstadt and Bowman (2019) find sentences containing RCs to be difficult for BERT in the CoLA acceptability tasks.

In this chapter, we focus on RCs in American English[1] to further enhance our understanding of the grammatical and semantic knowledge captured by pre-trained masked language models, evaluating three models: BERT, RoBERTa, and ALBERT. For our analysis, we train probing classifiers, consider each models' performance on diagnostic cases, and test predictions in a masked language modeling task on selected semantic and grammatical constraints of RCs.

RCs are clausal post-modifiers specifying a preceding noun phrase (antecedent) and are introduced by a relativizer (e.g., *which*). Extensive corpus research (Biber et al., 1999) has found that the most common English relativizers are *that*, *which*, and *who*. The relativizer occupies the subject or object position in a sentence (see examples (1-a) and (1-b)). In subject RCs, the relativizer is obligatory (Huddleston and Pullum, 2002, p. 1055), while in object position omission is licensed (e.g., *zero* in example (1-b)).

1 We make our focus on America English explicit as we obtain our probing data from the Corpus of Contemporary American English (Davies, 2015) and there exist subtle differences in the use of relative clauses between American and British English (Rohdenburg, 2014).

(1) a. Children *who* eat vegetables are likely to be healthy. (subject relativizer, relativizer is obligatory)

b. This is the dress [*that/which/zero*] I brought yesterday. (object relativizer, omission possible)

Relativizer choice depends on an interplay of different factors.[2] Among these factors, the animacy constraint (Quirk, 1957) is near-categorical: for animate head nouns the relativizer *who* (see Example (1-a)) is strongly prioritized, especially over *which* (D'Arcy and Tagliamonte, 2010).

Our aims are (1) to better understand whether sentence representations of pre-trained masked language models capture grammaticality in the context of RCs, (2) test the generalization abilities and weaknesses of probing classifiers with complex diagnostic cases, and (3) test prediction of antecedents and relativizers in a masked task considering linguistic constraints (see Figure 3.1).

From a linguistic perspective, we ask whether pre-trained masked language models correctly predict (a) grammatically plausible relativizers given certain types of antecedents (animate, inanimate) and grammatically plausible antecedents given certain relativizers (*who* vs. *which/that)*, and (2) semantically plausible antecedents given certain relativizers considering the degree of specificity of predicted antecedents in comparison to target antecedents (e.g., *boys* as a more specific option than *children* in Example (1-a)). Importantly, we are interested in how these findings agree with probing results and thus investigate model specific behavior, evaluating and comparing recent pre-trained masked language models: BERT, RoBERTa, and ALBERT. This is to our knowledge the first attempt at comparing and analyzing performance of different transformer-based

2 These factors include register (fiction, news, academic texts), restrictiveness (restrictive RCs add information about the head noun necessary for identification of the latter (see Example (1)); non-restrictive RCs add information elaborating on a head noun which "is assumed to be already known" Biber et al. (1999, p. 602) and are usually separated by a comma, e.g., *My children, who love me, eat vegetables.*), animacy of the head noun (animate or inanimate antecedent), American (AE) vs. British English (BE), and definiteness of a pronominal antecedent (demonstrative vs. indefinite pronoun).

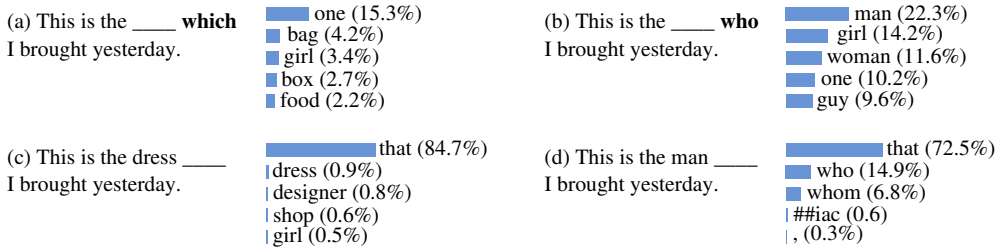


Figure 3.1: Top-5 predictions by BERT-base-cased when masking (a) inanimate antecedent, (b) animate antecedent, (c) inanimate relativizer and (d) animate relativizer.

masked language models in such detail, investigating grammatical and semantic knowledge beyond probing.

Our main contributions are the following: (1) the creation of a naturalistic dataset for probing, (2) a detailed model comparison of three recent pre-trained masked language models, and (3) fine-grained linguistic analysis on grammatical and semantic knowledge. Overall, we find that all three masked language models show good performance on the probing task. Further evaluation, however, reveals model-specific issues with agreement (where RoBERTa shows the strongest performance) and sensitivity to the distance between antecedent-relativizer and relativizer-RC verb (on which BERT and ALBERT are better). Considering linguistic knowledge, all models perform better on grammatical rather than semantic knowledge. Out of the relativizers, *which* is hardest to predict. Considering model-specific differences, BERT outperforms the others in predicting the actual targets, while RoBERTa is the best at capturing grammatical and semantic knowledge. ALBERT performs worst overall.

3.1.1 Models

For our experiments, we consider three transformer-based encoder-only models: BERT (Devlin et al., 2019), RoBERTa (Y. Liu et al., 2019b), and ALBERT (Lan et al., 2020). We provide details about each of the models in §2.3.2.

Importantly, for the scope of this chapter, we consider their base variants: *BERT-base-cased*, *RoBERTa-base*, *ALBERT-base-v1* with 110M, 125M, and 12M parameters, respectively and provide additional experiments for ALBERT-xxlarge-v1 in Appendix A.2.1.

3.2 Related Work

Related work has investigated grammaticality of unidirectional language models using *Minimal Pair Evaluation* (Marvin and Linzen, 2018; Wilcox et al., 2019; Warstadt et al., 2020a; J. Hu et al., 2020a; J. Hu et al., 2020b). Out of work on language model prediction-based evaluation (e.g., Goldberg (2019), Ettinger (2020), Petroni et al. (2019), Jiang et al. (2020), and Kassner and Schütze (2020)), only Goldberg (2019) has so far evaluated masked language model predictions in the context of grammaticality. Our study adds to this line of work, focusing on RCs in particular and importantly, combining masked language model prediction-based evaluation with probing.

Work related to evaluating sentence embeddings has considered prediction of sentence length, word content, and word order (Adi et al., 2017). Conneau et al. (2018) investigated an even broader range of linguistic properties. Extensive work has been done on probing token-level representations of pre-trained masked language models, especially BERT, for syntactic and semantic knowledge (see, e.g., Tenney et al. (2019b), N. F. Liu et al. (2019) and Rogers et al. (2020) for a more comprehensive overview).

Most similar to our work is Warstadt et al. (2019), which focuses on comparing evaluation methods including probing and masked language modeling evaluation to assess how models encode linguistic features.

We contribute to this strand of research by building datasets from naturalistic (rather than artificial, as in Warstadt et al. (2019)) data, and comparing three transformer models: BERT, RoBERTa, and ALBERT. Our focus is on RCs as challenging sentence types for pre-trained language models as shown by, e.g., Kim

et al. (2019a) who show relativizers to be challenging for BERT, and Warstadt and Bowman (2019) who find RCs difficult for BERT in the CoLA tasks.

3.3 Probing representations for knowledge of RCs

For our probing based evaluation, we train supervised *probing* classifiers (here: acceptability classifiers) to assess the linguistic knowledge contained in a model's representations, focusing on grammaticality. This assumes that a model's grammaticality awareness should be reflected in the hidden representations produced by the model for that particular sentence. Hence, by training a classifier on top of these representations, we should be able to discriminate between grammatical and ungrammatical sentences based on their sentence embeddings.[3] Besides capturing knowledge on grammaticality, we also examine whether this knowledge becomes more or less separable in the representation, considering representations produced by different layers of a model.

In the following, we describe the collection of our probing dataset and provide details about our probing setup before discussing our empirical results.

3.3.1 Dataset construction

To probe pre-trained masked language models' performance on sentences containing RCs, we construct a controlled set of 48,060 sentences and their acceptability labels using an automated procedure. Our dataset is a subset of naturally occurring sentences extracted from the fiction portion of the COCA corpus (Davies, 2015).

First, we extract all sentences containing only one pronoun from the set {*who, whom, whose, which, that*}. We then parse the sentences using SpaCy's

3 This further assumes that grammatical and ungrammatical sentences are *linearly separable* in the embedding space.

A	R	SubjRC	Modification	Sentence
1	1	1	no modification	Katrina Haus was a woman **who** sought to attract stares, not turn them away.
1	1	1	who → which	*Katrina Haus was a woman **which** sought to attract stares, not turn them away.
0	1	0	no modification	She pulls out a course catalog, various forms, and a letter **which** she hands to Kevin.
0	1	0	which → that	She pulls out a course catalog, various forms, and a letter **that** she hands to Kevin.
0	1	0	no modification	Never permit your muzzle to cover anything **which** you are unwilling to shoot.
0	1	0	relativizer omission	Never permit your muzzle to cover anything you are unwilling to shoot.
0	0	0	no modification	I never saw a penny in royalties, **which** was all right with me.
0	0	0	which → who	*I never saw a penny in royalties, **who** was all right with me.

Table 3.1: Examples from the dataset (minimal pairs). A denotes Animate, R denotes Restrictive. The relativizer is shown in bold. Modifying a sentence does not always result in an ungrammatical sentence. For example, when Restrictive=1 and SubjRC=0, relativizer omission yields a grammatical sentence.

dependency parser and keep only those sentences where the pronoun constitutes a relativizer (identified by the tag RELCL). From the parse tree, we also determine whether the relativizer fills the subject or the object position in the RC. The automatic selection procedure is illustrated in Figure A.1 in Appendix A.1.

On top of grammatical sentences from the corpus, we manipulate the data to obtain a set with unacceptable ones. To this end, we populate three boolean metadata variables for each grammatical sample in our data using a set of hand-crafted linguistic rules: ANIMATE, RESTRICTIVE, and SUBJRC. Based on the values of these three metadata variables, a set of modifications is applied to convert a grammatical sentence into an ungrammatical one. The dataset creation procedure is explained in detail in Appendix A.1. Our final dataset consists of 42.7K and 5.3K samples for training and evaluation, respectively. Both splits are balanced, i.e., the accuracy of a majority baseline is 50%. Table 3.1 presents a set of minimal pair examples that are generated using our procedure.

3.3.2 Experimental setup

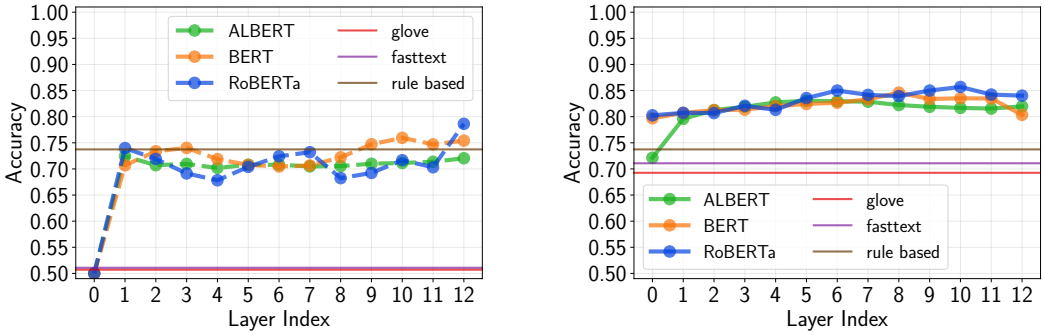
For our probing setup we follow the sentence-level probing paradigm introduced in §2.5.1 and train logistic regression acceptability classifiers on sentence embeddings obtained from the hidden layers of a pre-trained model.

We compute sentence embeddings of BERT, RoBERTa, and ALBERT and two non-contextualized baselines: GloVe embeddings (Pennington et al., 2014) trained on English Wikipedia and the Gigaword corpus, and fasttext embeddings (Bojanowski et al., 2017) trained on English Wikipedia and news data (Mikolov et al., 2018). As an additional baseline we use a rule-based classifier that simply classifies sentences containing a relativizer (who, which, that) as grammatical and as ungrammatical otherwise.

Input sentences are pre-processed by adding two special tokens, [CLS] and [SEP], at the beginning and end of each input sentence respectively.[4] To construct sentence embeddings, we apply two different pooling strategies: CLS- and mean-pooling. CLS-pooling simply returns the vector representation of the first token of the input sentence, i.e., the [CLS] token. Mean-pooling computes a sentence embedding by taking the mean over all (sub-word) token representations of the input sentence. We obtain sentence embeddings from all hidden layers of the masked language models, including the non-contextualized embedding layer (layer 0) and treat accuracy on the acceptability classification task as a proxy for the linguistic knowledge encoded in a model's sentence embeddings, which we evaluate on a held-out test set.

We use the huggingface transformers (Wolf et al., 2020) and flair (Akbik et al., 2018) libraries as well as scikit-learn (Pedregosa et al., 2011) to obtain embeddings and train the logistic regression classifiers.

4 To be precise, for RoBERTa the special tokens are <s> and </s>. For GloVe and fasttext embeddings as well as the rule-based classifier, we tokenize on the word level without using special tokens.



(a) CLS-pooling **(b)** mean-pooling

Figure 3.2: Side-by-side comparison of layer-wise probing accuracy on the test set for pre-trained transformer and baseline models using (a) CLS-pooling and (b) mean-pooling.

3.3.3 Probing results and discussion

Figure 3.2 shows the layer-wise probing accuracy for all models when using cls- and mean-pooling. Mean-pooling leads to higher probing accuracies for all models, suggesting a sub-optimal encoding of sentence-level information in the CLS token representation. Hence, we stick to mean-pooling for later experiments. We find that the rule-based classifier is a surprisingly strong baseline, outperforming both the GloVe and fasttext baselines. Moreover, Figure 3.2 shows that all three transformer-based models improve significantly over the baselines for almost all layers. The only exception is layer 0 of ALBERT, which performs similarly to the fasttext baseline and worse than the rule-based classifier. We attribute this finding to the embedding factorization of ALBERT. At lower (contextualized) layers (1–5), probing accuracies of BERT, RoBERTa, and ALBERT are almost identical. For higher layers, both BERT and RoBERTa improve over ALBERT, with ALBERT's probing accuracy remaining roughly constant. We note that ALBERT has significantly fewer parameters than BERT and RoBERTa (12M vs. 110M and 125M), which might explain the lower probing accuracy. We provide

Modification	GloVE	fasttext	Rule-based	BERT	RoBERTa	ALBERT
no modification	67.3	69.2	100	83.7 (78.4)	85.3 (79.7)	83.6 (71.5)
relativizer omission	77.2	85.3	97.6	96.0 (98.3)	95.1 (98.2)	96.4 (94.8)
who → which	88.0	81.7	0.00	84.9 (88.4)	87.3 (89.6)	78.8 (68.3)
which → who	18.2	21.3	0.00	47.9 (1.73)	50.0 (0.02)	44.2 (18.7)
which → that	12.8	8.0	100	80.0 (52.0)	77.6 (44.8)	72.0 (22.4)
total	69.2	71.0	73.7	84.6 (79.7)	85.5 (80.1)	83.0 (72.1)

Table 3.2: Test accuracy (in %) grouped by modification type (cf. Table A.3 for statistics). For BERT, RoBERTa, and ALBERT we select the best model according to the probing results shown in Figure 3.2. Numbers in parenthesis show the accuracy of the non-contextualized baseline (layer 0) for each model.

a more detailed discussion investigating the role of the number of parameters in Appendix A.2.1.

Overall, the fact that probing accuracy is above 80% for almost all layers suggests a reasonable encoding of sentence-level linguistic knowledge relevant for grammaticality classification in all pre-trained models. Notably, linear separability with respect to grammaticality emerges very early in the sentence embeddings of all three models.

To get a better understanding of the accuracy achieved by each of the models, we select the best classifier according to the results in Figure 3.2 and report test accuracy grouped by modification in Table 3.2. For comparison, we additionally report accuracies of the non-contextualized baselines (layer 0) of each model in parentheses.

The results show that while contextualization leads to higher probing accuracy overall, it is especially important for the `which` → `who` and `which` → `that` samples. From a linguistic viewpoint this is surprising: e.g., replacing *which* by *who* clearly makes a sentence ungrammatical and is typically easy to detect for humans. When looking at the training data, however, this observation is not

surprising at all. The vast majority of sentences (15K samples) belonging to the `no modification` group contain *who* as a relativizer. All of these sentences are grammatical (see Table A.3). On the other hand, `which` → `who` contains only 2.5K samples, all of them are ungrammatical. Hence, our results in Table 3.2 suggest that the non-contextualized baselines might have learned a simple heuristic, classifying all sentences that contain *who* as a relativizer as grammatical. The results of the rule-based baseline classifier give further evidence for this interpretation, showing that a simple classifier which bases its predictions only on the existence of a relativizer has a surprisingly high accuracy (comparable to the non-contextualized baselines) on our dataset. Interestingly, BERT and RoBERTa seem to be especially susceptible to learning this short-cut as shown by the results of the non-contextual baselines for BERT and RoBERTa on the `relativizer omission` and `which` → `who` modifications.

3.3.4 Diagnostics

To further investigate the generalization abilities and model-specific behavior of the best probing classifiers, we evaluate them on a diagnostic dataset containing sentences with the following properties: (1) adjacent antecedent and relativizer (see example (2-a), grammatical), (2) longer distance between antecedent and relativizer (see example (2-b), grammatical), (3) longer distance between relativizer and RC verb (see example (2-c), grammatical), (4) incorrect agreement between adjacent antecedent and relativizer (see example (2-d), ungrammatical), and (5) intervening agreement attractors leading to incorrect agreement (see example (2-e), ungrammatical).

To create the dataset, we manually select four sentences and manipulate these according to each of the above-mentioned cases. Three sentences have nominal antecedents and one has a clausal antecedent. For the grammatical manipulations (case 1–3), we additionally test restrictive and non-restrictive variants of each

sentence. Overall, we test a total of 32 sentences[5] and evaluate the models' confidence based on the un-normalized log probabilities (logits), where predictions > 0 result in classifying sentences as grammatical. In the case 2 manipulations, besides considering nominal vs. clausal antecedent, we look at the number of words of intervening phrases (length of 3–7 intervening words).[6]

(2) a. We just heard **a debate which** was about the differences in wage rates [...]. (case 1: adjacent antecedent and relativizer)

b. [...] **a debate** on one of the most famous television channels **which** was about [...]. (case 2: distance from antecedent to relativizer)

c. [...] a debate **which** in many regards **was** an important one about [...]. (case 3: distance from relativizer to verb)

d. *[...] **a debate who** was about [...]. (case 4: incorrect agreement between antecedent and relativizer)

e. *[...] a debate **by DeGeneres** who was about [...]. (case 5: agreement attractor)

Case 1 For restrictive and non-restrictive variants and nominal antecedents, BERT and ALBERT correctly predict sentences to be grammatical with relatively high confidence (restrictive: 2.7, non-restrictive: 1.6). With a clausal antecedent, they fail in the restrictive variant (-0.39 and -1.18, respectively), but they can deal with non-restrictive RCs (0.89 and 0.57), the comma possibly being a strong indicator of non-restrictiveness. RoBERTa always incorrectly predicts that sentences are ungrammatical.

Case 2 With intervening phrases between an antecedent and a relativizer, BERT and ALBERT again do well on both restrictive and non-restrictive RCs for

5 We opted for a small set for high control over factors influencing a choice.

6 Note that for case 3, the distance between the relativizer and the verb is always three words, so length is not a factor that needs to be evaluated.

Case	Factor	BERT		ALBERT		RoBERTa	
		restrictive	non-restrictive	restrictive	non-restrictive	restrictive	non-restrictive
1 (+)	nominal	2.74	1.62	2.35	1.75	−0.79	−0.85
	clausal	−0.39	0.89	−1.18	0.57	−0.85	−0.57
2 (+)	nominal	2.34	1.74	1.30	1.72	−1.03	−0.87
	clausal	−0.39	0.89	−1.18	0.57	−0.85	−0.57
	3–4 words	2.35	2.52	1.30	2.55	−1.03	−0.55
	> 4 words	−0.69	0.54	−1.15	0.31	−1.42	−1.04
3 (+)	nominal	1.99	1.97	2.04	1.78	−1.06	−0.80
	clausal	−0.84	0.43	−1.54	0.17	−0.89	−0.35
4 (*)	nominal	−0.47	–	−0.40	–	−1.46	–
	clausal	−1.38	–	−1.75	–	−1.24	–
5 (*)	nominal	0.56	–	0.08	–	−1.24	–
	clausal	−1.38	–	−1.75	–	−1.24	–

Table 3.3: Prediction confidence in mean logit (> 0: grammatical, < 0: ungrammatical). Sentences in case 1 to 3 should be predicted as grammatical (+) and case 4 and 5 as ungrammatical (*).

nominal antecedents, but fail on restrictive RCs in the clausal case. Considering the length of the intervening phrase, for restrictive RCs, BERT deals with intervening phrases with high confidence (2.35), provided they are relatively short (3–4 words). For longer distance, BERT is less confident (−0.69) and chooses the (wrong) ungrammatical class. ALBERT behaves similarly, but fails with higher confidence (−1.15). RoBERTa always fails without a clear pattern. For the non-restrictive RCs, both BERT and ALBERT correctly predict the grammatical class, even though distance affects their confidence (the longer the less confident: 2.52 for shorter, 0.54 for longer phrases). RoBERTa again predicts the ungrammatical class for all sentences, but is especially confident for greater distance between antecedent and relativizer (−0.55 for shorter, −1.04 for longer phrases).

Case 3 With restrictive RCs, RoBERTa has problems with intervening phrases between the relativizer and the RC verb, again always predicting the ungrammatical class. BERT predicts this case correctly, but fails on sentences with a clausal antecedent, even though with lower confidence than ALBERT (−0.84 vs. −1.54). In non-restrictive RCs, BERT and ALBERT obtain perfect accuracy for both nominal and clausal antecedents, being quite confident in their predictions, while RoBERTa still fails, sometimes with very high confidence.[7]

Case 4 With incorrect agreement and adjacent antecedents and relativizers, RoBERTa is quite confident of the ungrammaticality of the sentences (−1.46), while BERT and ALBERT are confident in the clausal antecedent case (−1.38 and −1.75, respectively), but much less confident or even wrong with nominal antecedents.

Case 5 With intervening agreement attractors and incorrect agreement (see example (2-e), where *DeGeneres* is considered the antecedent instead of *debate*), BERT gets confused due to the attractors and is only confident in the clausal antecedent case (−1.38). ALBERT gets confused as well, but less often and with much lower confidence than BERT (0.08). RoBERTa, on the other hand, is very confident in recognizing the incorrect agreement (−1.24).

In summary, RoBERTa is quite confident in the case of incorrect agreement. BERT and ALBERT deal much better than RoBERTa with longer distances between antecedent and relativizer. Also, BERT and ALBERT learn to recognize non-restrictive RCs quite well and can deal with phrases between the relativizer and the RC verb. Thus, *even though the models achieve very high probing accuracy overall (see § 3.3.3), evaluating on more complex cases reveals that each model seems to rely on different kinds of information, strongly affecting the generalization*

7 BERT and ALBERT seem to rely on the comma as an indicator of grammaticality for non-restrictive RCs (cases 0–2).

abilities of the probing classifiers. While we are aware of the diagnostic set's very limited size, hindering generalizable conclusions, the controlled diagnostic evaluation gives an indication of possible differences underlying prediction choices across models.

3.4 Analyzing predictions for RC awareness

To get a more comprehensive picture of the differences between models, we perform a masked language modeling evaluation (Goldberg, 2019), looking at the models' predictions of relativizers as well as antecedents. Besides grammaticality, we also test whether the models capture semantic knowledge.

3.4.1 Analyzing grammatical and semantic knowledge

We extract sentences containing restrictive[8] object or subject RCs with one of the three relativizers from the *magazines* and *academic* registers[9] from the COCA corpus (Davies, 2015). All sentences are formatted as described in §3.3.2, e.g., `[CLS] The woman [MASK] studies linguistics. [SEP]` The size of each set of sentences of a particular RC type depends on the frequency of available sentences in the COCA corpus and therefore varies between types from 20 to 50 sentences.

3.4.1.1 Relativizer prediction

We test all three models by masking the relativizer, considering the following metrics for evaluation: (1) mean precision at 1 (MP@1) (model's precision of target prediction at first position), (2) mean target rank (MTR), and (3) normalized mean entropy (NME) (uncertainty of the model's prediction).

8 We deliberately exclude non-restrictive RCs due to the indicative comma facilitating RC recognition (see §3.3.4).

9 We exclude fiction as it was used for the probing experiments, and spoken, due to possible high noise in spoken data.

		objRC			subjRC		
		who	which	that	who	which	that
MP@1	BERT	**0.80**	0.29	0.89	**0.98**	0.02	**0.92**
	RoBERTa	**0.80**	0.24	**0.96**	0.92	0.04	**0.92**
	ALBERT	0.32	**0.43**	0.86	0.96	**0.16**	0.76
MTR	BERT	**1.23**	1.83	1.11	**1.00**	2.46	**1.10**
	RoBERTa	**1.23**	1.95	**1.04**	1.08	2.18	1.22
	ALBERT	2.29	**1.73**	1.39	1.02	**1.18**	1.37
MNE	BERT	0.09	0.08	**0.05**	0.05	0.07	0.06
	RoBERTa	**0.08**	**0.06**	0.05	**0.04**	**0.05**	**0.04**
	ALBERT	0.19	0.21	0.18	0.12	0.13	0.17

(a) Multi-metric evaluation (MP@1: mean precision at 1, MTR: mean target rank, MNE: mean normalized entropy)

		objRC			subjRC		
		who	which	that	who	which	that
AN	BERT	**0.87**	**1.00**	0.93	**1.00**	**0.94**	**0.96**
	RoBERTa	0.81	**1.00**	**0.96**	0.94	0.90	0.94
	ALBERT	0.71	**1.00**	0.93	**1.00**	**0.94**	0.92
PL	BERT	0.81	**0.98**	0.93	**1.00**	0.92	0.96
	RoBERTa	**0.97**	**0.98**	**0.96**	0.98	**0.98**	**0.98**
	ALBERT	0.68	0.95	0.89	0.98	0.96	0.92
GR	BERT	0.94	**1.00**	0.93	**1.00**	**0.98**	**1.00**
	RoBERTa	**0.97**	**1.00**	**0.96**	**1.00**	**0.98**	**1.00**
	ALBERT	0.68	0.95	0.89	0.98	**0.98**	0.96

(b) Semantic (AN: animacy, PL: plausibility) and grammaticality (GR) evaluation.

Table 3.4: *Relativizer* prediction: quantitative (a) and qualitative (b) evaluation. **Bold**: best result for each metric and category across models. Underline: best result for each model per metric across categories.

For MP@1, all three models generally perform well across RCs apart from *which* (Table 3.4a). BERT and RoBERTa show similar performance, while ALBERT diverges slightly, with fairly weak predictions of *who* in object RCs and comparatively accurate predictions of *which* in both object and subject RCs. MTR is negatively correlated with MP@1 (the larger the divergence from 1, the lower the precision of the prediction). NME reflects the two other measures: BERT and RoBERTa are quite confident about their predominantly successful predictions, while ALBERT shows a higher uncertainty about overall much weaker predictions.

Next, we manually evaluate[10] the actual predictions according to three criteria: *animacy* (agreement between antecedent and relativizer), *plausibility* (semantically plausible sentence) and *grammaticality* (Table 3.4b). Results show that the actual felicity of the predictions is much stronger than MP@1 would suggest, since non-target predictions are not necessarily infelicitous. This is especially true for *which*, due to high interchangeability with *that*. While both relativizers are synonymous,

10 Evaluations were done by two linguistic experts.

which is primarily used in non-restrictive RCs. Since the sample sentences are restrictive, the models seem to predict the most frequent relativizer in restrictive RCs, which is *that* (see example (3)).

(3) The action [MASK] it contemplates is command. (all models) (object RC, target= *which*, prediction= that)

Animacy is predicted reliably by BERT and RoBERTa, while ALBERT seems to have issues with *who* object predictions. This is due to ALBERT's preference to predict *that* instead of *who*, especially in fuzzy cases of general nouns describing humans (*person*, *people*, *family*, etc. – see example (4)). Sometimes animacy is predicted correctly but case marking is infelicitous; the model seems to only take into account the subject and verb of the clause ignoring possible clausal complements (see example (5)).

(4) He spared no expense in moving, reassembling, and restoring buildings of people [MASK] he felt were the backbone of our nation. (ALBERT) (object RC, target= *who*, prediction=*that*)

(5) Jennifer had been having an online affair with a person [MASK] she believed was a man named Christopher. (ALBERT) (object RC, target= *who*, prediction= *whom*)

Accuracy for plausibility is very high as well, with ALBERT making the least plausible predictions. Note that plausibility entails grammaticality. Example (5) has animacy agreement, but is neither grammatical nor plausible. Grammaticality in contrast does not entail plausibility (see example (6)). Here, BERT predicts the most frequent collocate of *whirl* according to COCA (Davies, 2015).

(6) This is something else, for I saw a whirl [MASK] I knew was a large bass. (BERT) (object RC, target= *which*, prediction= *wind*)

Of all criteria, predictions are most accurate for grammaticality. Even ALBERT reaches a precision of 90% on average. Finally, we consider the ratio of (any) relativizers predicted vs. other word classes, indicating how well the models recognize a syntagmatic environment typical for RCs. RoBERTa is again the most successful model here (95% of predictions are relativizers), followed by BERT (94%) and ALBERT (90%).

In summary, BERT and RoBERTa perform equally well at target prediction, while RoBERTa is most successful qualitatively (relativizer predictions that satisfies animacy, plausibility and grammaticality expectations). Overall, our qualitative analysis shows that all models largely make grammatical, plausible and animacy-conforming predictions.

3.4.1.2 Antecedent prediction

Next, we test all three models by masking the antecedent (see example (7)), considering again, mean precision (MP@1), mean target rank (MTR), and normalized mean entropy (NME) (Table 3.5a). Due to higher variation in lexical choices for antecedents, MP@1 is expectedly much lower than for relativizers.

Best @1 predictions are achieved by all models for *who* and *that* object RCs. Comparing models, BERT achieves best performance in all cases. MTR is around 2.5 and 3.5, with antecedents in *who* subject RCs being predicted most easily by all models (rank around 2.5), also reflected in NME. The models are equally confident in *who* object RCs, while they are less confident for the other cases, with ALBERT being the least confident and RoBERTa the most confident model.

(7) Rheumatologists have to be medical detectives, because so many of the [MASK] **that** we treat are obscure. (object RC, masked target: **diseases**)

Considering the first predicted antecedent, we qualitatively evaluate whether (a) the animacy constraint is met, i.e., *who* RCs should have animate antecedents,

		objRC			subjRC		
		who	which	that	who	which	that
MP@1	BERT	**0.38**	**0.22**	**0.38**	**0.27**	**0.31**	**0.29**
	RoBERTa	0.31	0.08	0.29	0.22	0.17	0.14
	ALBERT	0.28	0.14	0.29	0.20	0.15	0.14
MTR	BERT	3.04	**2.82**	3.41	**2.42**	3.30	2.93
	RoBERTa	**2.65**	3.80	**2.33**	2.60	**2.67**	**2.82**
	ALBERT	3.20	3.23	2.45	2.58	3.17	3.18
NME	BERT	0.23	0.38	0.27	0.26	0.32	0.36
	RoBERTa	**0.18**	**0.28**	**0.22**	**0.21**	**0.24**	**0.27**
	ALBERT	0.40	0.50	0.41	0.39	0.42	0.50

(a) Multi-metric evaluation (MP@1: mean precision at 1, MTR: mean target rank, MNE: mean normalized entropy)

		objRC			subjRC		
		who	which	that	who	which	that
AN	BERT	0.97	0.94	**0.95**	0.98	0.94	0.98
	RoBERTa	**1.00**	0.97	**0.95**	**1.00**	**1.00**	**0.96**
	ALBERT	**1.00**	**1.00**	0.90	0.96	**1.00**	**0.96**
PL	BERT	**0.97**	0.94	0.95	**1.00**	0.90	**0.96**
	RoBERTa	**0.97**	**0.97**	**1.00**	0.98	**0.92**	0.92
	ALBERT	**0.97**	**0.97**	0.76	0.96	0.65	0.73
GR	BERT	**1.00**	0.97	**1.00**	**1.00**	0.98	**1.00**
	RoBERTa	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ALBERT	**1.00**	**1.00**	0.95	0.96	0.98	0.94

(b) Semantic (AN: animacy, PL: plausibility) and grammaticality (GR) evaluation.

Table 3.5: *Antecedent* prediction: quantitative (a) and qualitative (b) evaluation. **Bold**: best result for each metric and category across models. Underline: best result for each model per metric across categories.

and *that* and *which* most prominently inanimate ones[11] (see example (8)), (b) the sentence is plausible, (c) the sentence is grammatical (example (9) shows ungrammaticality because of incorrect noun-verb agreement between *animals* (plural) and *was* (singular)[12]).

(8) They picked a great foreman in the middle-aged, an African-American [MASK] **who** she said is a science professor with a Ph.D. (target = man, prediction = comma)

(9) By the time Ganivet drafted Idearium español, he was applying Ribot's psychological theories to the apparent inertia of [MASK] **which** he deemed was suffering from a collective psychological malady. (target = Spain, prediction = animals)

11 In cases where an elliptical version or other is predicted, we still check whether the animacy was kept or changed.

12 Here, BERT seems to consider *inertia* as the antecedent, matching the agreement with *was*.

Overall, BERT and RoBERTa show very high accuracy for these cases (see Table 3.5b). In a few cases, the animacy of the antecedent is changed (see example (10)) and sentences are grammatical but semantically implausible or ungrammatical and implausible (see examples (10) and (11)). ALBERT performs worst when it comes to plausibility, especially for *which* in subject RCs (see example (12)), most cases being repetitions of words in the sentence (see example (13)).

(10) The only thing that brightened the gloom stretching out before him was the goodness of the [MASK] **that** had offered him refuge before his trial. (target = family, prediction = darkness)

(11) Coffee, the cash, and the goods they purchased, even when used in traditional exchange, were devoid of the social relations with [MASK] **which** were present in traditional products. (target = predecessors, prediction = humans)

(12) He and Carolyn saw all of Kevin's college [MASK] **that** were within a day's drive of McIntyre. (target = games, prediction = campuses)

(13) The bill makes it illegal to adopt or enforce any law or [MASK] **which** allows gays to claim discrimination. (target = policy, prediction = law)

In summary, all models have problems with the relativizer *which*and they perform best on *who*. The models perform worst on animacy and plausibility, especially for subject RCs, but perform quite well for grammaticality, indicating a better awareness for grammatical than semantic knowledge.

We further evaluate the models' semantic knowledge considering predicted types of antecedents based on their relationship to the target, i.e., (a) identical, (b) synonym, (c) hypernym, general noun, determiner, pronoun, (d) hyponym[13], or (e) not directly related (i.e., no direct hierarchical relationship) or completely unrelated.[14] Results (see Table A.5 in Appendix A.2.2) show that all models

13 Hyponyms are semantically lower in hierarchy and thus more specific.

14 We relied on WordNet whenever possible (low coverage) and on other linguistic resources as well as the expertise of two linguists.

perform less well on subject than object RCs. ALBERT performs the worst as the majority of its predictions are in the category of not directly related/unrelated antecedents, or hypernyms (more general words). RoBERTa is quite good at predicting *identical* targets, outperforming the other two models especially in object RCs with *who*. In *who* subject RCs, BERT and ALBERT most often predict more general antecedents given more specific targets (hypernyms, e.g., *person* instead of *girl*, *workers*, etc.).

3.5 Discussion and conclusion

Our work makes progress towards tackling some issues in the evaluation of the linguistic capabilities of pre-trained transformer-based masked language models as, e.g., proposed by Rogers et al. (2020). Most prominently, we try to better understand how strong performance on supervised probing tasks is reflected in the predictions of the language models. To do so, we create a dataset based on naturalistic (not artificially generated) data and perform an extensive evaluation of masked language predictions in the context of RCs. Moreover, rather than considering only one model, we compare three models to investigate the extent to which findings for BERT can be generalized to other transformer-based models.

Our probing results show a significant improvement of all three transformer-based models over the baselines for almost all layers, suggesting that models do indeed encode linguistic knowledge relevant for grammaticality classification, as shown in previous work (e.g., Goldberg (2019)). Performance is similar across models, with BERT and RoBERTa performing slightly better than ALBERT, and while contextualization improves performance overall, we have shown that it helps immensely when considering less frequent RC modification types such as `which` → `who`, for which uncontextualized baselines learn simple heuristics. Evaluation on a diagnostic set, however, clearly reveals the weaknesses of probing classifiers, and model-specific behavior. RoBERTa is quite confident in the case of incorrect agreement between antecedent and relativizer, while this is tougher

for BERT and ALBERT. Considering distance between relativizer and antecedent/RC verb, however, both models clearly outperform RoBERTa. Based on these insights, **we conclude that viewing probing results in isolation can lead to overestimating the linguistic capabilities of a model**.

Our masked language modeling evaluation provided deeper insights into model-specific differences. We evaluated relativizer as well as antecedent prediction. Overall, all models show better performance on grammatical than semantic knowledge (animacy and plausibility). Regarding relativizer prediction, all models perform worst on the target word *which* (plausible, as it is the most versatile of the relativizers). Comparing models, BERT is best at predicting the actual targets, RoBERTa outperforms the others in capturing grammatical and semantic knowledge, and ALBERT performs worst overall.

Evaluation on semantic types of antecedents shows prediction of unrelated or not directly related antecedents, especially for ALBERT in *which* RCs. Interestingly, both BERT and ALBERT predict hierarchically more general antecedents in *who* RCs, while RoBERTa is able to better capture specificity.

We believe that more work in this direction will lead to (a) awareness of the complexity of linguistic knowledge that such models might have to capture, considering e.g., generalization tasks, and (b) improve how evaluation strategies capture linguistic knowledge, e.g., through a combination of probing, diagnostic, and cloze tests, and also aid in developing evaluation best practices.

4

On the Interplay Between Fine-tuning and Probing

Contents

Fine-tuning pre-trained language models is an integral part of the modern NLP pipeline. Recently, probing has emerged as a way to investigate the linguistic

knowledge captured by pre-trained models. However, very little is understood about how fine-tuning affects the representations of pre-trained models and thereby the linguistic knowledge they encode. This chapter contributes towards closing this gap. We study three different pre-trained models: BERT, RoBERTa, and ALBERT, and investigate through sentence-level probing how fine-tuning affects the linguistic knowledge encoded in their representations. We find that for some probing tasks fine-tuning leads to substantial changes in accuracy, suggesting that fine-tuning introduces or even removes linguistic knowledge from a pre-trained model. These changes, however, vary greatly across different models, fine-tuning and probing tasks. Our analysis reveals that while fine-tuning does change the representations of a pre-trained model and these changes are typically larger for higher layers, it has a large positive effect on probing accuracy only in very few cases, and is often beaten by just using the pre-trained model with a strong pooling method. Based on our findings, we argue that both positive and negative effects of fine-tuning on probing results require a careful interpretation.

The content presented in this chapter is based on:

> **Mosbach, Marius**, Anna Khokhlova, Michael A. Hedderich, and Dietrich Klakow (2020). "On the Interplay Between Fine-tuning and Sentence-level Probing for Linguistic Knowledge in Pre-trained Transformers." In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, pp. 2502–2516. URL: https://aclanthology.org/2020.findings-emnlp.227.

As a first author, Marius Mosbach conceptualized the research, conducted most of the experiments and led the paper writing. Anna Khokhlova ran the perplexity experiments and helped with writing. Michael Hedderich suggested and ran the attention entropy experiment and helped with writing. Dietrich Klakow advised and provided feedback.

4.1 Introduction

Fine-tuned transformer-based language models like BERT (Devlin et al., 2019), RoBERTa (Y. Liu et al., 2019b) and ALBERT (Lan et al., 2020) recently became the state-of-the-art on a large variety of NLP downstream tasks. These models are pre-trained on large amounts of text and subsequently fine-tuned via supervised training on downstream tasks. Their strong empirical performance prompted questions about the linguistic knowledge they encode in their representations and how it is affected by the training objective and model architecture (Kim et al., 2019b; A. Wang et al., 2019c). One prominent technique to gain insights about the linguistic knowledge encoded in pre-trained models is *probing* (Adi et al., 2017; Tenney et al., 2019a; Conneau et al., 2018; Rogers et al., 2020, *inter alia*). However, to the best of our knowledge, works on probing have so far focused only on pre-trained models and it is still unclear how the representations of a pre-trained model change when fine-tuning on a downstream task. Further, little is known about whether and to what extent this process adds or removes linguistic knowledge from a pre-trained model. Addressing these issues, this chapter investigate the following questions:

1. How and where does fine-tuning affect the representations of a pre-trained model?

2. To which extent (if at all) can changes in probing accuracy be attributed to a change in linguistic knowledge encoded by the model?

To answer these questions, we investigate three different pre-trained masked language models: BERT, RoBERTa, and ALBERT. We fine-tune them on sentence-level classification tasks from the GLUE benchmark (A. Wang et al., 2018) and evaluate the linguistic knowledge they encode leveraging three sentence-level probing tasks from the SentEval probing suite (Conneau et al., 2018). We focus on sentence-level probing tasks to measure linguistic knowledge encoded by a model for two reasons: 1) during fine-tuning we explicitly train a model to represent

sentence-level context in its representations and 2) we are interested in the extent to which this affects existing sentence-level linguistic knowledge already present in a pre-trained model.

We find that while fine-tuning does affect a model's sentence-level probing accuracy and these effects are typically larger for higher layers, changes in probing accuracy vary depending on the model, fine-tuning and probing task combination. Our results also show that sentence-level probing accuracy is highly dependent on the pooling method being used. In fact, fine-tuning only has a greater positive effect on probing accuracy than just using the pre-trained model with a strong pooling method in very few cases. Our findings suggest that changes in probing performance cannot exclusively be attributed to an improved or deteriorated encoding of linguistic knowledge and should be carefully interpreted. We present further evidence for this interpretation by investigating changes in the attention distribution and language modeling capabilities of fine-tuned models which suggest alternative explanations for changes in probing accuracy.

4.2 Related work

Probing language models for linguistic knowledge A large body of previous work focuses on analyzing the internal representations of neural models and the linguistic knowledge they encode (Shi et al., 2016; Ettinger et al., 2016; Adi et al., 2016; Belinkov et al., 2017; Hupkes et al., 2018). In a similar spirit to these first works on probing, Conneau et al. (2018) were the first to compare different sentence embedding methods based on the linguistic knowledge they encode. Krasnowska-Kieraś and Wróblewska (2019) extended this approach to study sentence-level probing tasks on English and Polish sentences.

Alongside sentence-level probing, a lot of recent work (Peters et al., 2018; N. F. Liu et al., 2019; Tenney et al., 2019b; Lin et al., 2019; Hewitt and Manning, 2019) has focused on token-level probing tasks investigating more recent contextualized embedding models such as ELMo (Peters et al., 2018), GPT (Radford et al., 2019),

and BERT (Devlin et al., 2019). Two of the most prominent works following this methodology are N. F. Liu et al. (2019) and Tenney et al. (2019b). While N. F. Liu et al. (2019) use linear probing classifiers as we do, Tenney et al. (2019b) use more expressive, non-linear classifiers. However, in contrast to our work, most studies that investigate pre-trained contextualized embedding models focus on pre-trained models and not fine-tuned ones. We aim to assess how probing performance changes with fine-tuning and how these changes differ based on the model architecture, as well as probing and fine-tuning task combination.

Anaylzing fine-tuned language models While fine-tuning pre-trained language models leads to strong empirical performance across various supervised NLP downstream tasks (A. Wang et al., 2018), fine-tuning itself (Howard and Ruder, 2018; Devlin et al., 2019) and its effects on the representations learned by a pre-trained model are poorly understood. As an example, Phang et al. (2018) show that downstream accuracy can benefit from an intermediate fine-tuning task, but leave the investigation of why certain tasks benefit from intermediate task training to future work. Recently, Pruksachatkun et al. (2020b) extended this approach using 11 diverse intermediate fine-tuning tasks. They view probing task performance after fine-tuning as an indicator of the acquisition of a particular language skill during intermediate task fine-tuning. This is similar to our work in the sense that probing accuracy is used to understand how fine-tuning affects a pre-trained model. Talmor et al. (2019) try to understand whether the performance on downstream tasks should be attributed to the pre-trained representations or rather the fine-tuning process itself. They fine-tune BERT and RoBERTa on a large set of symbolic reasoning tasks and find that while RoBERTa generally outperforms BERT in its reasoning abilities, the performance of both models is highly context-dependent.

Most similar to our work is contemporaneous work by Merchant et al. (2020). They investigate how fine-tuning leads to changes in the representations of a pre-trained model. In contrast to our work, their focus, however, lies on edge-

probing (Tenney et al., 2019b) and structural probing tasks (Hewitt and Manning, 2019), and they study only a single pre-trained encoder: BERT. We consider our work complementary to theirs since we study sentence-level probing tasks, use different analysis methods and investigate the impact of fine-tuning on three different pre-trained encoders: BERT, RoBERTa, and ALBERT, providing a more comprehensive picture.

4.3 Methodology and setup

The focus of our work is on studying how fine-tuning affects the representations learned by a pre-trained model. We assess this change through sentence-level probing tasks. We focus on sentence-level probing tasks since during fine-tuning we explicitly train a model to represent sentence-level context in the CLS token.

The fine-tuning and probing tasks we study operate at different linguistic levels, requiring a model to focus more on syntactic, semantic or discourse information. The extent to which knowledge of a particular linguistic level is needed to perform well differs from task to task. For instance, no deep discourse understanding is needed to judge if the syntactic structure of a sentence is intact. Our hypothesis is that if a pre-trained model encodes certain linguistic knowledge, this acquired knowledge should lead to good performance on a probing task testing for the same linguistic phenomenon. Extending this hypothesis to fine-tuning, one might argue that if fine-tuning introduces new or removes existing linguistic knowledge into/from a model, this should be reflected by an increase or decrease in probing performance. In fact, examples of this reasoning can be found in previous work: Merchant et al. (2020) find that fine-tuning on the task of dependency parsing leads to an improvement on constituents probing, and attribute this to the improved linguistic knowledge of the fine-tuned model. Similarly, Pruksachatkun et al. (2020b) view probing task performance as "an indicator for the acquisition of a particular language skill." We argue that *encoding or forgetting linguistic knowledge is not necessarily the only explanation for observed changes in probing*

accuracy. Hence, the goal of our work is to test the above-stated hypotheses assessing the interaction between fine-tuning and probing tasks across three different encoder models.

4.3.1 Fine-tuning tasks

We study models fine-tuned on the following datasets taken from the GLUE benchmark (A. Wang et al., 2018): CoLA, SST-2, and RTE. All three datasets are sentence-level classification tasks and cover different levels of linguistic proficiency (we elaborate on this below). In addition to the three classification datasets, we also study models fine-tuned on SQuAD (Rajpurkar et al., 2016), a popular question answering dataset. Statistics for each of the datasets can be found in the Appendix B.1. Below, we provide details about each of the datasets.

CoLA The Corpus of Linguistic Acceptability (Warstadt et al., 2018) is an acceptability dataset which tests a model's knowledge of grammatical concepts. We expect that fine-tuning on CoLA would result in changes in accuracy on syntactic probing tasks, as CoLA contains sentences with syntactic, morphological and semantic violations. We identified that about 15% of the sentences in the CoLA training set are labeled with morphological and semantic violations. Thus, we suppose that fine-tuning on CoLA should increase a model's sensitivity to syntactic violations to a greater extent than, e.g., semantic violations.

SST-2 We use the binary version of the Stanford Sentiment Treebank (Socher et al., 2013) where the task is to categorize movie reviews as having either positive or negative valence. Making sentiment judgments requires knowing the meanings of isolated words and combining them on the sentence and discourse level (e.g., in case of irony). Hence, we expect to see a difference for semantic and/or discourse probing tasks when fine-tuning on SST-2.

RTE The Recognizing Textual Entailment dataset is a collection of sentence-pairs in either neutral or entailment relationship collected from a series of annual textual entailment challenges (Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009). The task requires a deeper understanding of the relationship of two sentences, hence, fine-tuning on RTE might affect accuracy on discourse-level probing tasks.

SQuAD The Stanford Questions Answering Dataset (Rajpurkar et al., 2016) is a popular extractive reading comprehension dataset. The task involves a broader discourse understanding as a model trained on SQuAD is required to extract the answer to a question from an accompanying paragraph.

4.3.2 Probing tasks

We select three sentence-level probing tasks from the SentEval probing suite (Conneau et al., 2018), testing for syntactic, semantic and broader discourse information on the sentence-level.

Bigram-shift is a syntactic binary classification task that tests a model's sensitivity to word order. The dataset consists of intact and corrupted sentences, where for corrupted sentences, two random adjacent words have been inverted, e.g., ``She wondered how time much had passed."

Semantic-odd-man-out tests a model's sensitivity to semantic incongruity on a collection of sentences where random verbs or nouns are replaced by another verb or noun, e.g., ``I managed to block out everything, and loud music, governing bodies, and thumping bass were all I cared about."

Coordination-inversion is a collection of sentences made out of two co-ordinate clauses. In half of the sentences, the order of the clauses is inverted,

, e.g., ``I can't and I'll have to deal with them." Coordinate-inversion tests for a model's broader understanding of discourse.

4.3.3 Pre-trained models

It is unclear to what extent findings on the encoding of certain linguistic phenomena generalize from one pre-trained model to another. Hence, we examine three different pre-trained encoder models in our experiments: BERT (Devlin et al., 2019), RoBERTa (Y. Liu et al., 2019b), and ALBERT (Lan et al., 2020). We introduce all three models in greater detail in §2.3.2 and note that we focus on the base variant for each of the models.

4.3.4 Fine-tuning and probing setup

For fine-tuning and sentence-level probing, we follow the general methodologies introduced in §2.4.1 and §2.5 and provide additional details below.

Fine-tuning For fine-tuning, we follow the default vanilla fine-tuning approach proposed by Devlin et al. (2019). A single randomly initialized task-specific classification layer is added on top of the pre-trained encoder. As input, the classification layer receives $\mathbf{z} = \tanh(\mathbf{W}\mathbf{h} + \mathbf{b})$, where $\mathbf{h}$ is the hidden representation of the first token on the last hidden layer and $\mathbf{W}$ and $\mathbf{b}$ are the randomly initialized parameters of the classifier.[1] During fine-tuning all model parameters are updated jointly. We train for 3 epochs on CoLA and for 1 epoch on SST-2, using a learning rate of 2e−5. The learning rate is linearly increased for the first 10% of steps (warmup) and kept constant afterwards. An overview of all hyperparameters for

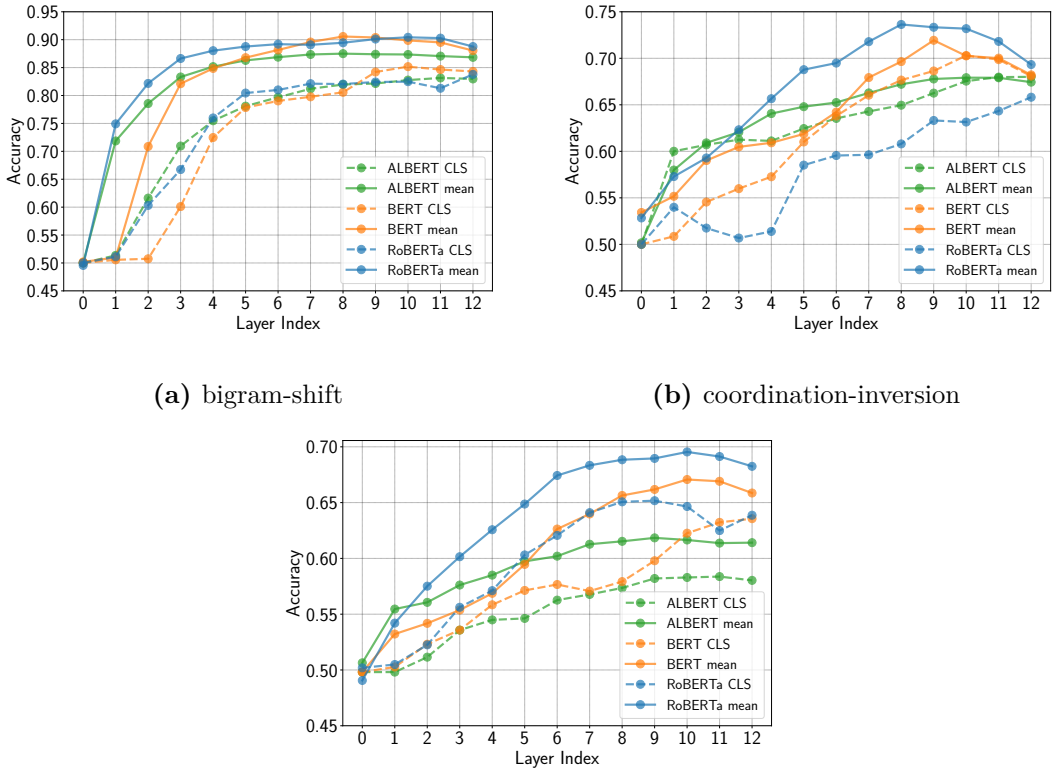
1 For BERT and ALBERT $\mathbf{h}$ corresponds to the hidden state of the [CLS] token. For RoBERTa the first token of every sentence is the <s> token. We will refer to both of them as CLS token.

Model	Task			
	CoLA	SST-2	RTE	SQuAD
Devlin et al. (2019)	52.1	93.5	66.4	80.8/88.5
BERT	59.5	92.4	64.6	78.6/86.5
RoBERTa	60.3	93.6	73.6	81.7/89.3
ALBERT	45.8	88.5	69.6	79.9/87.6

Table 4.1: Fine-tuning performance on the development set on select downstream tasks. For comparison we also report the fine-tuning accuracy of BERT-base-cased as reported by Devlin et al. (2019) on the test set of each of the tasks taken from the GLUE and SQuAD leaderboards. We report Matthews correlation coefficient for CoLA, accuracy for SST-2 and RTE, and exact match (EM) and F_1 score for SQuAD.

each model and task can be found in Appendix B.1. Fine-tuning performance on the development set of each of the tasks can be found in Table 4.1.

Probing For probing, our setup largely follows that of previous works (Tenney et al., 2019b; N. F. Liu et al., 2019; Hewitt and Liang, 2019) where a *probing classifier* is trained on top of the contextualized embeddings extracted from a pre-trained or—as in our case—-fine-tuned encoder model. Notably, we train *linear* (logistic regression) probing classifiers and use two different *pooling methods* to obtain sentence embeddings from the encoder hidden states: **CLS-pooling**, which simply returns the hidden state corresponding to the first token of the sentence and **mean-pooling** which computes a sentence embedding as the mean over all hidden states. We do this to assess the extent to which the CLS token captures sentence-level context. We use linear probing classifiers because intuitively we expect that if a linguistic feature is useful for a fine-tuning task, it should be linearly separable in the embeddings. For all probing tasks, we measure layer-wise accuracy to investigate how the linear separability of a particular linguistic phenomenon changes across the model. In total, we train 390 probing classifiers on top of 12 pre-trained and fine-tuned encoder models.



(a) bigram-shift

(b) coordination-inversion

(c) odd-man-out

Figure 4.1: Layer-wise probing accuracy on bigram-shift, coordination inversion, and odd-man-out for BERT, RoBERTa, and ALBERT. For all models mean-pooling (solid lines) consistently improves probing accuracy compared to CLS-pooling (dashed lines), highlighting the importance of sentence-level information for each of the tasks.

4.4 Experiments

4.4.1 Probing accuracy

Figure 4.1 shows the layer-wise probing accuracy of BERT, RoBERTa, and ALBERT on each of the probing tasks. These results establish baselines for our comparison with fine-tuned models below. Consistent with previous work (Krasnowska-Kieraś and Wróblewska, 2019), we observe that mean-pooling gener-

ally outperforms CLS-pooling across all probing tasks, highlighting the importance of sentence-level context for each of the probing tasks. We also find that probing accuracy is substantially higher for *bigram-shift* than for *coordination-inversion* and *odd-man-out*. Again, this is consistent with findings in previous works (Tenney et al., 2019b; N. F. Liu et al., 2019; Tenney et al., 2019a) reporting better performance on syntactic than semantic probing tasks.

When comparing the three encoder models, we observe some noticeable differences. On *odd-man-out*, ALBERT performs significantly worse than both BERT and RoBERTa, with RoBERTa performing best across all layers. We attribute the poor performance of ALBERT to its use of weight-sharing, effectively applying the same non-linear transformation on all layers. We also observe that on *coordination-inversion*, RoBERTa with CLS pooling performs much worse than both BERT and ALBERT with CLS pooling. We attribute this to the fact that RoBERTa lacks a sentence-level pre-training objective and the CLS token hence fails to capture relevant sentence-level information for this particular probing task. The small differences in probing accuracy for BERT and ALBERT when comparing CLS to mean-pooling and the fact that RoBERTa with mean-pooling outperforms all other models on *coordination-inversion* provides evidence for this interpretation.

4.4.2 How does fine-tuning affect probing accuracy?

Having established baselines for the probing accuracy of the pre-trained models, we now turn to the question of how it is affected by fine-tuning. Table 4.2 shows the effect of fine-tuning on CoLA and SST-2 on the layer-wise accuracy for all three encoder models across the three probing tasks. Results for RTE and SQuAD can be found in Table B.3 in the Appendix.

For all models and tasks we find that *fine-tuning mostly has an effect on higher layers, sometimes positively and sometimes negatively affecting accuracy.*

Probing Task	**BERT-base-cased**							
	CLS-pooling				mean-pooling			
	CoLA		SST-2		CoLA		SST-2	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	0.07	4.73	−1.02	−4.63	0.23	1.45	−0.37	−3.24
coordinate-inversion	−0.10	1.90	−0.25	−1.15	0.14	0.29	−0.48	−0.85
odd-man-out	−0.20	0.26	−0.02	−1.28	−0.34	−0.29	−0.30	−1.09
Probing Task	**RoBERTa-base**							
	CLS-pooling				mean-pooling			
	CoLA		SST-2		CoLA		SST-2	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	0.58	5.35	−2.41	−7.22	0.69	1.74	−0.23	−4.87
coordinate-inversion	−0.72	1.84	−1.28	−0.63	−0.22	0.02	−0.18	−3.83
odd-man-out	−0.66	1.05	−1.09	−2.40	−0.08	−0.55	−0.46	−3.61
Probing Task	**ALBERT-base-v1**							
	CLS-pooling				mean-pooling			
	CoLA		SST-2		CoLA		SST-2	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	1.55	3.39	−1.94	−5.15	0.26	0.66	−0.70	−2.73
coordinate-inversion	−0.69	−1.53	−1.07	−2.87	−0.07	−1.19	−0.35	−1.53
odd-man-out	−0.42	−1.39	−0.90	−2.75	−0.27	−1.40	−0.60	−2.82

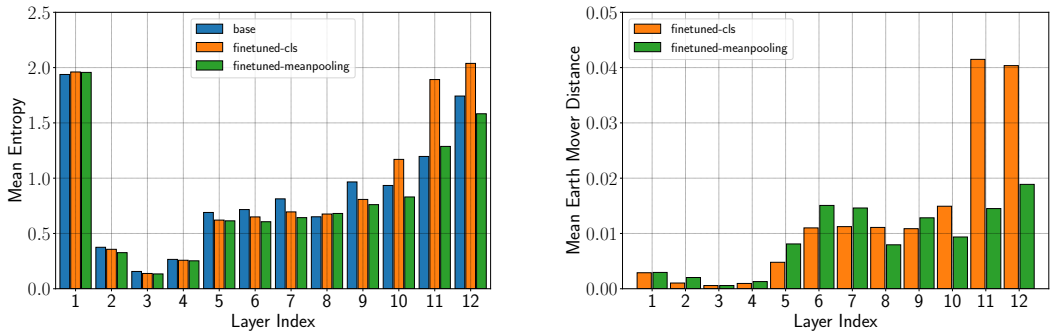
Table 4.2: Change in probing accuracy (in %) of **CoLA** and **SST-2** fine-tuned models compared to the pre-trained models when using CLS and mean-pooling. We average the difference in probing accuracy over two different layer groups: layers 0 to 6 and layers 7 to 12.

The impact varies depending on the fine-tuning/probing task combination and the underlying encoder model.

Positive changes in accuracy Fine-tuning on CoLA results in a substantial improvement on the *bigram-shift* probing task for all the encoder models; fine-tuning on RTE improves the *coordination-inversion* accuracy for RoBERTa. These findings are in line with our expectations: *bigram-shift* and CoLA require syntactic information, whereas *coordination-inversion* and RTE require deeper discourse-level understanding. However, a more detailed look reveals a problem with this logic: the improvement is only visible when using CLS-pooling and becomes negligible when probing with mean-pooling. Moreover, the gains are not large enough to improve significantly over the mean-pooling baseline (as shown by the stars and the second y-axis in Figure B.1 in Appendix B). This suggests that adding new linguistic knowledge is not necessarily the *only* driving force behind the improved probing accuracy and we provide further evidence for this in §4.5.1.

Negative changes in accuracy Across all models and pooling methods, fine-tuning on SST-2 has a negative impact on probing accuracy on *bigram-shift* and *odd-man-out*, and the decrease in probing accuracy is particularly large for RoBERTa. Fine-tuning on SQuAD follows a similar trend: it has a negative effect on probing accuracy on *bigram-shift* and *odd-man-out* for both CLS- and mean-pooling (see Table B.3 in Appendix B), while the impact on *coordination-inversion* is negligible. We argue that this strong negative impact on probing accuracy is the consequence of more dramatic changes in the representations and further investigate this observation in §4.5.2.

Changes in probing accuracy for other fine-tuning/probing combinations are not substantial, which suggests that representations did not change significantly with regard to the probed information.



(a) Entropy

(b) Earth Mover's Distance

Figure 4.2: Entropy and Earth Mover's Distance of the attention for the CLS token for each layer with the RoBERTa model on the bigram-shift dataset. The mean over all input sequences and the mean over all attention heads of a layer are taken. The Earth Mover's Distance is computed between the base model and each fine-tuned model.

4.5 What happens during fine-tuning?

In the previous section, we saw the effects of different fine-tuning approaches on model performance. In this section, we study two hypotheses about what causes these changes.

4.5.1 Analyzing attention distributions

If the improvement in probing accuracy with CLS-pooling can be attributed to a better sentence representation in the CLS token, this can be due to a corresponding change in a model's attention distribution. The model might change the attention of the CLS token to cover more tokens and thereby build a better representation for the entire sentence.

To study this hypothesis, we fine-tune RoBERTa on CoLA using two different methods: the default CLS-pooling approach and mean-pooling (cf. §4.3.4). We compare the layer-wise attention distribution on bigram-shift after fine-tuning

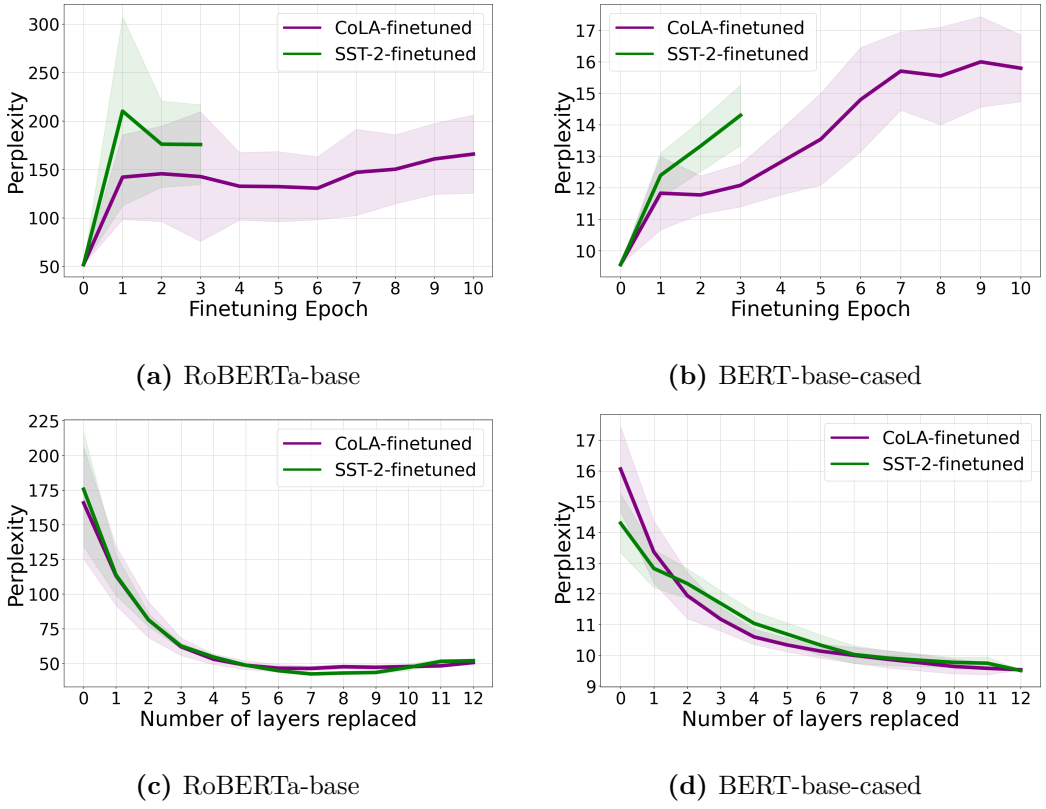
to that data. We expect to see more dramatic changes for CLS-pooling than for mean-pooling. To investigate how the attention distribution changes, we analyze its entropy H according to the following equation:

$$H_j = -\sum_i a_j(x_i) \cdot log\left(a_j(x_i)\right) \tag{4.1}$$

where x_i is the i-th token of an input sequence and $a(x_i)$ the corresponding attention at position j given to it by a specific attention head. Entropy is maximal when the attention is uniform over the whole input sequence and minimal if the attention head focuses on just one input token.

Figure 4.2a shows the mean entropy for the CLS token (i.e., H_0) before and after fine-tuning. We observe a large increase in entropy in the last three layers when fine-tuning on the CLS token (orange bars). This is consistent with our interpretation that during fine-tuning, the CLS token learns to take more sentence-level information into account, thus spreading its attention over more tokens. For mean-pooling (green bars) this might not be required as taking the mean over all token-states could already provide sufficient sentence-level information during fine-tuning. Accordingly, there are only small changes in entropy for mean-pooling, with the mean entropy actually decreasing in the last layer.

Entropy is, however, not sufficient on its own to analyze changes in the attention distribution. Even when the amount of entropy is similar, the underlying attention distribution might have changed. Figure 4.2b therefore compares the attentions of an attention head for an input sequence before and after fine-tuning using *Earth Mover's Distance* (Rubner et al., 1998). We find a similar pattern to the entropy results where changes in attention tend to increase with the layer number. Again, the largest change in the attention distribution occurs in the first token for layer 11 and 12 when pooling on the CLS-token, while the change is much smaller for mean-pooling. This confirms our hypothesis that improvements in fine-tuning with CLS-pooling can be attributed to a change in the attention distribution which is less necessary for the mean-pooling.



(a) RoBERTa-base

(b) BERT-base-cased

(c) RoBERTa-base

(d) BERT-base-cased

Figure 4.3: Perplexity on Wikitext-2 of models consisting of a fine-tuned encoder and a pre-trained MLM head. Plots (a) and (b) show how perplexity changes over the course of fine-tuning with epoch 0 showing the perplexity of the pre-trained model. (c) and (d) show how perplexity changes when a number of last layers of the fine-tuned encoder are replaced with corresponding layers from the pre-trained model. Note the different y-axes for RoBERTa and BERT.

4.5.2 Analyzing perplexity

If fine-tuning has more dramatic effects on the representations of a pre-trained model potentially introducing or removing linguistic knowledge, we expect to see larger changes to the language modeling abilities of the model when compared to the case where fine-tuning just changes the attention distribution of the CLS token. We thus investigate how fine-tuning on CoLA and SST-2 affect the language modeling abilities of a pre-trained model. A change in perplexity should reveal

whether the representations of the model did change during fine-tuning and we expect this change to be larger for SST-2 fine-tuning where we observe a large decrease in probing accuracy.

For the first experiment, we evaluate the pre-trained masked language model heads of BERT and RoBERTa on the Wikitext-2 test set (Merity et al., 2017) and compare it to the masked language modeling perplexity, hereafter perplexity, of fine-tuned models.[2] In the second experiment, we test which layers contribute most to the change in perplexity and replace layers of the fine-tuned encoder by pre-trained layers, starting from the last layer. For both experiments, we evaluate the perplexity of the resulting model using the pre-trained masked language modeling head. We fine-tune and evaluate each model 5 times, and report the mean perplexity as well as standard deviation. Our reasoning is that if fine-tuning leads to dramatic changes to the hidden representations of a model, the effects should be reflected in the perplexity.

Perplexity during fine-tuning Figure 4.3a and 4.3b show how the perplexity of a pre-trained model changes during fine-tuning. Both BERT and RoBERTa show a similar trend where perplexity increases with fine-tuning. Interestingly, for RoBERTa the increase in perplexity after the first epoch is much larger compared to BERT. Additionally, our results show that for both models the increase in perplexity is larger when fine-tuning on SST-2. This confirms our hypothesis and also our findings from §4.4, suggesting that fine-tuning on SST-2 has more dramatic effects on the representations of both models compared to fine-tuning on CoLA.

Perplexity when replacing fine-tuned layers While fine-tuning leads to worse language modeling abilities for both CoLA and SST-2, it is not clear from the first experiment alone which layers are responsible for the increase in perplexity.

2 Note that perplexity results are not directly comparable between BERT and RoBERTa since both models have different vocabularies. However, we are interested in how perplexity changes with fine-tuning.

Figure 4.3c and 4.3d show the perplexity results when replacing fine-tuned layers with pre-trained ones starting from the last hidden layer. Consistent with our probing results in §4.4, we find that *the changes that lead to an increase in perplexity happen in the last layers*, and this trend is the same for both BERT and RoBERTa. Interestingly, we observe no difference between CoLA and SST-2 fine-tuning in this experiment.

4.5.3 Discussion

The main implications of our experiments and analyses are as follows.

Fine-tuning affects mostly upper layers We conclude that fine-tuning does affect the representations of a pre-trained model and in particular those of the last hidden layers, which is supported by our perplexity analysis. However, our perplexity analysis does not reveal whether these changes have a positive or negative effect on the encoding of linguistic knowledge.

For CLS-pooling, fine-tuning can improve probing performance Some fine-tuning/probing task combinations result in substantial improvements in probing accuracy when using CLS-pooling. Our attention analysis supports our interpretation that the improvement in probing accuracy can not simply be attributed to the encoding of linguistic knowledge, but can at least partially be explained by changes in the attention distribution for the CLS token. We note that this is also consistent with our findings that the improvement in probing accuracy vanishes when compared to the mean-pooling baseline.

The negative impact of fine-tuning on probin performance is hard to interpret Some other task combinations have a negative effect on the probing task performance, suggesting that the linguistic knowledge our probing classifiers are testing for is indeed no longer (linearly) accessible. However, it remains unclear

whether fine-tuning removes the linguistic knowledge our probing classifiers are testing for from the representations or whether it is simply no longer linearly separable.

4.6 Conclusion

We investigated the interplay between fine-tuning and layer-wise sentence-level probing accuracy and found that fine-tuning can lead to substantial changes in probing accuracy. However, these changes vary greatly depending on the encoder model and fine-tuning and probing task combination. Our analysis of attention distributions after fine-tuning showed that changes in probing accuracy cannot be attributed to the encoding of linguistic knowledge alone but might also be caused by changes in the attention distribution. At the same time, our perplexity analysis showed that fine-tuning strongly affects the representations of a pre-trained model but our probing analysis is not sufficient to determine whether this leads to forgetting of the probed linguistic information. Hence we argue that the effects of fine-tuning on pre-trained representations should be carefully interpreted.

5

Investigating Fine-tuning Stability

Contents

Fine-tuning pre-trained transformer-based language models has become a common practice dominating leaderboards across various NLP benchmarks. Despite the strong empirical performance of fine-tuned models, fine-tuning itself is an unstable process: training the same model with multiple random seeds can result in a large variance of the task performance. Previous work (Devlin et al., 2019; Lee et al., 2020; Dodge et al., 2020a) identified two potential reasons for the observed instability: catastrophic forgetting, and the small size of the fine-tuning

datasets. However, both hypotheses are anecdotal and no empirical work has yet substantiated these claims.

In this chapter, we address this challenge by showing that both hypotheses fail to explain the observed fine-tuning instability. We analyze three widely used pre-trained language models (BERT, RoBERTa, and ALBERT) fine-tuned on several datasets from the GLUE benchmark, and show that fine-tuning instability is caused by optimization difficulties during training that lead to vanishing gradients. Additionally, we show that the remaining variance of the downstream task performance can be attributed to differences in generalization where fine-tuned models with the same training loss exhibit noticeably different test performance. Based on our analysis, we present a simple but strong baseline that makes fine-tuning pre-trained language models significantly more stable than the previously proposed approaches while maintaining or even improving overall performance.

The content presented in this chapter is based on:

> **Mosbach, Marius**, Maksym Andriushchenko, and Dietrich Klakow (2021). "On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines." In: *International Conference on Learning Representations: ICLR 2021*. Online. URL: https://openreview.net/forum?id=nzpLWnVAyah.

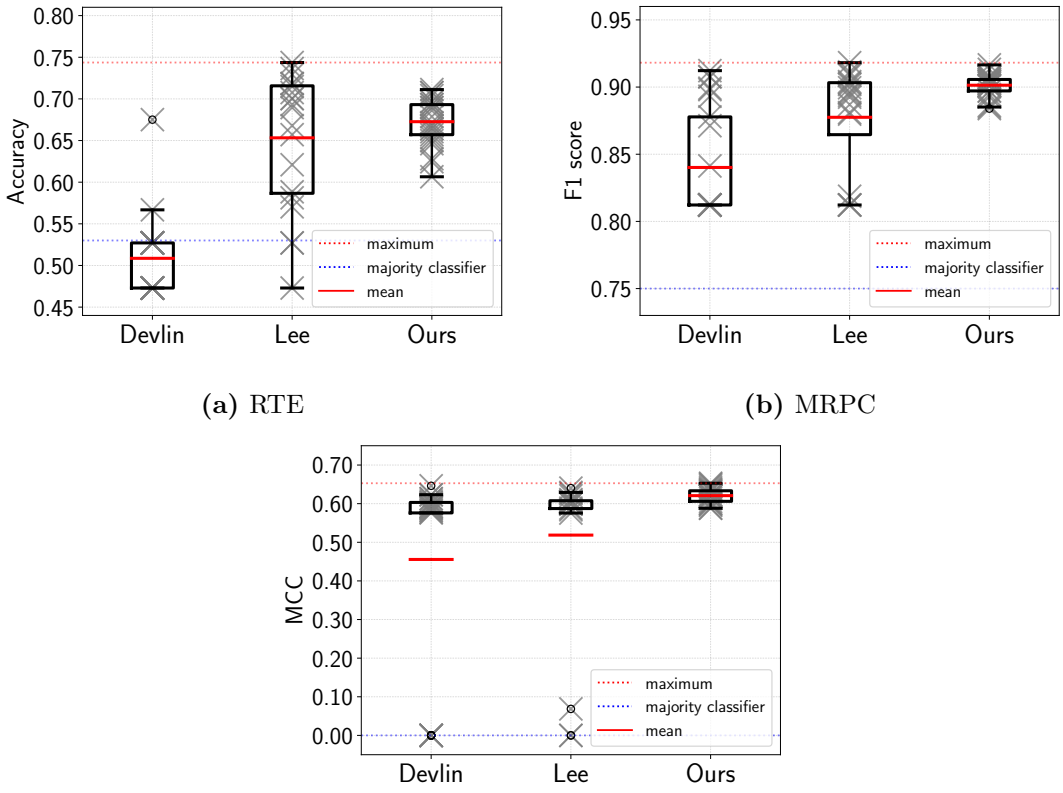
As a first author, Marius Mosbach conceptualized the research, conducted the experiments and led the paper writing. Maksym Andriushchenko helped to shape the research questions, provided feedback and helped with the writing. Dietrich Klakow advised and provided feedback.

5.1 Introduction

Pre-trained transformer-based masked language models such as BERT (Devlin et al., 2019), RoBERTa (Y. Liu et al., 2019b), and ALBERT (Lan et al., 2020) have had a dramatic impact on the NLP landscape in the recent year. The standard recipe for using such models typically involves adapting a pre-trained model for a few epochs on a supervised downstream dataset, which is known as *fine-tuning*. While fine-tuning has led to impressive empirical results, dominating a large variety of English NLP benchmarks such as GLUE (A. Wang et al., 2018) and SuperGLUE (A. Wang et al., 2019a), it is still poorly understood. Not only have fine-tuned models been shown to pick up spurious patterns and biases present in the training data (Niven and Kao, 2019; T. McCoy et al., 2019), but also to exhibit a large training *instability*: fine-tuning a model multiple times on the same dataset, varying only the random seed, leads to a large standard deviation of the fine-tuning accuracy (Devlin et al., 2019; Dodge et al., 2020a). Few methods have been proposed to solve the observed instability (Phang et al., 2018; Lee et al., 2020), however without providing a sufficient understanding of why fine-tuning is prone to such failure. The goal of this chapter is to address this shortcoming. More specifically, we investigate the following question:

Why is fine-tuning prone to failures and how can we improve its stability?

We start by investigating two common hypotheses for fine-tuning instability: catastrophic forgetting and the small size of the fine-tuning datasets and demonstrate that both hypotheses fail to explain fine-tuning instability. We then investigate fine-tuning failures on datasets from the popular GLUE benchmark and show that the observed fine-tuning instability can be decomposed into two separate aspects: (1) optimization difficulties early in training, characterized by vanishing gradients, and (2) differences in generalization late in training, characterized by a large variance of development set accuracy for runs with almost equivalent training loss.



(a) RTE **(b)** MRPC

(c) CoLA

Figure 5.1: Our proposed fine-tuning strategy leads to very stable results with very concentrated development set performance over 25 different random seeds across all three datasets on BERT. In particular, we significantly outperform the recently proposed approach of Lee et al. (2020) in terms of fine-tuning stability.

Based on our analysis, we present a simple but strong baseline for fine-tuning pre-trained language models that significantly improves the fine-tuning stability compared to previous works (Figure 5.1). Moreover, we show that our findings apply not only to the widely used BERT model but also to more recent pre-trained models such as RoBERTa and ALBERT.

5.2 Related work

The fine-tuning instability of BERT has been pointed out in various studies. Devlin et al. (2019) report instabilities when fine-tuning BERT-large on small datasets and resort to performing multiple restarts of fine-tuning and selecting the model that performs best on the development set. Recently, Dodge et al. (2020a) performed a large-scale empirical investigation of the fine-tuning instability of BERT. They found dramatic variations in fine-tuning accuracy across multiple restarts and argue how it might be related to the choice of random seed and the dataset size.

Few approaches have been proposed to directly address the observed fine-tuning instability. Phang et al. (2018) study intermediate task training (STILTS) before fine-tuning with the goal of improving performance on the GLUE benchmark. They also find that their proposed method leads to improved fine-tuning stability. However, due to the intermediate task training, their work is not directly comparable to ours. Lee et al. (2020) propose a new regularization technique termed Mixout. The authors show that Mixout improves stability during fine-tuning which they attribute to the prevention of catastrophic forgetting.

Another line of work investigates optimization difficulties of *pre-training* transformer-based language models (Xiong et al., 2020; L. Liu et al., 2020). Similar to our work, they highlight the importance of the learning rate warmup for optimization. Both works focus on pre-training and we hence view them as orthogonal to our work.

5.3 Datasets

We study four datasets from the GLUE benchmark (A. Wang et al., 2018) following previous work studying instability during fine-tuning: CoLA, MRPC, RTE, and

QNLI. Detailed statistics for each of the datasets can be found in Appendix C.2 in Appendix C.

CoLA The Corpus of Linguistic Acceptability (CoLA) (Warstadt et al., 2018) is a sentence-level classification task containing sentences labeled as either grammatical or ungrammatical. Fine-tuning on CoLA was observed to be particularly stable in previous work (Phang et al., 2018; Dodge et al., 2020a; Lee et al., 2020). Performance on CoLA is reported in Matthew's correlation coefficient (MCC).

MRPC The Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005) is a sentence-pair classification task. Given two sentences, a model has to judge whether the sentences paraphrases of each other. Performance on MRPC is measured using the F_1 score.

RTE The Recognizing Textual Entailment (RTE) dataset is a collection of sentence-pairs collected from a series of textual entailment challenges (Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009). RTE is the second smallest dataset in the GLUE benchmark and fine-tuning on RTE was observed to be particularly unstable (Phang et al., 2018; Dodge et al., 2020a; Lee et al., 2020). Accuracy is used to measure performance on RTE.

QNLI The Question-answering Natural Language Inference (QNLI) dataset contains sentence pairs obtained from SQuAD (Rajpurkar et al., 2016). A. Wang et al., 2018 converted SQuAD into a sentence pair classification task by forming a pair between each question and each sentence in the corresponding paragraph. The task is to determine whether the context sentence contains the answer to the question, i.e. entails the answer. Accuracy is used to measure performance on QNLI.

5.4 Fine-tuning

Unless mentioned otherwise, we follow the vanilla fine-tuning strategy introduced in §2.4.1 and use the hyperparameters recommended by Devlin et al., 2019: we fine-tune uncased BERT-large (henceforth BERT) using a batch size of 16 and a learning rate of 2e−5. The learning rate is linearly increased from 0 to 2e−5 for the first 10% of iterations—which is known as a *warmup*—and linearly decreased to 0 afterward. We apply dropout with probability $p = 0.1$ and weight decay with $\lambda = 0.01$. We train for 3 epochs on all datasets and use global gradient clipping. Following Devlin et al., 2019, we use the AdamW optimizer (Loshchilov and Hutter, 2019) *without* bias correction.

We do not report results for BERT-base since previous works observed no instability when fine-tuning BERT-base which we also confirmed in preliminary experiments. Instead, we provide additional results on RoBERTa-large (Y. Liu et al., 2019b) and ALBERT-large (Lan et al., 2020) using the same fine-tuning strategy. We note that compared to BERT, both RoBERTa and ALBERT have slightly different hyperparameters. RoBERTa uses weight decay with $\lambda = 0.1$ and no gradient clipping, and ALBERT does not use dropout. A detailed list of all default hyperparameters for all models can be found in Appendix C.3.

Fine-tuning stability. By *fine-tuning stability* we mean the standard deviation of the fine-tuning performance (measured, e.g., in terms of accuracy, MCC or F_1 score) over the randomness of an algorithm. We follow previous works (Phang et al., 2018; Dodge et al., 2020a; Lee et al., 2020) and measure fine-tuning stability using the development sets from the GLUE benchmark. We discuss alternative notions of stability in Appendix C.1 in the Appendix.

Failed runs. Following Dodge et al. (2020a), we refer to a fine-tuning run as a *failed run* if its accuracy at the end of training is less or equal to that of a

majority classifier on the respective dataset. The majority baselines for all tasks are found in Appendix C.2.

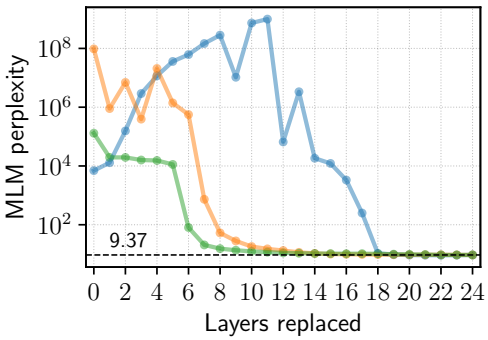
5.5 Investigating previous hypotheses for fine-tuning instability

Previous works on fine-tuning predominantly state two hypotheses for what can be related to fine-tuning instability: *catastrophic forgetting* and *small training data size* of the downstream tasks. Despite the ubiquity of these hypotheses (Devlin et al., 2019; Phang et al., 2018; Dodge et al., 2020a; Lee et al., 2020), we argue that none of them has a causal relationship with fine-tuning instability.

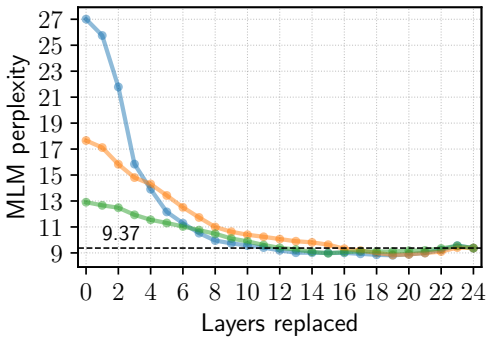
5.5.1 Does catastrophic forgetting cause fine-tuning instability?

Catastrophic forgetting (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017) refers to the phenomenon when a neural network is sequentially trained to perform two different tasks, and it loses its ability to perform the first task after being trained on the second. More specifically, in our setup, it means that after fine-tuning a pre-trained model, it can no longer perform the original masked language modeling task used for pre-training. This can be measured in terms of the perplexity on the original training data. Although the language modeling performance of a pre-trained model correlates with its fine-tuning accuracy (Y. Liu et al., 2019b; Lan et al., 2020), there is no clear motivation for why preserving the original masked language modeling performance after fine-tuning is important.[1]

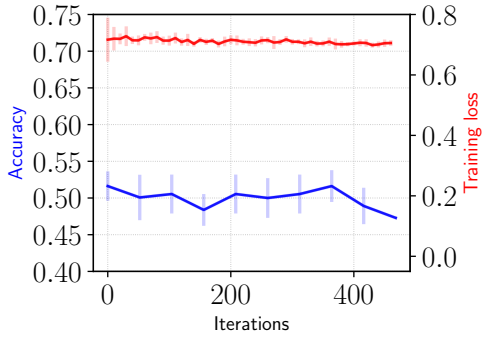
1 An exception could by the case where supervised fine-tuning is performed as an intermediate training step, e.g. with the goal of domain adaptation.



(a) Perplexity of failed models



(b) Perplexity of successful models



(c) Training of failed models

Figure 5.2: Language modeling perplexity for three failed (a) and successful (b) fine-tuning runs of BERT on RTE where we replace the weights of the top-k layers with their pre-trained values. (c) shows the average training loss and validation accuracy for three failed and three successful fine-tuning runs.

In the context of fine-tuning BERT, Lee et al., 2020 suggest that their regularization method has an effect of alleviating catastrophic forgetting. Thus, it is important to understand how exactly catastrophic forgetting occurs during fine-tuning and how it relates to the observed fine-tuning instability. To better understand this, we perform the following experiment: we fine-tune BERT on RTE, following the default strategy by Devlin et al., 2019. We select three successful and three failed fine-tuning runs and evaluate their masked language modeling perplexity on the test set of the WikiText-2 language modeling benchmark (Merity

et al., 2016).[2] We sequentially substitute the top-k layers of the network varying k from 0 (i.e. all layers are from the fine-tuned model) to 24 (i.e. all layers are from the pre-trained model). We show the results in Figures 5.2a and 5.2b.

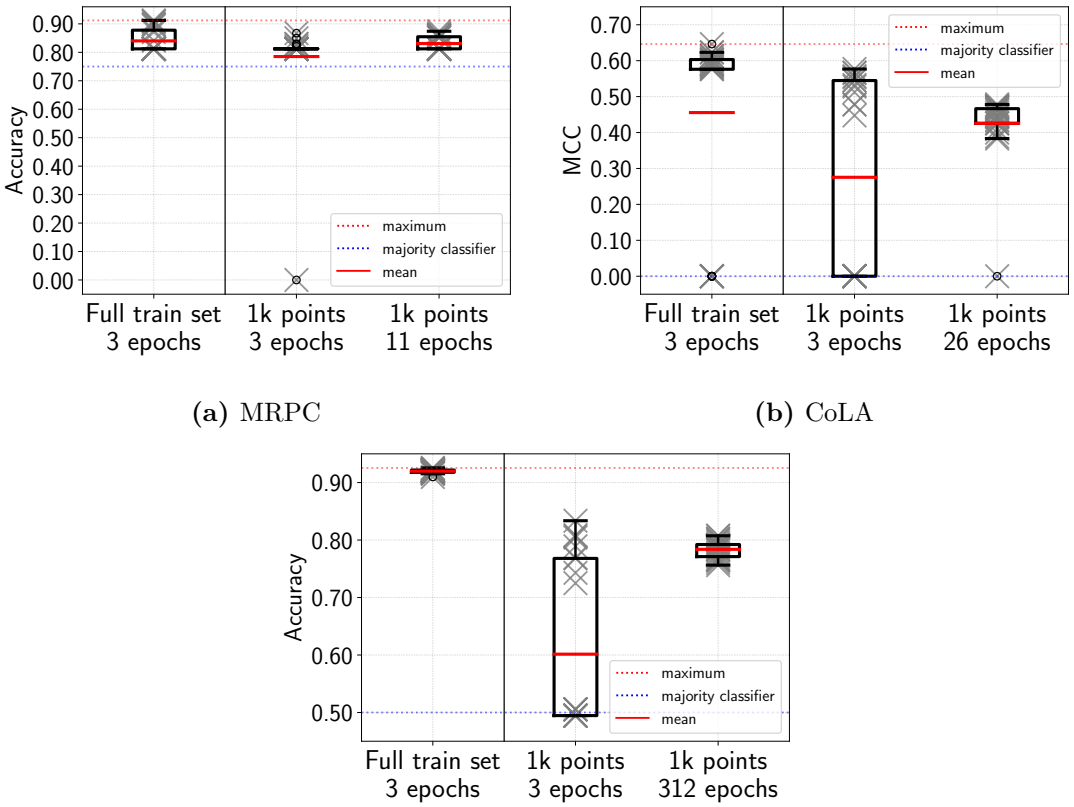
We can observe that although catastrophic forgetting occurs for the failed models (Figure 5.2a)—perplexity on WikiText-2 is indeed degraded for $k = 0$—the phenomenon is much more nuanced. Namely, catastrophic forgetting affects only the top layers of the network, in our experiments often around 10 out of 24 layers, and the same is however also true for the successfully fine-tuned models, except for a much smaller increase in perplexity.

Another important aspect of our experiment is that catastrophic forgetting typically requires that the model at least successfully learns how to perform the new task. However, this is not the case for the failed fine-tuning runs. Not only is the development accuracy equal to that of the majority classifier, but also the training loss on the fine-tuning task (here RTE) is trivial, i.e. close to $-\ln(1/2)$ (see Figure 5.2c). This suggests that the observed fine-tuning failure is rather an optimization problem *causing* severe catastrophic forgetting in the top layers of the pre-trained model. We will show later that the optimization aspect is actually sufficient to explain most of the fine-tuning variance.

5.5.2 Do small datasets cause fine-tuning instability?

Having a small training dataset is by far the most commonly stated hypothesis for fine-tuning instability. Multiple recent works (Devlin et al., 2019; Phang et al., 2018; Lee et al., 2020; C. Zhu et al., 2020; Dodge et al., 2020a; Pruksachatkun et al., 2020b) that have observed BERT fine-tuning to be unstable relate this finding to the small number of training examples.

2 BERT was trained on English Wikipedia, hence WikiText-2 can be seen as a subset of its training data.



(a) MRPC **(b)** CoLA

(c) QNLI

Figure 5.3: Development set results on down-sampled MRPC, CoLA, and QNLI using the default fine-tuning scheme of BERT (Devlin et al., 2019). The leftmost boxplot in each sub-figure shows the development accuracy when training on the full training set.

To test if having a small training dataset inherently leads to fine-tuning instability we perform the following experiment:[3] we randomly sample 1,000 training samples from the CoLA, MRPC, and QNLI training datasets and fine-tune BERT using 25 different random seeds on each dataset. We compare two different settings: first, training for 3 epochs on the reduced training dataset, and second, training for the same number of *iterations* as on the full training dataset.

We show the results in Figure 5.3 and observe that training on less data does indeed affect the fine-tuning variance, in particular, there are many more failed

3 We remark that a similar experiment was done in Phang et al., 2018, but with a different goal of showing that their extended pre-training procedure can improve fine-tuning stability.

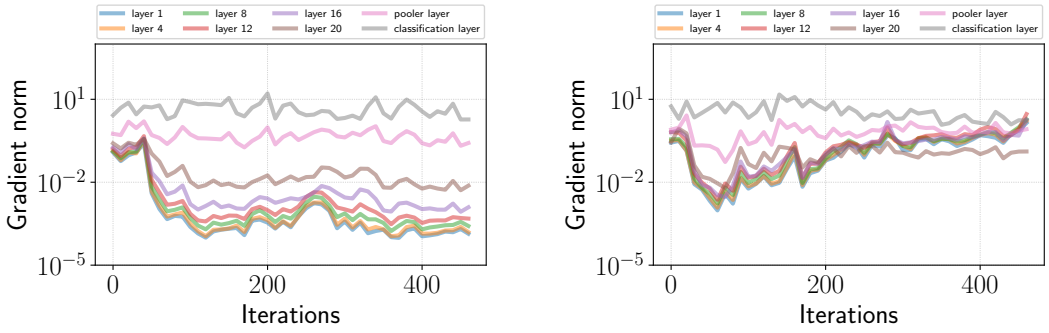
runs. However, when we simply train for as many *iterations* as on the full training dataset, we almost completely recover the original variance of the fine-tuning performance. We also observe no failed runs at all on MRPC and QNLI and only a single failed run on CoLA which is similar to the results obtained by training on the full training set. Further, as expected, we observe that training on fewer samples affects the generalization of the model, leading to a worse development set performance on all three tasks.[4]

We conclude from this experiment, that the role of training dataset size per se is *orthogonal* to fine-tuning stability. *What is crucial is rather the number of training iterations.* As our experiment shows, the observed increase in instability when training with smaller datasets can rather be attributed to the reduction of the number of iterations (that changes the effective learning rate schedule) which, as we will show in the next section, has a crucial influence on the fine-tuning stability.

5.6 Disentangling optimization and generalization

Our findings in §5.5 detail that while both catastrophic forgetting and small size of the datasets indeed *correlate* with fine-tuning instability, none of them are causing it. In this section, we argue that the fine-tuning instability is an optimization problem, and it admits a simple solution. Additionally, we show that even though a large fraction of the fine-tuning instability can be explained by optimization, the remaining instability can be attributed to generalization issues where fine-tuning runs with the same training loss exhibit noticeable differences in the development set performance.

4 Appendix C.8 shows that the same holds true for datasets from different domains than the pre-training data.



(a) Failed run

(b) Successful run

Figure 5.4: Gradient norms (plotted on a *logarithmic scale*) of different layers on RTE for a failed and successful run of BERT fine-tuning. We observe that the failed run is characterized by *vanishing gradients* in the bottom layers of the network. Additional plots for other weight matrices can be found in the Appendix.

5.6.1 The role of optimization

Failed fine-tuning runs suffer from vanishing gradients We observed in Figure 5.2c that the failed runs have practically constant training loss throughout training (see Figure C.6 in Appendix C for a comparison with successful fine-tuning). In order to better understand this phenomenon, in Figure 5.4 we plot the ℓ_2 gradient norms of the loss function with respect to different layers of BERT, for one failed and one successful fine-tuning run. For the failed run we see large enough gradients only for the top layers and *vanishing gradients* for the bottom layers. This is in large contrast to the successful run. While we also observe small gradients at the beginning of training (until iteration 70), gradients start to grow as training continues. Moreover, at the end of fine-tuning, we observe gradient norms nearly 2× orders of magnitude larger than that of the failed run. Similar visualizations for additional layers and weights can be found in Figure C.2 in Appendix C. Moreover, we observe the same behavior also for RoBERTa and

ALBERT models, and the corresponding figures can be found in Appendix C as well (Figure C.3 and C.4).

Importantly, we note that the vanishing gradients we observe during fine-tuning are harder to resolve than the standard *vanishing gradient problem* (Hochreiter, 1991; Bengio et al., 1994). In particular, common weight initialization schemes (Glorot and Bengio, 2010; He et al., 2015) ensure that the pre-activations of each layer of the network have zero mean and unit variance in expectation. However, we cannot simply modify the weights of a pre-trained model on each layer to ensure this property since this would conflict with the idea of using the pre-trained weights in the first place.

Importance of bias correction in ADAM Following Devlin et al., 2019, subsequent works on fine-tuning BERT-based models use the ADAM optimizer (D. P. Kingma and Ba, 2015, see also §2.2.3.1). A subtle detail of the fine-tuning scheme of Devlin et al., 2019 is that it *does not* include the bias correction in ADAM.

D. P. Kingma and Ba (2015) already describe the effect of the bias correction as to reduce the learning rate at the beginning of training. By rewriting the update equations of the ADAM in Algorithm 2 as follows, we can clearly see this effect of bias correction.

$$\eta_t \leftarrow \eta \cdot \sqrt{1-\beta_2^t}/(1-\beta_1^t), \tag{5.1}$$

$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \cdot \boldsymbol{m}_t/(\sqrt{\boldsymbol{v}_t}+\epsilon), \tag{5.2}$$

Equation (5.1) shows that bias correction simply boils down to reducing the original step size η by a multiplicative factor $\sqrt{1-\beta_2^t}/(1-\beta_1^t)$ which is significantly below 1 for the first iterations of training and approaches 1 as the number of training iterations t increases (see Figure 5.5). Along the same lines, You et al., 2020 explicitly remark that bias correction in ADAM has a similar effect to the warmup which is widely used in deep learning to prevent divergence

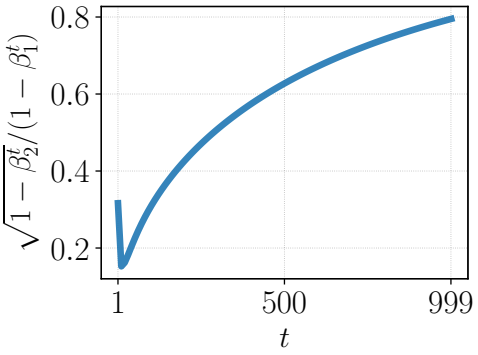


Figure 5.5: The bias correction term of ADAM as a function of the training steps t.

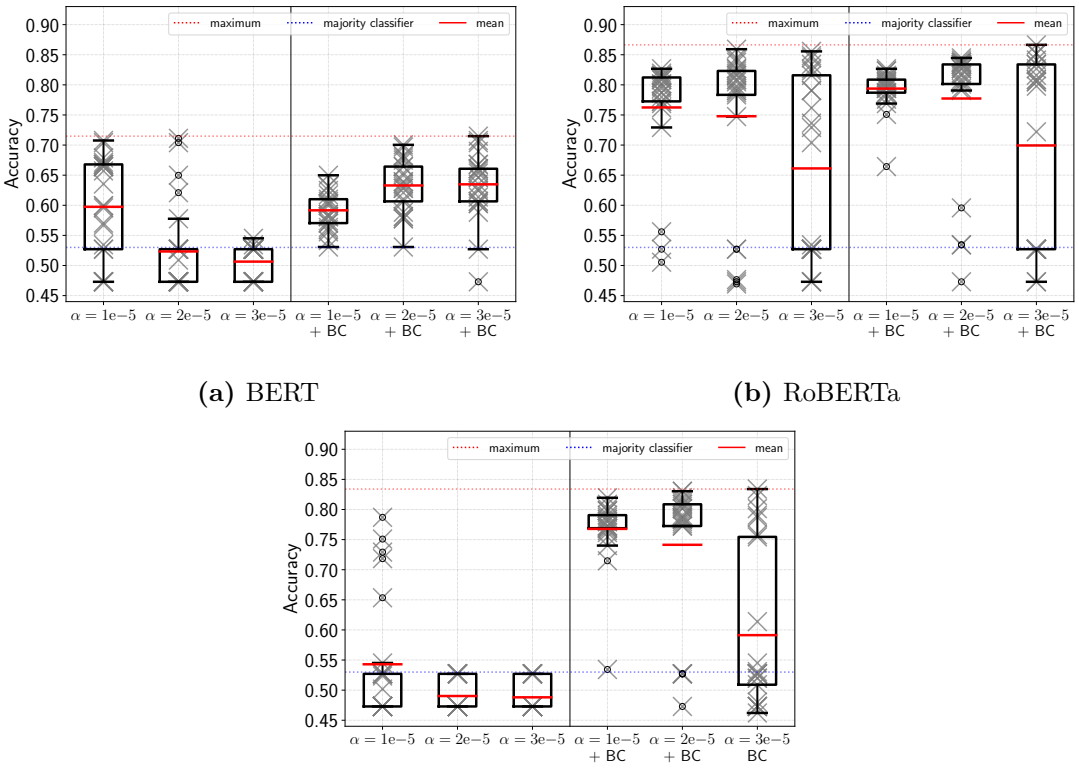
early in training (He et al., 2016; Goyal et al., 2017; Devlin et al., 2019; Wong et al., 2020).

The implicit warmup of ADAM is likely to be an important factor that contributed to its success. We argue that fine-tuning BERT-based language models is not an exception. In Figure 5.6 we show the results of fine-tuning on RTE with and without bias correction for BERT, RoBERTa, and ALBERT models.[5] We observe that there is a significant benefit in combining warmup with bias correction, particularly for BERT and ALBERT. Even though for RoBERTa fine-tuning is already more stable even without bias correction, adding bias correction gives an additional improvement.

Our results show that bias correction is useful if we want to get the best performance within 3 epochs, the default recommendation by Devlin et al. (2019). An alternative solution is to simply train longer with a smaller learning rate, which also leads to much more stable fine-tuning.

We provide a more detailed ablation study in Appendix C (Figure C.1) with analogous box plots for BERT using various learning rates, numbers of training epochs, with and without bias correction.

5 Some of the hyperparameter settings lead to a small fine-tuning variance where all runs lead to a performance below the majority baseline. Obviously, such fine-tuning stability is of limited use.



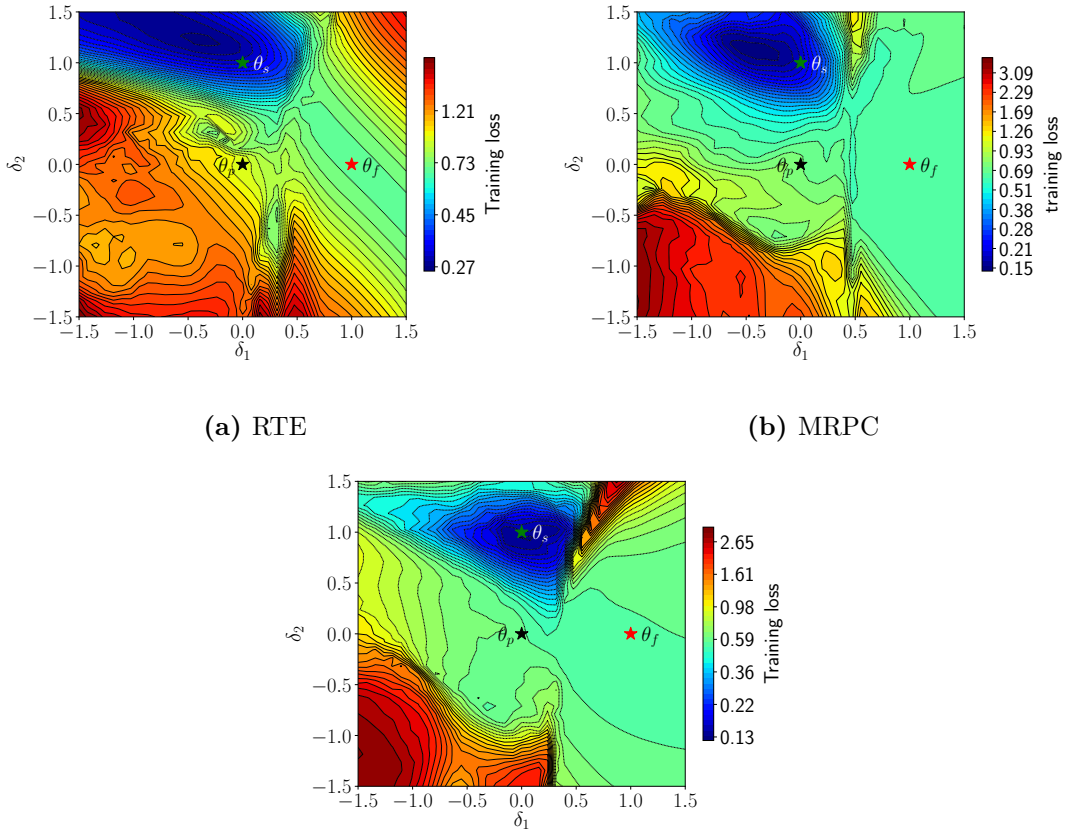
(a) BERT **(b)** RoBERTa

(c) ALBERT

Figure 5.6: Box plots showing the fine-tuning performance of (a) BERT, (b) RoBERTa, (c) ALBERT for different learning rates α with and without bias correction (BC) on RTE. For BERT and ALBERT, having bias correction leads to more stable results and allows to train using larger learning rates. For RoBERTa, the effect is less pronounced but still visible.

Finally, concurrently to our work, T. Zhang et al., 2021 also make a similar observation about the importance of bias correction and longer training which gives further evidence to our findings.

Loss surfaces To get further intuition about the fine-tuning failure, we provide loss surface visualizations (Li et al., 2018; Hao et al., 2019) of failed and successful runs when fine-tuning BERT. Denote by $\boldsymbol{\theta}_p$, $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_s$ the parameters of the pre-trained model, failed model, and successfully trained model, respectively. We plot a two-dimensional loss surface $f(\alpha, \beta) = \mathcal{L}(\boldsymbol{\theta}_p + \alpha\delta_1 + \beta\delta_2)$ in the subspace spanned



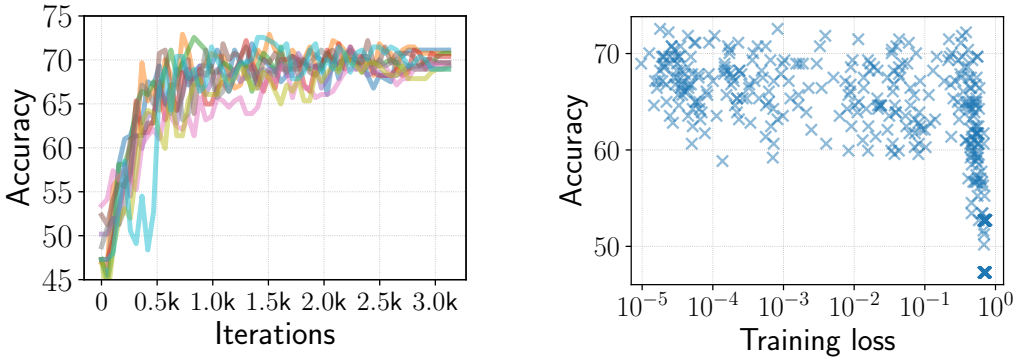
(a) RTE **(b)** MRPC

(c) CoLA

Figure 5.7: 2D loss surfaces in the subspace spanned by $\delta_1 = \boldsymbol{\theta}_f - \boldsymbol{\theta}_p$ and $\delta_2 = \boldsymbol{\theta}_s - \boldsymbol{\theta}_p$ on RTE, MRPC, and CoLA. $\boldsymbol{\theta}_p$, $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_s$ denote the parameters of the pre-trained, failed, and successfully trained model, respectively.

by $\delta_1 = \boldsymbol{\theta}_f - \boldsymbol{\theta}_p$ and $\delta_2 = \boldsymbol{\theta}_s - \boldsymbol{\theta}_p$ centered at the weights of the pre-trained model $\boldsymbol{\theta}_p$. Additional details are specified in Appendix C.6 in Appendix C.

Contour plots of the loss surfaces for RTE, MRPC, and CoLA are shown in Figure 5.7. They provide additional evidence to our findings on vanishing gradients: for failed fine-tuning runs gradient descent converges to a "bad" valley with a sub-optimal training loss. Moreover, this bad valley is separated from the local minimum (to which the successfully trained run converged) by a barrier (see also Figure C.5 in Appendix C). Interestingly, we observe a highly similar geometry for all three datasets providing further support for our interpretation of fine-tuning instability as a primarily optimization issue.



(a) Development set accuracy over training

(b) Generalization vs. training loss

Figure 5.8: Development set accuracy for multiple fine-tuning runs on RTE. The models for (a) are trained with 10 different seeds, and models for (b) are taken at the end of the training, and trained with different seeds and hyperparameters.

5.6.2 The role of generalization

We now turn to the generalization aspects of fine-tuning instability. In order to show that the remaining fine-tuning variance on the development set can be attributed to generalization, we perform the following experiment: we fine-tune BERT on RTE for 20 epochs and show the development set accuracy for 10 successful runs in Figure 5.8a. Further, we show in Figure 5.8b the development set accuracy vs. training loss of all BERT models fine-tuned on RTE for the full ablation study (shown in Figure C.1 in Appendix C)), in total 450 models.

We find that despite achieving close to zero training loss overfitting is not an issue during fine-tuning. This is consistent with previous work (Hao et al., 2019), which arrived at a similar conclusion. Based on our results, we argue that it is even desirable to train for a larger number of iterations since the development accuracy varies considerably during fine-tuning and it does not degrade even when the training loss is as low as 10^{-5}.

Combining these findings with results from the previous section, we conclude that the fine-tuning instability can be decomposed into two aspects: optimization and generalization. In the next section, we propose a simple solution addressing both issues.

5.7 A simple but hard-to-beat baseline for fine-tuning BERT

As our findings in §5.6 show, the empirically observed instability of fine-tuning BERT can be attributed to vanishing gradients early in training as well as differences in generalization late in training. Given the new understanding of fine-tuning instability, we propose the following guidelines for fine-tuning transformer-based masked language models on small datasets:

- Use small learning rates with bias correction to avoid vanishing gradients early in training.
- Increase the number of iterations considerably and train to (almost) zero training loss.

This leads to the following simple baseline scheme: we fine-tune BERT using ADAM with bias correction and a learning rate of 2e−5. The training is performed for 20 epochs, and the learning rate is linearly increased for the first 10% of steps and linearly decayed to zero afterward. All other hyperparameters are kept unchanged. A full ablation study on RTE testing various combinations of the changed hyperparameters is presented in Appendix C.4 in Appendix C).

Results Despite the simplicity of our proposed fine-tuning strategy, we obtain strong empirical performance. Figure 5.1 and Table 5.1 show the results of fine-tuning BERT on RTE, MRPC, and CoLA. We compare to the default strategy of Devlin et al. (2019) and the recent Mixout method proposed by Lee et al. (2020).

Approach	RTE			MRPC			CoLA		
	std	mean	max	std	mean	max	std	mean	max
Devlin et al. (2019)	4.5	50.9	67.5	3.9	84.0	91.2	25.6	45.6	64.6
Lee et al. (2020)	7.9	65.3	**74.4**	3.8	87.8	**91.8**	20.9	51.9	64.0
Ours	**2.7***	**67.3**	71.1	**0.8***	**90.3**	**91.7**	**1.8***	**62.1**	**65.3**

Table 5.1: Standard deviation, mean, and maximum performance on the development set of RTE, MRPC, and CoLA when fine-tuning BERT over 25 random seeds. Standard deviation: lower is better, i.e. fine-tuning is more stable. * denotes significant difference ($p < 0.001$) when compared to the second smallest standard deviation.

First, we observe that our method leads to a much more stable fine-tuning performance on all three datasets as evidenced by the significantly smaller standard deviation of the final performance. To further validate our claim about the fine-tuning stability, we run Levene's test (Levene, 1960) to check the equality of variances for the distributions of the final performances on each dataset. For all three datasets, the test results in a p-value less than 0.001 when we compare the variances between our method and the method achieving the second smallest variance. Second, we also observe that our method improves the overall fine-tuning performance: in Table 5.1 we achieve a higher mean value on all datasets and also comparable or better maximum performance on MRPC and CoLA respectively.

We also note that we suggest to increase the number of fine-tuning iterations only for small datasets, and thus the increased computational cost of our proposed scheme is not a problem in practice. In fact, we argue that overall our findings lead to *more efficient* fine-tuning because of the significantly improved stability which effectively reduces the number of necessary fine-tuning runs.

5.8 Conclusions

In this chapter, we have discussed the existing hypotheses regarding the reasons behind fine-tuning instability and proposed a new baseline strategy for fine-tuning that leads to significantly improved fine-tuning stability and overall improved results on commonly used datasets from the GLUE benchmark.

By analyzing failed fine-tuning runs, we find that neither catastrophic forgetting nor small dataset sizes sufficiently explain fine-tuning instability. Instead, our analysis reveals that fine-tuning instability can be characterized by two distinct problems: (1) optimization difficulties early in training, characterized by vanishing gradients, and (2) differences in generalization, characterized by a large variance of development set accuracy for runs with almost equivalent training performance.

Based on our analysis, we propose a simple but strong baseline strategy for fine-tuning BERT which outperforms the previous works in terms of fine-tuning stability while maintaining or even increasing overall performance.

6

Investigating Generalization of Task-adapted Models

Contents

Few-shot fine-tuning and in-context learning are two alternative strategies for task adaptation of pre-trained language models. Recently, in-context learning has gained popularity over fine-tuning due to its simplicity and improved out-of-domain generalization, and because extensive evidence shows that fine-tuned

models pick up on spurious correlations. Unfortunately, previous comparisons of the two approaches were done using models of different sizes. This raises the question of whether the observed weaker out-of-domain generalization of fine-tuned models is an inherent property of fine-tuning or a limitation of the experimental setup. In this chapter, we compare the generalization of few-shot fine-tuning and in-context learning to challenge datasets, while controlling for the models used, the number of examples, and the number of parameters, ranging from 125M to 30B. Our results show that fine-tuned language models *can* in fact generalize well out-of-domain. We find that both approaches generalize similarly; they exhibit large variation and depend on properties such as model size and the number of examples, highlighting that robust task adaptation remains a challenge.

The content presented in this chapter is based on:

> **Mosbach, Marius**, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar (July 2023). "Few-shot Fine-tuning vs. In-context Learning: A Fair Comparison and Evaluation." In: *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto, Canada: Association for Computational Linguistics, pp. 12284–12314. URL: https://aclanthology.org/2023.findings-acl.779.

As a first author, Marius Mosbach conceptualized the research, conducted the experiments and led the paper writing. Tiago Pimentel helped to shape the research questions, provided feedback and helped with the writing. Shauli Ravfogel provided feedback during earlier stages of the project and helped with conceptualizing the research and proofreading. Dietrich Klakow provided feedback. Yanai Elazar advised the project, ran some of the experiments, helped with writing and was involved in discussing ideas and shaping the research questions.

6.1 Introduction

Adapting a pre-trained language model to a target task is of high practical importance to the natural language processing (NLP) community (as seen in Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019; Brown et al., 2020, *inter alia*). Among the commonly used *task adaptation* strategies, two stand out: fine-tuning (FT) and in-context learning (ICL). In short, fine-tuning a model involves a supervised learning setup on a target dataset; while in-context learning involves prompting a model with a series of input–label pairs, without updating the model's parameters.

Both approaches come with pros and cons: in-context learning reuses a single pre-trained model for various downstream tasks, allows specifying the desired behavior via natural language, and has recently shown impressive results on challenging reasoning tasks (Brown et al., 2020; Wei et al., 2022a; Press et al., 2022b). However, the model's context size limits the number of demonstrations that can be used. For instance, using 32 randomly selected examples from the RTE dataset (Dagan et al., 2006) already exceeds the context size of OPT models (S. Zhang et al., 2022).[1] In addition, in-context learning is highly sensitive to the format and order of its inputs (Lu et al., 2022; Min et al., 2022).

Fine-tuning, on the other hand, typically results in a single specialized model per task,[2] and can be applied to training sets of arbitrary size. However, such models are sensitive to initialization (Dodge et al., 2020b) and can suffer from instability during training (see Chapter 5).

For text classification tasks, where both strategies often lead to similar performance on in-domain data (when using the same amount of data), recent works have argued that in-context learning leads to better out-of-domain (OOD)

1 While GPT-3 and OPT both have a context size of 2048 tokens, more recent models such as GPT-4 (OpenAI, 2023) – which has been developed concurrently to this work – support larger contexts of up to 8192 tokens.

2 Parameter-efficient fine-tuning methods (e.g. Ben Zaken et al. (2022) and E. J. Hu et al. (2022)) address this issue and allow to re-use most of the pre-trained weights across tasks.

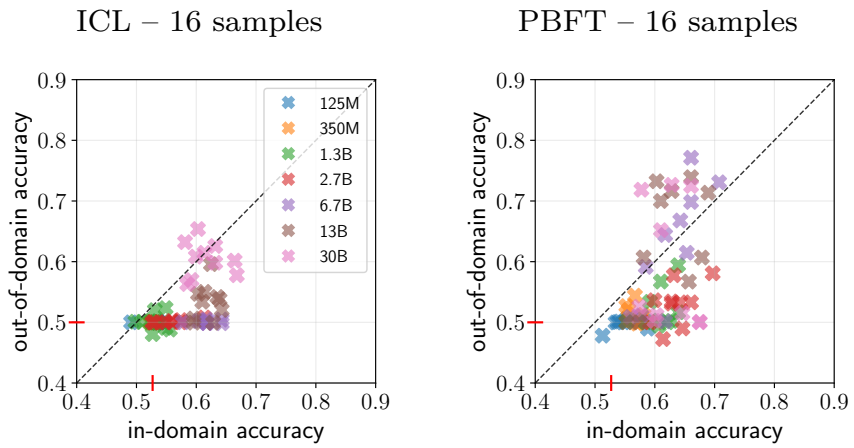


Figure 6.1: In-domain (RTE) and out-of-domain performance (HANS) for in-context learning (ICL) and fine-tuning (FT) with OPT models of various sizes. We fine-tune models using pattern-based fine-tuning (PBFT). We report results using 10 different data seeds. When using 16 samples, in-context learning's performance with a 30B model is comparable to that of fine-tuning with smaller models (6.7B) and for most model sizes, fine-tuning outperforms in-context learning (see Table 6.1a for significance tests). — in the x- and y-axes indicates majority class accuracy.

generalization (Si et al., 2023; Awadalla et al., 2022). However, these comparisons of generalization abilities were not conducted under equal conditions. Most studies compare the in-context learning abilities of large models (e.g. GPT-3, 175B; Brown et al., 2020) to the fine-tuning abilities of much smaller models (e.g. RoBERTa-large, 350M; Y. Liu et al., 2019a). These comparisons raise the question of whether fine-tuning indeed leads to weaker OOD generalization than in-context learning, or whether this is just a byproduct of the experimental setup. In Figure 6.1, we show this is indeed the case: when given only 16 examples, fine-tuning a 6.7B parameters model already achieves similar results to in-context learning with a 30B model, and fine-tuning performance keeps improving with larger models.[3] Moreover, we show in Section 6.4.1 that fine-tuning performance improves even further when training on more data.

3 Table 6.1a presents significance tests for these results.

In this chapter, we compare in-context learning and fine-tuning on an **equal footing** (§6.3). We compare both strategies using the same model (OPT; S. Zhang et al., 2022), the same number of parameters (from 125M to 30B), and the same number of examples. Our results and analyses (§6.4) show that both approaches often achieve comparable results. Both methods are unstable and can perform badly on in-domain and OOD data due to training instability, or prompt choice. We also find that both approaches improve as we increase model size, and that, for the models and datasets we consider, fine-tuning often generalizes even better than in-context learning. Notably, this is in contrast to prior work (§6.7), highlighting the need for fair comparisons of task adaptation strategies.

Based on our findings, we discuss the strengths and limitations of fine-tuning and in-context learning (§6.6), which can inform when to use and how to get the most out of each method.

6.2 Background

For our experiments, we consider pattern-based fine-tuning (PBFT) and in-context learning (ICL), which we introduce in detail in §2.4.1. In the following, we provide additional details relevant for the scope of this chapter.

6.2.1 Fine-tuning

Pattern-based fine-tuning is a recently proposed fine-tuning approach that uses the pre-trained language modeling head[4] instead of a randomly initialized classifier (as used in standard fine-tuning; Howard and Ruder 2018; Devlin et al. 2019), to obtain predictions (Schick and Schütze, 2021; T. Gao et al., 2021, *inter alia*). Compared to vanilla fine-tuning, we have to specify an *input pattern* (to cast the

4 In the case of encoder-only masked language models, such as BERT, this is usually an MLP layer. In the case of decoder-only models, such as OPT, this is a linear projection.

task as a language modeling problem) and define a *verbalizer* (which maps tokens in the pre-trained model's vocabulary to labels; Schick et al., 2020). For example, a NLI pattern might look as follows: `{premise} Question: {hypothesis} Yes or No?`, and the verbalizer will use `Yes` and `No` as tokens. Given these inputs and targets, model parameters are fine-tuned as usual. This method has been shown to be efficient for few-shot learning despite having no advantage over vanilla fine-tuning when the number of examples is large (Tam et al., 2021; Logan IV et al., 2022).

6.2.2 In-context learning

In-context learning is a task adaptation strategy that does not update the weights of the pre-trained model (Brown et al., 2020); instead, in-context learning adapts a model to a task by conditioning it on a sequence of *demonstrations*. A demonstration typically refers to an input x accompanied by its ground-truth label y, both of which have been converted to a specific format using a *pattern* and a *verbalizer* (similar to PBFT). In-context learning thus feeds the model a sequence of such demonstrations, followed by the test input (modified by applying the pattern transformation). The language model is then expected to predict the label of this final data point.[5] Recent work has argued that in-context learning leads to better out-of-domain performance, when compared to fine-tuning (Si et al., 2023; Awadalla et al., 2022). We show that this often does not hold.

5 The evaluation only considers the probabilities assigned to the verbalizer tokens, ignoring any probability mass assigned to other tokens. See §6.3 for details.

6.3 A fair comparison of fine-tuning and in-context learning

We perform a fair comparison of task adaptation via fine-tuning and in-context learning, focusing on in-domain and out-of-domain generalization generalization. We compare them in the few-shot setting using the same models. In the following paragraphs, we provide details about our setup.

In-domain generalization Following the definition of in domain generalization established in §2.2.1.3, we measure in-domain generalization by measuring accuracy on the validation set of each dataset. This is a common practice in analysis works, and used in previous work (Utama et al., 2021; Bandel et al., 2022).

Out-of-domain generalization We consider out-of-domain generalization (OOD) generalization under *covariate shift* (Hupkes et al., 2022). Specifically, we focus on generalization to *challenge datasets*, designed to test whether models adopt a particular heuristic, or make predictions based on spurious correlations during inference (T. McCoy et al., 2019; Elazar et al., 2021).

Models We run all our experiments using 7 different OPT models (S. Zhang et al., 2022) ranging from 125 million to 30 billion parameters, all of which have been trained on the same data. This allows us to study the effect of model size on performance without the confound of using different training data.[6]

Tasks and datasets We focus on two classification tasks in English: natural language inference (NLI) and paraphrase identification. For NLI, we use the Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2018) and Recognizing

6 OPT 30B is the largest model we were able to fit given our resources.

Textual Entailment (RTE) datasets (Dagan et al., 2006) for task adaptation and in-domain generalization, and evaluate OOD generalization on the lexical overlap subset of the Heuristic Analysis for NLI Systems (HANS) (T. McCoy et al., 2019) dataset.[7] We binarize MNLI by removing the neutral examples[8] which allows us to better compare MNLI with RTE (which only has two labels). For paraphrase identification, we train on Quora Question Pairs (QQP) (Sharma et al., 2019) and evaluate OOD generalization on PAWS-QQP (Y. Zhang et al., 2019). Given the large size of the QQP validation set (more than 300k examples), we randomly select 1000 validation examples.

Few-shot setup We follow the same procedure for both approaches. We randomly sample $n \in \{2, 16, 32, 64, 128\}$ examples from the in-domain training set of a given dataset (unless stated otherwise).[9] Due to the high sensitivity of both approaches to the used pattern, as well as to the ordering of the demonstrations in in-context learning (Webson and Pavlick, 2022; Lu et al., 2022), we sample 10 different sets of examples for each n. We also experiment with 3 different patterns, resulting in 30 runs per n and adaption method.[10] Table D.2 in Appendix D.1.3 provides an overview of the patterns and verbalizers for each task.

Fine-tuning setup We perform few-shot pattern-based fine-tuning using a minimal pattern (Logan IV et al., 2022), which simply adds a question mark at the end of every example. For the NLI verbalizer, we use `Yes` and `No`, which we map to the task's labels `entailment` and `not-entailment` respectively. For QQP,

7 Due to similar trends on different HANS subsets in preliminary experiments, we focus on the lexical overlap subset.

8 We compare this to merging the neutral and contradiction classes in Appendix D.2.3, and obtain very similar results.

9 We sample an equal number of examples per label.

10 Except for QQP, where we experiment with only 2 patterns, as one of the patterns is not applicable.

we also use `Yes` and `No` and map them to `not-duplicate` and `duplicate`.[11] We follow the recommendations of given in Chapter 5 and fine-tune all models for 40 epochs using a learning rate of 10^{-5} which increases linearly (warmup) for the first 10% of the training steps and is kept constant afterward. Details of all hyper-parameters are provided in Appendix D.1.5.

In-context learning setup Given OPT's fixed context size of 2048 tokens we are limited in the number of examples used for demonstration. Our main experiments focus on 16 demonstrations, but we also present additional experiments using 2 and 32 demonstrations in Appendix D.2.[12]. We consider a prediction to be correct if the probability assigned to the verbalizer token of the ground-truth label is larger than the probability of the other verbalizer token. We use the same verbalizer tokens as for fine-tuning.

6.4 Results

We present the results for in-domain and OOD model performance in Figure 6.2, comparing both in-context learning and fine-tuning. We perform task adaptation using 16 examples for both strategies. For in-context learning, we provide additional results that demonstrate the importance of choosing the right pattern and number of demonstrations in Appendix D.2.2. For fine-tuning, we provide more details, ablations and discussion of various choices later in this section.

In-domain performance For MNLI and RTE, both in-context learning and fine-tuning exhibit in-domain performance above the majority baseline for most model sizes. Focusing on in-context learning, MNLI and RTE in-domain performance improves as model size increases. On MNLI the largest model (30B)

11 Preliminary experiments showed that `Yes` and `No` is a strong verbalizer for binary classification tasks. This is consistent with previous findings (Webson and Pavlick, 2022).

12 With the exception of RTE, where 32 examples do not fit OPT's context size

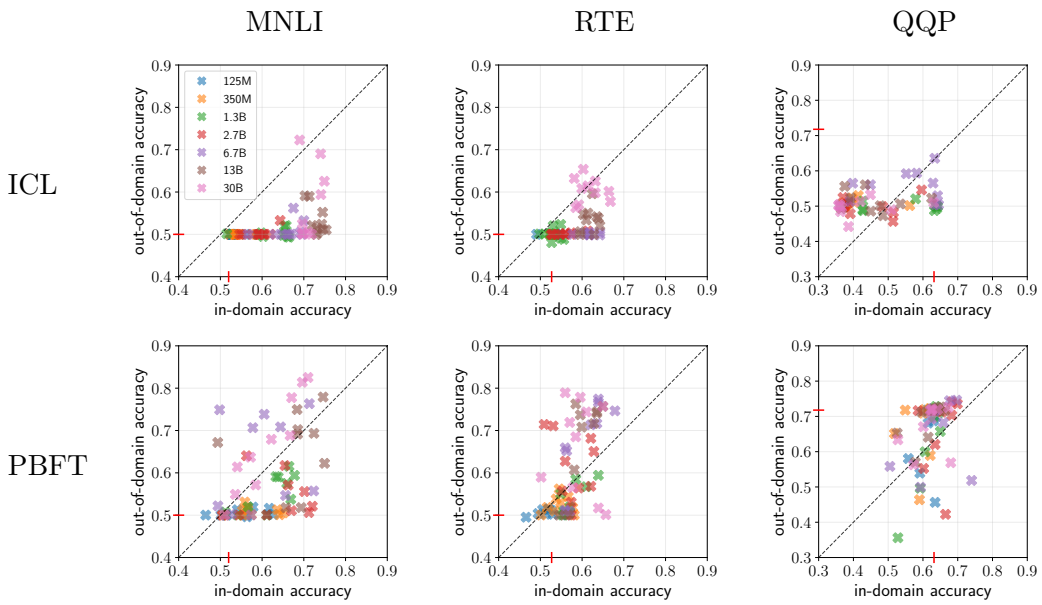


Figure 6.2: ICL and PBFT results for OPT models of various sizes. For each approach, we use 16 examples and perform model selection according to OOD performance. We plot 10 runs per model size which differ only in the data seed. — in the x- and y-axis indicates majority class accuracy.

obtains an average performance of 71.4% and a maximum performance of 74.9%. On RTE, in-context learning with the same model achieves an average and maximum performance of 61.7% and 66.8% respectively. On QQP, the trend of improved performance with increasing model size is less clear and most models perform worse than the majority baseline. Table D.3 (in Appendix D.1.4) compares our in-context learning results with previous work.

For fine-tuning, we similarly observe that in-domain performance increases with model size. Moreover, across all datasets and model sizes, fine-tuning with just 16 examples leads to similar in-domain performance as in-context learning (see Tables D.5 and D.6 in Appendix D.2.1 for statistical tests comparing in-domain performance of fine-tuning and in-context learning on RTE and MNLI). On QQP, we again observe no clear relationship between model size and performance. Only 10 out of 70 models perform better than the majority baseline.

Out-of-domain performance Turning to OOD performance, we find that for MNLI and QQP most of the in-context learning models perform close to the majority baseline. On MNLI, only the largest model (30B) shows good OOD generalization for 4 out of 10 runs. On RTE, in-domain and OOD performance of the 30B model mostly overlap, which is consistent with the findings of Si et al. (2023). In particular, when comparing the relationship between the in-domain and OOD performance of the 30B model to the smallest fine-tuned models (125M and 350M) one might conclude that in-context learning leads to better OOD performance; for fine-tuning on MNLI and RTE, indeed, the smallest models have poor OOD performance.

However, as model size increases, OOD performance increases as well, demonstrating that even in the challenging few-shot setting, fine-tuned models can generalize OOD. Focusing on the largest models (6.7B, 13B, and 30B) fine-tuned on MNLI, we find that for most runs, OOD performance is on par or even better than in-domain performance. On RTE, the trend is even stronger. Even with the 1.3B model, we observe good in-domain and OOD performance, and both improve as the models get larger. Notably, for many models, OOD performance is even better than in-domain performance.

In summary, *our comparison shows that fine-tuned language models can generalize OOD as well or even better than models adapted via in-context learning* (see statistical tests comparing them in Table 6.1). This highlights the importance of comparing adaptation approaches using models of the same size.

6.4.1 A closer look at fine-tuning generalization

Having established that few-shot fine-tuning can also lead to strong in-domain and OOD performance, we now focus on better understanding the individual choices that impact the in-domain and out-of-domain performance of fine-tuning. Given that on QQP, most models achieve close to majority accuracy, we focus on MNLI and RTE in the following and present results for QQP in Appendix D.2.

		PBFT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	−0.00	0.01	0.02	0.03	0.12	0.14	0.09
	350M	−0.00	0.01	0.02	0.03	0.12	0.14	0.09
	1.3B	−0.00	0.01	0.02	0.03	0.12	0.14	0.09
	2.7B	−0.00	0.01	0.02	0.03	0.12	0.14	0.09
	6.7B	−0.00	0.01	0.02	0.03	0.12	0.14	0.09
	13B	−0.04	−0.02	−0.01	−0.00	0.09	0.11	0.05
	30B	−0.11	−0.09	−0.08	−0.08	0.02	0.03	−0.02

(a) RTE

		PBFT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	−0.00	0.00	0.02	0.01	0.10	0.11	0.07
	350M	−0.00	0.00	0.02	0.01	0.10	0.11	0.07
	1.3B	−0.01	−0.00	0.01	0.01	0.10	0.11	0.07
	2.7B	−0.01	−0.00	0.01	0.01	0.09	0.10	0.07
	6.7B	−0.01	−0.01	0.01	0.00	0.09	0.10	0.06
	13B	−0.03	−0.03	−0.02	−0.02	0.07	0.08	0.04
	30B	−0.07	−0.07	−0.05	−0.06	0.03	0.04	0.00

(b) MNLI

Table 6.1: Difference between average **out-of-domain performance** of in-context learning and fine-tuning on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For in-context learning, we use the `gpt-3` pattern. For fine-tuning, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: in-context learning performs significantly better than fine-tuning, fine-tuning performs significantly better than in-context learning. For cells without color, there is no significant difference.

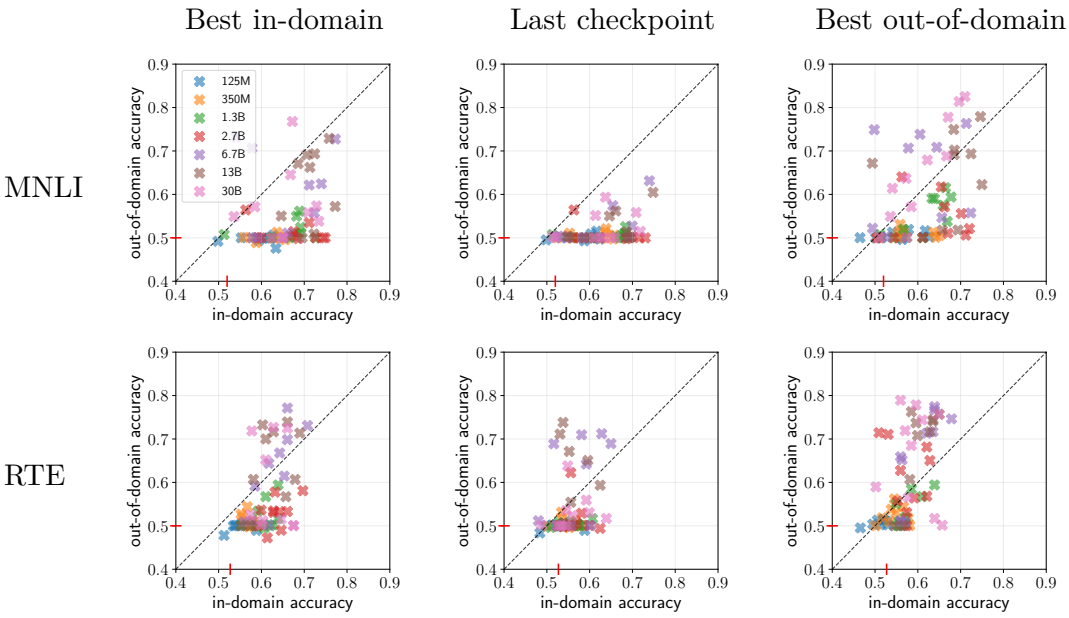


Figure 6.3: Comparing model selection strategies for fine-tuning. The first and second rows show results for MNLI and RTE respectively. We train on 16 examples and plot results for 10 runs for each model size. — in the x- and y-axes indicates majority class accuracy.

The role of model selection Our fine-tuning results in Figure 6.2 show that many fine-tuned models lead to good out-of-domain generalization. But what is the role of model selection in identifying these checkpoints? To answer this question, we compare selecting the model (a) with the best in-domain performance, (b) at the end of fine-tuning, and (c) with the best out-of-domain performance. Figure 6.3 shows the results when fine-tuning on 16 examples. Results for additional sample sizes are shown in Figures D.4 to D.6 in Appendix D.2.3.

Our results show that when performing model selection according to in-domain performance, only the largest models achieve good OOD performance. On the other hand, when performing model selection according to OOD performance, smaller models can also generalize well (e.g. for the 2.7B model on RTE, 7 out of 10 models have equal or even better OOD than in-domain performance), and this trend persists as model size increases. Interestingly, on RTE, we also observe

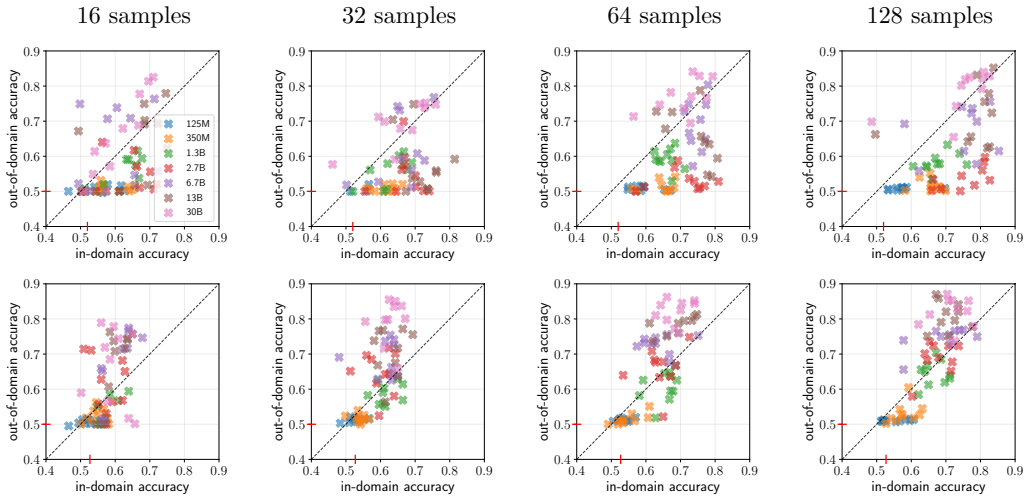


Figure 6.4: Exploring the effect of increasing training examples on fine-tuning. The first and second rows show results for MNLI and RTE respectively. We plot results for 10 runs for each model size and perform model selection according to out-of-domain performance. — in the x- and y-axes indicates majority class accuracy.

models with a strong OOD performance when selecting the last checkpoint, which typically leads to poor OOD performance on MNLI.

Training on more data In contrast to in-context learning, where the maximum number of demonstrations is limited by the context size of a model, fine-tuning allows us to perform task adaptation using arbitrary amounts of data. Here, we analyze how the relationship between in-domain and OOD performance is impacted by training on more data. Figure 6.4 shows the results for MNLI and RTE, and results for QQP are provided in Figure D.6 in Appendix D.2.3. For the smallest models, we find that while in-domain performance increases with more training data, OOD performance remains low, which is consistent with previous work (Utama et al., 2021). However, for larger models, OOD performance improves as the amount of training data increases and the same trend can be observed when performing model selection according to in-domain performance (see Figures D.4 to D.6 in Appendix D.2.3).

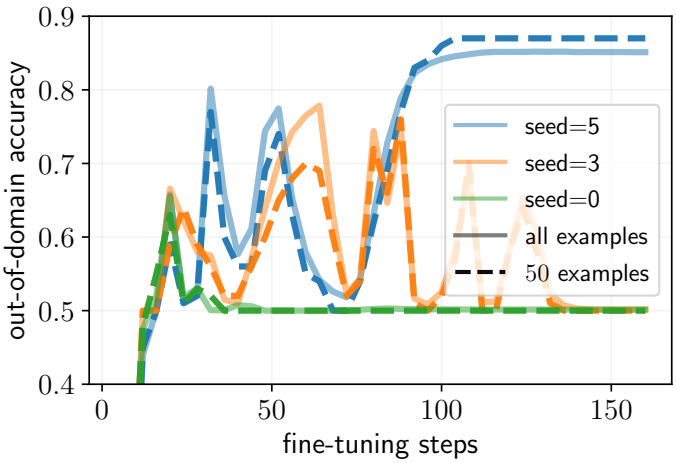


Figure 6.5: Estimating OOD performance using less data. We compare OOD performance estimated using all vs. 50 examples when fine-tuning OPT 13B on RTE. Each color corresponds to a run with a different data seed.

How much OOD data is needed? In the experiments so far, we evaluated the models on the full evaluation set (unless mentioned otherwise). Further, we selected fine-tuning models based on this evaluation; choosing the best model according to its in-domain or OOD performance on this entire set. This setup is not realistic, since in such a scenario where large amounts of data are available for evaluation, it can be used more effectively for training (D. Zhu et al., 2023). Hence, in this experiment, we quantify the ability to estimate a model's performance on OOD data using smaller evaluation sets. We fine-tune OPT 13B on MNLI using 128 examples using three different data seeds and plot the OOD generalization in Figure 6.5. Our results show that using just 50 randomly selected examples is sufficient to distinguish checkpoints that generalize well from those that do not, which would allow us to select, with only these 50 examples, the best OOD checkpoint in a model's training run. This is also reflected in the Pearson correlation of the OOD performance during fine-tuning when evaluating it on all vs. 50 examples, which is very high: 0.99.

Comparing fine-tuning approaches Lastly, we investigate the importance of performing pattern-based fine-tuning instead of vanilla fine-tuning by fine-tuning

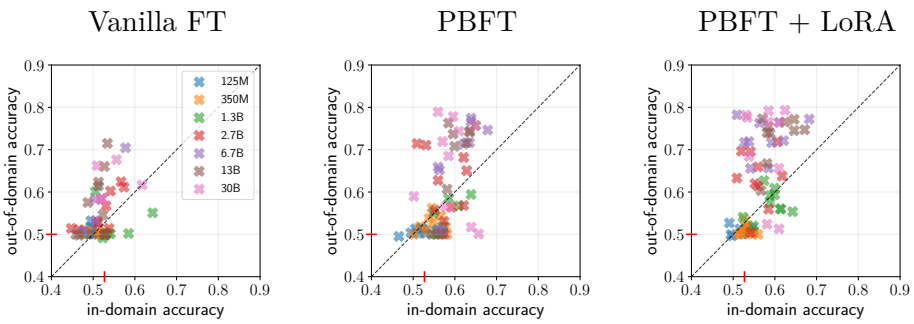


Figure 6.6: Comparing fine-tuning approaches on RTE. We use 16 examples and perform model selection according to out-of-domain performance. — in the x- and y-axes indicates the accuracy of the majority class label.

a model with a randomly initialized classification head (Howard and Ruder, 2018; Devlin et al., 2019). Further, as an extra fine-tuning strategy, we also apply LoRA (E. J. Hu et al., 2022) – a recently proposed approach for parameter-efficient fine-tuning – on top of pattern-based fine-tuning for comparison. This makes adaptation via fine-tuning more similar to adaptation via in-context learning as it allows the re-use of a large fraction of the weights of a pre-trained language model across tasks.[13] We fine-tune all models on 16 examples from RTE and present the results in Figure 6.6. For all fine-tuning approaches, we observe a clear improvement in both in-domain and OOD performance as models become larger. Compared to vanilla fine-tuning, pattern-based fine-tuning leads to better overall performance. When combined with LoRA, pattern-based fine-tuning leads to very similar performance as training all parameters. These results demonstrate the generality of our findings beyond a specific fine-tuning method.

6.4.2 Our findings generalize beyond OPT

Figure 6.7 provides a comparison of in-context learning and fine-tuning using Pythia models[14] of different sizes ranging from 410M to 12B parameters (Biderman

13 We provide more details on both approaches in Appendix D.1.5.
14 We use the non-deduped models.

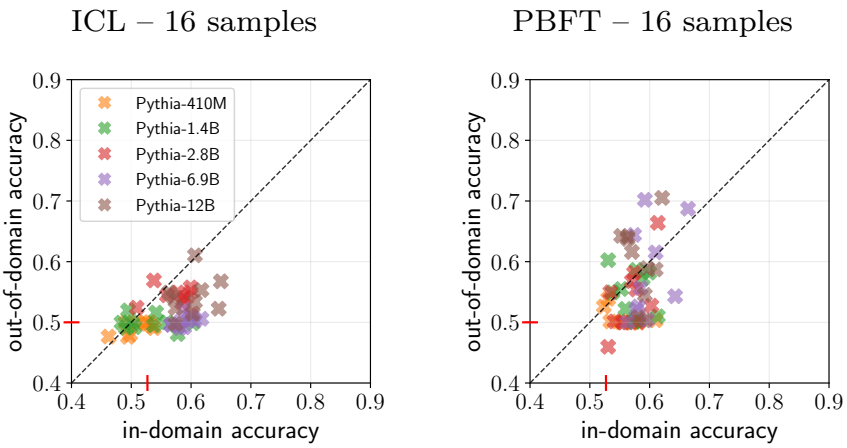


Figure 6.7: ICL and fine-tuning results for Pythia models on RTE. We fine-tune models using PBFT. For each approach, we use 16 examples and perform model selection according to in-domain performance. We plot 10 runs per model size which differ only in the data seed. — in the x- and y-axes indicates majority class accuracy.

et al., 2023). The corresponding significance tests for OOD performance are shown in Table 6.2 (significance tests for in-domain performance are in Appendix D.3). Similar to OPT, all Pythia models have been trained on the same data, and in the same order. We fine-tune using PBFT and select models according to in-domain performance. The results for additional patterns, model selection strategies, and sample sizes are discussed in Appendix D.3.

Similarly to OPT, we observe a clear effect of model size on both in-domain and OOD performance. For most model sizes, fine-tuning leads to significantly better OOD performance than in-context learning and both the in-domain and OOD performance of Pythia models improve drastically as we fine-tune on more data (see Figure D.9). This demonstrates the generality of our findings beyond a specific language model.

		FT				
		410M	**1.4B**	**2.8B**	**6.9B**	**12B**
ICL	**410M**	0.02	0.06	0.05	0.09	0.11
	1.4B	0.01	0.05	0.04	0.08	0.10
	2.8B	−0.03	0.01	−0.00	0.04	0.06
	6.9B	0.01	0.05	0.04	0.08	0.10
	12B	−0.03	0.01	−0.00	0.04	0.06

Table 6.2: Difference between average **out-of-domain performance** of in-context learning and fine-tuning with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For in-context learning, we use the `gpt-3` pattern. For fine-tuning, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: in-context learning performs significantly better than fine-tuning, fine-tuning performs significantly better than in-context learning. For cells without color, there is no significant difference.

6.5 Discussion

Our findings in the previous section demonstrate that fine-tuned language models can generalize OOD too, highlighting the importance of comparing adaptation approaches fairly. In this section, we present further insights from our experiments and provide a high-level comparison of the pros and cons of adaptation via in-context learning and fine-tuning.

What signal to learn from? Both our in-context learning and fine-tuning results exhibit a large variance in both in-domain and OOD performance. Our results show different OOD behavior during fine-tuning when varying only the data seed. In addition, as previous work has shown, the choice of patterns and verbalizers

impact both in-context learning and pattern-based fine-tuning performance in unintuitive ways. For instance, Webson and Pavlick (2022) find that pattern-based fine-tuned models perform well even when using misleading patterns. Here, we find that in-context learning's generalization is heavily dependent on the choice of pattern and verbalizer. This shows the importance of the choice of training data and patterns for task adaptation.

Advances in task adaptation The success of in-context learning led to the development of new methods for improving on it further, such as calibration (Zhao et al., 2021), and chain-of-thought prompting (Wei et al., 2022a). In this work, we focus on the 'vanilla' version of in-context learning and the fine-tuning approach most similar to it – pattern-based fine-tuning. Our results suggest that these two approaches are more similar than previously thought, as they achieve similar performance both in-domain and OOD. As such, new methods for in-context learning can also be applied to pattern-based fine-tuning, and we expect them to achieve similar results.

Analyzing the fine-tuning loss surface Looking at the OOD generalization curves throughout fine-tuning (in Figure 6.5 and additional plots in Appendix D.4), we observe that for some runs, OOD performance fluctuates heavily and models change their generalization 'strategy' during fine-tuning. In Figure 6.5, we can see that some fine-tuning runs undergo a dramatic change in OOD performance after 75 steps. We leave it to future work to further study this behavior and the relationship between the fine-tuning loss surface and OOD generalization (Shwartz-Ziv et al., 2022; Juneja et al., 2023).

Feature	**Fine-tuning**	**In-context learning**
Users	Experts	Experts & Non-experts
Interaction	Pre-defined	Textual
Reusability	Medium	High
Applicability to low-resource languages	High	Limited
Requires training	Yes	No
Inference time	|test example|	|test example| + |demonstrations|
|Demonstrations|	Unlimited	$\leq$100
Variance	High	High
SOTA	Yes	Yes
Size scaling	Standard	Standard
|Demonstrations| scaling	Standard	Limited
Invented	2018	2020
Well understood	No	No

Table 6.3: A high-level comparison between key features of fine-tuning and in-context learning.

6.6 Comparing fine-tuning and in-context learning

This section examines the key features for task adaptation and compares fine-tuning and in-context learning. We summarize our findings in Table 6.3. We begin by discussing features related to user interaction, which can be found in the first part of the table. Fine-tuning requires expertise in model training, whereas in-context learning only requires natural language, i.e., non-experts can use this approach more easily. In-context learning is also highly reusable as it does not modify the pre-trained model and hence, the same model can be used for many tasks; fine-tuning, however, is not as reusable (with the exception of

parameter-efficient methods) and typically results in a specialized model per task. Unfortunately, despite its user-friendliness and reusability, in-context learning does not work out of the box for some tasks which require more sophisticated prompting (Wei et al., 2022a).

In-context learning requires large models to work in contrast to fine-tuning, which works well even with small models (Devlin et al., 2019). This hinders the applicability of in-context learning to models developed for low-resource languages, as training billion parameter-scale models requires huge amounts of training data, which are simply unavailable for many languages. As such, fine-tuning is still the dominating adaptation approach in this setting (Pfeiffer et al., 2022; Alabi et al., 2022, *inter alia*).

Next, we compare technical details regarding the training and inference of such approaches. While fine-tuning requires training (which when dealing with large models can become expensive), in-context learning does not. On the other hand, the inference time of fine-tuned models is much smaller than in-context learning, since it only includes the time that it takes to process the minimal pattern and the test instance. When using in-context learning, each test instance has to include all of the demonstrations as well, which increases the inference time. The fixed context size of the model also limits the number of demonstrations that can be used[15], while fine-tuning allows for unlimited training examples. We show in this work that both methods can achieve strong performance on both in-domain and OOD datasets. Both approaches improve with model size, but fine-tuning benefits more from additional samples than in-context learning does, as was also shown in previous work (Min et al., 2022).

Finally, we highlight that both methods are relatively recent: vanilla fine-tuning was invented in 2018 (Howard and Ruder, 2018) and in-context learning in 2020 (Brown et al., 2020).[16] As such, these methods are still poorly understood,

15 Note that some methods allow an infinite context e.g. Press et al., 2022a; Martins et al., 2022. Most current successful LMs, however, have limited context sizes.

16 PBFT was also invented in 2020 (Schick and Schütze, 2021).

and more research is required on both approaches to better understand their strengths and weaknesses.

6.7 Related work

Brown et al. (2020) compare GPT-3's few-shot in-context learning performance with fine-tuned language models trained in the fully supervised setting, finding that both approaches lead to similar results in question answering. However, the fine-tuned models they compare in-context learning to are smaller models, making the task adaptation comparison unfair. For SuperGLUE, while using smaller models, they find that fine-tuning largely outperforms in-context learning. This is consistent with our findings. Even in the few-shot setting, fine-tuned language models can outperform in-context learning when comparing models of the same size. Recently, H. Liu et al. (2022) compared parameter-efficient few-shot FT of T0 (Sanh et al., 2022) to in-context learning with GPT-3, finding that their parameter-efficient fine-tuning approach outperforms in-context learning. This is consistent with our findings; however, unlike our work, they only consider in-domain performance.

Focusing on OOD performance, Si et al. (2023) investigate the generalization of GPT-3 along various axes, including generalization under covariate shift – as we do. However, they compare models of different sizes, i.e., RoBERTa-large and GPT-3 (which has 500 times the number of parameters), and different training settings, i.e., fully supervised for fine-tuning vs. few-shot for in-context learning. They observe much better OOD performance for in-context learning than fine-tuning, concluding that in-context learning with GPT-3 is more robust than fine-tuning using BERT or RoBERTa. While this conclusion is valid, it holds for specific models, rather than the methods in general. We show how important it is to compare methods fairly. Based on our comparable results, fine-tuning language models results in similar or even better OOD generalization. Another work that compares the OOD generalization of different adaptation approaches

is Awadalla et al. (2022). Unlike our choice of MNLI and RTE, they investigate the robustness of question answering models under various types of distribution shifts and find that in-context learning is more robust to distribution shifts than fine-tuning. Moreover, they argue that for fine-tuning, increasing model size does not have a strong impact on generalization. However, they don't scale beyond 1.5B parameters. Our findings suggest that the relationship between in-domain and OOD performance does depend on model size.

While we focus on the task adaptation of decoder-only models, Utama et al. (2021) investigate the OOD generalization of encoder-only models adapted via pattern-based few-shot fine-tuning. For MNLI and HANS, they find that these models adopt similar inference heuristics to those trained with vanilla fine-tuning and hence perform poorly OOD. They observe that models rely even more on heuristics when fine-tuned on more data. This is in contrast to our results where we find that pattern-based few-shot fine-tuning can lead to good OOD generalization, and OOD generalization improves as we train on more data. We attribute this to the fact that they experiment with a smaller model (RoBERTa-large; 350M).[17]

Lastly, Bandel et al. (2022) show that masked language models can generalize well on HANS if fine-tuned for a sufficient number of steps. While they focus on fine-tuning on the entire dataset, their findings provide additional evidence that fine-tuned language models can generalize well OOD.

6.8 Conclusions

We perform a fair comparison between in-domain and OOD generalization of two alternative task adaptation strategies: Few-shot in-context learning and fine-tuning. We compare OPT models (S. Zhang et al., 2022) ranging from 125M to 30B parameters on three classification datasets across two tasks. We find that for both approaches, performance improves as models become larger. For the largest

17 This is also related to the results of (Warstadt et al., 2020b), who show that better pre-trained models are less prone to rely on superficial (and potentially spurious) features for predictions.

models we experiment with (OPT-30B), we find that fine-tuning outperforms in-context learning on both in-domain and OOD performance and even improves further as we train on more data.

However, our results also demonstrate that the performance of both fine-tuning and in-context learning exhibits high variance, highlighting that truly robust task adaptation remains an open challenge. We end by providing a high-level comparison between the two approaches, listing the benefits and limitations of each, and discussing some future directions.

6.9 Limitations

We focus on a specific type of OOD generalization, namely, covariate shift (Hupkes et al., 2022). Under this setup, we refer to OOD as the specific challenge datasets we use. As such, different conclusions might be reached by repeating the experiments and evaluating different datasets.

Further, we focus specifically on OPT decoder-only models as our goal was to compare the generalization of adaptation via fine-tuning vs. in-context learning using the same pre-trained model. To the best of our knowledge, existing encoder-only models do not have strong in-context learning abilities. For encoder–decoder models such as T5, only recent variants such as Flan-T5 (Chung et al., 2022) demonstrate the ability to respond well to instructions. However, these models require an additional supervised fine-tuning step on instruction data. This makes it challenging to attribute generalization abilities (or the lack thereof) to specific adaptation techniques (fine-tuning vs in-context learning). Hence, we focus on decoder-only models pre-trained exclusively with a language modeling objective.

Many recent papers that experiment with in-context learning use GPT-3. While fine-tuning GPT-3 is possible via an API, it is unclear what fine-tuning approach is used behind that API. Since this makes a fair comparison difficult, we chose not to experiment with GPT-3.

While similarly large models (e.g. OPT-175B) are publicly available, we do not have the computational resources to run such models. While we expect the trends we observe in this work to hold with larger models, we are not able to empirically test that. Moreover, we only experiment with English language models as, to the best of our knowledge, there are no publicly available models which are similar to OPT (decoder-only models of various sizes trained on the same data) for other languages.

Finally, we only experiment with basic fine-tuning and in-context learning methods. However, for both approaches there exist more advanced techniques which we do not consider (e.g. calibration; Zhao et al., 2021). We note that such techniques can typically be applied for both adaptation approaches. Hence we expect an improvement for one method to improve the other as well.

7

Conclusion and Future Directions

Contents

In this final chapter, we summarize the main contributions of this thesis and discuss how they address the shortcomings posed in §1.2. Additionally, we discuss perspectives for future work.

7.1 Summary of contributions

This thesis provides four analyses on crucial parts of the modern NLP pipeline ranging from investigating the linguistic capabilities of pre-trained language models (Chapter 3) and how these linguistic capabilities are affected by fine-tuning (Chapter 4), to a rigorous analysis of the fine-tuning process itself (Chapter 5), and a critical evaluation and comparison of the generalization of task-adapted language models (Chapter 6). Below, we summarize our main contributions.

Interplay between linguistic knowledge and model performance The first shortcoming of pre-trained and fine-tuned language models we identified is a lack of understanding of the relationship between their strong capabilities on NLP downstream tasks and their encoding of linguistic and factual knowledge, which is presumed to be required to solve these tasks. Our contributions to better understand this relationship are twofold. First, in Chapter 3, we perform a fine-grained analysis of the syntactic knowledge encoded in pre-trained masked language models. Specifically, we investigate the extent to which these models encode linguistic knowledge about relative clauses in American English. Importantly, we compare probing accuracy—a commonly used proxy for linguistic knowledge encoded in pre-trained models—to the actual token predictions of a model. Our results show that while all models achieve high probing accuracy, there exist model-specific differences when evaluating language model predictions directly. Our contributions highlight the importance of building up claims about a model's linguistic knowledge using a more fine-grained evaluation that considers model predictions and performance on challenging edge cases in addition to probing-based evaluation.

We build on this finding in Chapter 4, where we investigate how the linguistic knowledge of pre-trained models changes when fine-tuning them on NLP downstream tasks. We demonstrate that fine-tuning on downstream tasks can considerably affect probing performance, suggesting a change in the amount of linguistic information encoded by the model. However, as we demonstrate, positive changes in probing performance can in large part be attributed to changes in the attention distribution of the fine-tuned model, which result in better sentence representations for probing. These results provide further evidence that probing performance should be carefully interpreted, particularly when comparing probing performance of pre-trained and fine-tuned models.

Taken together, we provide novel insight into how to perform a fine-grained evaluation of the linguistic knowledge of pre-trained language models and on the interaction between probing performance and fine-tuning, and argue for a careful interpretation of the latter.

Fine-tuning stability Next, we focus on providing a better understanding of the large variance in fine-tuning performance, a phenomenon referred to as fine-tuning instability. As motivated in §1.2, previous work attributed the observed instability to the small size of the fine-tuning datasets or catastrophic forgetting. However, these observations were anecdotal and no empirical work had substantiated these claims. In Chapter 5, we rigorously investigated fine-tuning instability and demonstrated that both hypotheses in fact fail to explain the observed instability. We show that fine-tuning instability is mainly a result of optimization difficulties, specifically vanishing gradients, which result in convergence to poorly generalizing local minima. Based on our analysis, we provide simple suggestions to improve fine-tuning stability by training with small learning rates and bias correction.

Our proposed approach leads to strong empirical performance, outperforming previous work in terms of stability while maintaining or even improving overall performance. Moreover, we demonstrate how a critical investigation of the failure cases of fine-tuning can lead to simple but well-motivated changes that have a significant impact on performance, outperform competing approaches, and overall lead to a better understanding of a crucial component of the modern NLP pipeline.

Generalization of task-adapted models Finally, in Chapter 6 we provide a comparison of the in.-domain and out-of-domain generalization of two prominent task adaptation approaches, fine-tuning and in-context learning, making a first step towards addressing an open question about the fundamental differences in generalization between these two adaptation approaches. Our work demonstrates that—in contrast to previous findings—fine-tuned language models can generalize well out-of-domain and in fact, both fine-tuning and in-context learning generalize similarly, both exhibiting large variance in performance and sensitivity to the choice of training examples. Additionally, we find that for both approaches, large models generalize significantly better than small models, highlighting the importance of fair comparisons.

Our findings are an important first step towards a better understanding of the fundamental differences in model behavior from different task adaptation approaches and shows that robust task adaptation remains an open challenge, even with the latest generation of billion parameter language models.

7.2 Future directions

We end this chapter by discussing interesting directions for future work in the context of analyzing pre-trained and fine-tuned language models.

7.2.1 Modular (task-)adaptation

Compared to most of the models studied in this thesis, the latest generation of pre-trained language models have significantly increased in size, with the most capable publicly available models reaching sizes of up to 70B parameters (Touvron et al., 2023b; Touvron et al., 2023a). This increase in model size leads to challenges when it comes to adapting models to new tasks, domains, or languages. For example, gradient-based adaptation via fine-tuning is becoming less viable due to high memory requirements.[1] This trend is particularly affecting academic researchers who often only have a small computational budget for their experiments. A potential solution to this problem is to use parameter-efficient fine-tuning methods, such as LoRA (E. J. Hu et al., 2022), which has shown promising results. Especially when combined with quantization techniques (Dettmers et al., 2023), it becomes feasible to adapt even billion-parameter language models efficiently. However, despite its increasing popularity and empirical success, the mechanisms and limitations of modular task adaptation are currently poorly understood.

1 For example, fine-tuning all parameters of a 30B parameter OPT model requires at least 600GB of GPU memory, i.e. access to 8x A100 80GB GPUs.

To make progress towards a better understanding of modular task adaptation, we propose future work that studies the limits of modular task adaptation, such as investigating whether modular adaptation approaches can be used to adapt a pre-trained model to arbitrary (task) distributions or whether the target distribution must be, in some sense, "close" to the pre-training data.

7.2.2 Limits of update-free task adaptation

A related direction of investigation is the limitations of update-free task adaptation. Due to the strong in-context learning abilities of the latest generation of large language models, it is becoming increasingly common to perform task adaptation via in-context learning, which leaves the pre-trained weights unmodified. There is currently mixed evidence for how adaptation approaches that update parameters perform, compared to those that don't. While H. Liu et al. (2022)'s and our findings in Chapter 6 document the advantages of fine-tuning for classification tasks, Wei et al. (2022a) obtain significantly better performance on challenging reasoning tasks with models that are adapted via in-context learning.

We hope that these findings spur future work that investigates the limitations of update-free adaptation approaches such as in-context learning and we are particularly excited about future work that studies differences in task learnability and sample efficiency for task adaptation approaches with and without weight updates.

7.2.3 Good vs. bad fine-tuning minima

Another direction for future work is inspired by some of our findings in Chapter 6, where we observe that the out-of-domain generalization of individual fine-tuning runs can change during training (§6.5). Notably, these changes in generalization happen at a point in training where the fine-tuned model's training loss suddenly

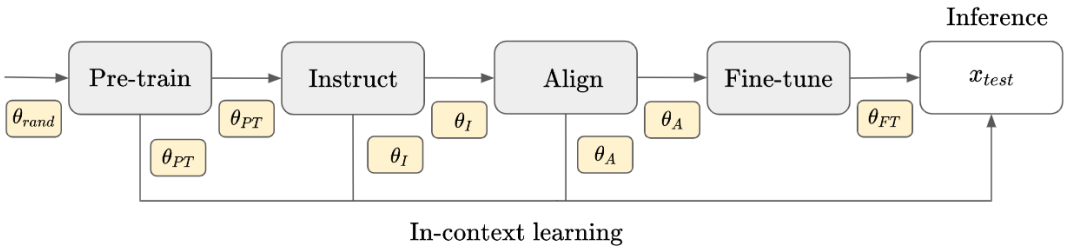


Figure 7.1: The modern NLP pipeline including the additional instruction fine-tuning (Instruct) and alignment to human preferences (Align) steps. Both adaptation steps seamlessly fit into the general pipeline introduced in §1.1 and typically involve updating the model weights via gradient-based fine-tuning.

spikes after approaching zero, and then drops again after. Intuitively, this could suggest that during fine-tuning, stochastic gradient descent visits different minima that are highly similar in their training loss but significantly different in their generalization behavior (we note that this is also consistent with our findings in §5.6.2, where we observe fine-tuning runs that differ quite substantially in their in-domain generalization but all achieve a training loss close to zero). This intuition is supported by recent results from Juneja et al. (2023), who provide evidence for the existence of such minima.

Inspired by these findings, we hope to see future work that investigates the generalization properties of the fine-tuning loss surface and eventually allows us to reliably identify minima in weight space that generalize well.

7.2.4 The pre-train–instruct–align–fine-tune pipeline

Lastly, recent improvements in the capabilities of pre-trained and fine-tuned language models are not just a result of scaling-up the amount of training data and the number of parameters (Kaplan et al., 2020; Hoffmann et al., 2022, *inter alia*), but also come from the addition of new intermediate training steps to the modern NLP pipeline. Two of the now most commonly used intermediate

steps are *instruction fine-tuning* (Sanh et al., 2022; Y. Wang et al., 2022; Chung et al., 2022, *inter alia*) and *alignment to human preferences* (Christiano et al., 2017; Ouyang et al., 2022). We show how instruction fine-tuning and alignment to human preferences extend the modern NLP pipeline in Figure 7.1. Both of these steps essentially adapt the pre-trained model via language modeling on very specific input distributions. Instruction fine-tuning trains on a collection of task instructions while alignment to human preferences requires a dataset of instructions and potential completions and adapts the model to output completions that are preferred by humans[2].

Given the impressive capabilities of models like ChatGPT[3] which have been adapted via instruction fine-tuning and alignment to human preferences, it will be important for future work to rigorously analyze the contributions of the individual adaptation steps to overall model performance. While much recent work is enthusiastic about the capabilities of alignment to human preferences, we are interested in seeing work that also investigates its downsides, such as introducing implicit biases. As for instruction fine-tuning, we are curious about the generalization behavior of instruction fine-tuning, especially in multi-lingual settings. As these questions are similar to the ones posed in this thesis, we hope that the research presented here serves as an inspiration for future work that tackles these challenges and propels the field forward.

2 Human preference is typically approximated by a reward model which outputs a scalar preference value for every completion.

3 https://chat.openai.com/

List of Figures

List of Tables

List of Acronyms

ALBERT A light BERT 27, 33, 35–38, 41–43, 45–47, 49–55, 58, 59, 62, 65–68, 78–80, 83, 90–92, 133, 135, 138, 139, 143, 186, 188, 189, 193, 194, 197, 199, 203

BERT Bidirectional encoder representations from transformer ix, 27, 33, 35–39, 41–47, 49–55, 58, 59, 61, 62, 65–68, 73–75, 77–81, 83, 85–87, 89–92, 94–97, 133–135, 138, 139, 143, 144, 146, 186, 188, 189, 193, 194, 197–199, 201, 204, 205

CoLA Corpus of Linguistic Acceptability 80–82, 87, 88, 93, 95, 96, 135, 138, 139, 144, 189–191, 193, 194, 197, 204

ERM empirical risk minimization 17–19

FT fine-tuning 28, 101–105, 107–123, 136, 137, 144, 145, 147, 148, 209–212, 214–216

HANS Heuristic Analysis for NLI Systems 102, 106, 121, 136, 208

ICL in-context learning 3, 4, 28, 101–110, 112, 114–123, 136, 137, 139–141, 144–148, 209–212, 214–216, 218–220, 226

MLE maximum likelihood estimation 18, 19, 21

MNLI Multi-Genre Natural Language Inference 105–113, 121, 136, 139–142, 144, 147, 148, 208–215, 217, 218, 221, 224, 225, 229

MRPC Microsoft Research Paraphrase Corpus 80–82, 87, 88, 93, 95, 96, 135, 139, 144, 197, 204

NLP natural language processing 1–3, 5, 8, 23, 24, 30, 33, 57, 59, 125–127, 130

OOD out-of-domain generalization 101–103, 105–109, 111–117, 119–122, 136, 210, 215

OPT Open Pre-trained Transformer 26

PBFT pattern-based fine-tuning 102–104, 106, 108, 110, 113–117, 119, 136, 137, 140, 141, 144–146, 209, 210, 221–225

QNLI Question-answering Natural Language Inference 82, 87, 88, 135, 197, 204

QQP Quora Question Pairs 106, 108, 109, 112, 140, 141, 208–211, 220, 223

RC relative clauses 35, 36, 38, 39, 45–51, 53–55, 137, 145, 181–184, 186, 188

RERM regularized empirical risk minimization 20

RoBERTa Robustly optimized BERT approach 27, 33, 35–38, 41–47, 49–55, 58, 59, 61, 62, 65–68, 70, 71, 73–75, 78–80, 83, 89, 91, 92, 133–135, 138, 139, 143, 186, 188, 189, 193, 194, 197, 199, 202

RTE Recognizing Textual Entailment 80–82, 85, 86, 89, 91–96, 101, 102, 105–116, 121, 134–140, 142, 144–149, 189–194, 197, 198, 200–205, 208–217, 219, 222, 227, 228, 230

SQuAD Stanford Question Answering Dataset 138, 146, 189–194

SST-2 Stanford Sentiment Treebank 138, 189–191, 193, 194

Bibliography

Adi, Yossi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg (2016). "Fine-grained analysis of sentence embeddings using auxiliary prediction tasks." In: *arXiv preprint arXiv:1608.04207* (cit. on p. 60).

– (2017). "Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=BJh6Ztuxl (cit. on pp. 31, 38, 59).

Akbik, Alan, Duncan Blythe, and Roland Vollgraf (2018). "Contextual String Embeddings for Sequence Labeling." In: *COLING 2018, 27th International Conference on Computational Linguistics*, pp. 1638–1649 (cit. on p. 41).

Alabi, Jesujoba O., David Ifeoluwa Adelani, **Mosbach, Marius**, and Dietrich Klakow (2022). "Adapting Pre-trained Language Models to African Languages via Multilingual Adaptive Fine-Tuning." In: *Proceedings of the 29th International Conference on Computational Linguistics*. Best paper award 🏆. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, pp. 4336–4349. URL: https://aclanthology.org/2022.coling-1.382 (cit. on p. 119).

Alain, Guillaume and Yoshua Bengio (2016). "Understanding intermediate layers using linear classifier probes." In: *arXiv preprint arXiv:1610.01644* (cit. on p. 31).

Awadalla, Anas, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt (Dec. 2022). "Exploring The Landscape of Distributional Robustness for Question Answering Models." In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Abu Dhabi,

United Arab Emirates: Association for Computational Linguistics, pp. 5971–5987. URL: https://aclanthology.org/2022.findings-emnlp.441 (cit. on pp. 102, 104, 121).

Bandel, Elron, Yoav Goldberg, and Yanai Elazar (Dec. 2022). "Lexical Generalization Improves with Larger Models and Longer Training." In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, pp. 4398–4410. URL: https://aclanthology.org/2022.findings-emnlp.323 (cit. on pp. 105, 121).

Bar-Haim, Roy, Ido Dagan, Bill Dolan, Lisa Ferro, and Danilo Giampiccolo (Jan. 2006). "The second PASCAL recognising textual entailment challenge." In: *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment* (cit. on pp. 64, 82).

Belinkov, Yonatan, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass (July 2017). "What do Neural Machine Translation Models Learn about Morphology?" In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 861–872. DOI: 10.18653/v1/P17-1080. URL: https://aclanthology.org/P17-1080 (cit. on p. 60).

Ben Zaken, Elad, Yoav Goldberg, and Shauli Ravfogel (May 2022). "BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models." In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Dublin, Ireland: Association for Computational Linguistics, pp. 1–9. DOI: 10.18653/v1/2022.acl-short.1. URL: https://aclanthology.org/2022.acl-short.1 (cit. on pp. 29, 101).

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). "Learning long-term dependencies with gradient descent is difficult." In: *IEEE transactions on neural networks* 5.2, pp. 157–166 (cit. on p. 90).

Bentivogli, Luisa, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini (2009). "The Fifth PASCAL Recognizing Textual Entailment Challenge." In: *Proceedings of the Second Text Analysis Conference (TAC 2009)* (cit. on pp. 64, 82).

Besag, Julian (1975). "Statistical Analysis of Non-Lattice Data." In: *Journal of the Royal Statistical Society. Series D (The Statistician)* 24.3, pp. 179–195. ISSN: 00390526, 14679884. URL: http://www.jstor.org/stable/2987782 (visited on 07/26/2023) (cit. on p. 22).

Biber, Douglas, Stig Johansson, Geoffrey Leech, Susan Conrad, and Edward Finegan (1999). *Longman Grammar of Spoken and Written English*. Harlow, UK: Longman (cit. on pp. 35, 36).

Biderman, Stella et al. (2023). *Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling*. arXiv: 2304.01373 [cs.CL] (cit. on pp. 26, 114, 215).

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). "Enriching Word Vectors with Subword Information." In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tacl_a_00051. URL: https://aclanthology.org/Q17-1010 (cit. on p. 41).

Brown, Tom et al. (2020). "Language Models are Few-Shot Learners." In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 1877–1901. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (cit. on pp. 3, 26, 30, 101, 102, 104, 119, 120, 209).

Christiano, Paul F, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei (2017). "Deep Reinforcement Learning from Human Preferences." In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R.

Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf (cit. on p. 131).

Chung, Hyung Won et al. (2022). "Scaling Instruction-Finetuned Language Models." In: *arXiv preprint arXiv:2210.11416*. URL: https://arxiv.org/abs/2210.11416 (cit. on pp. 122, 131).

Conneau, Alexis, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni (July 2018). "What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties." In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 2126–2136. DOI: 10.18653/v1/P18-1198. URL: https://www.aclweb.org/anthology/P18-1198 (cit. on pp. 31, 38, 59, 60, 64).

D'Arcy, Alexandra and Sali A. Tagliamonte (2010). "Prestige, accommodation, and the legacy of relative who." In: *Language in Society* 39.3, pp. 383–410. DOI: 10.1017/S0047404510000205 (cit. on p. 36).

Dagan, Ido, Oren Glickman, and Bernardo Magnini (2005). "The PASCAL Recognising Textual Entailment Challenge." In: *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*. MLCW'05. Southampton, UK: Springer-Verlag, pp. 177–190. ISBN: 3540334270. DOI: 10.1007/11736790_9. URL: https://doi.org/10.1007/11736790_9 (cit. on pp. 64, 82).

– (2006). "The PASCAL Recognising Textual Entailment Challenge." In: *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*. Ed. by Joaquin Quiñonero-Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché-Buc. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 177–190. ISBN: 978-3-540-33428-6 (cit. on pp. 101, 106).

Dao, Tri, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Re (2022). "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness." In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho. URL: https://openreview.net/forum?id=H4DqfPSibmx (cit. on p. 26).

Davies, Mark (2015). *Corpus of Contemporary American English (COCA)*. Version V2. DOI: 10.7910/DVN/AMUDUW. URL: https://doi.org/10.7910/DVN/AMUDUW (cit. on pp. 35, 39, 48, 50).

Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer (2023). *QLoRA: Efficient Finetuning of Quantized LLMs*. arXiv: 2305.14314 [cs.LG] (cit. on pp. 29, 128).

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://www.aclweb.org/anthology/N19-1423 (cit. on pp. 1, 9, 24, 27, 28, 37, 59, 61, 65, 66, 77, 79, 81, 83–87, 90, 91, 95, 96, 101, 103, 114, 119, 135, 144, 204, 205).

Dodge, Jesse, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith (2020a). "Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping." In: *arXiv preprint arXiv:2002.06305* (cit. on pp. 9, 77, 79, 81–84, 86, 195).

Dodge, Jesse, Gabriel Ilharco, Roy Schwartz, Hannaneh Farhadi Aliand Hajishirzi, and Noah A. Smith (2020b). "Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping." In: URL: https://arxiv.org/abs/2002.06305 (cit. on pp. 4, 101).

Dolan, William B. and Chris Brockett (2005). "Automatically Constructing a Corpus of Sentential Paraphrases." In: *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. URL: https://www.aclweb.org/anthology/I05-5002 (cit. on p. 82).

Elazar, Yanai, Hongming Zhang, Yoav Goldberg, and Dan Roth (Nov. 2021). "Back to Square One: Artifact Detection, Training and Commonsense Disentanglement in the Winograd Schema." In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 10486–10500. DOI: 10.18653/v1/2021.emnlp-main.819. URL: https://aclanthology.org/2021.emnlp-main.819 (cit. on p. 105).

Ettinger, Allyson (2020). "What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models." In: *Transactions of the Association for Computational Linguistics* 8, pp. 34–48. DOI: 10.1162/tacl_a_00298. eprint: https://doi.org/10.1162/tacl_a_00298. URL: https://doi.org/10.1162/tacl_a_00298 (cit. on p. 38).

Ettinger, Allyson, Ahmed Elgohary, and Philip Resnik (Aug. 2016). "Probing for semantic evidence of composition by means of simple classification tasks." In: *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. Berlin, Germany: Association for Computational Linguistics, pp. 134–139. DOI: 10.18653/v1/W16-2524. URL: https://aclanthology.org/W16-2524 (cit. on p. 60).

Gao, Leo et al. (2020). *The Pile: An 800GB Dataset of Diverse Text for Language Modeling*. arXiv: 2101.00027 [cs.CL] (cit. on p. 26).

Gao, Leo et al. (Sept. 2021). "A framework for few-shot language model evaluation." In: Zenodo. DOI: 10.5281/zenodo.5371628. URL: https://doi.org/10.5281/zenodo.5371628 (cit. on p. 209).

Gao, Tianyu, Adam Fisch, and Danqi Chen (Aug. 2021). "Making Pre-trained Language Models Better Few-shot Learners." In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, pp. 3816–3830. DOI: 10.18653/v1/2021.acl-long.295. URL: https://aclanthology.org/2021.acl-long.295 (cit. on p. 103).

Giampiccolo, Danilo, Bernardo Magnini, Ido Dagan, and Bill Dolan (2007). "The Third PASCAL Recognizing Textual Entailment Challenge." In: *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. RTE '07. Prague, Czech Republic: Association for Computational Linguistics, pp. 1–9 (cit. on pp. 64, 82).

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256 (cit. on p. 90).

Goldberg, Yoav (2019). "Assessing BERT's syntactic abilities." In: *arXiv preprint arXiv:1901.05287* (cit. on pp. 4, 38, 48, 54).

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press (cit. on pp. 21, 22).

Goyal, Priya, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He (2017). *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*. arXiv: 1706.02677 [cs.CV] (cit. on p. 91).

Hao, Yaru, Li Dong, Furu Wei, and Ke Xu (Nov. 2019). "Visualizing and Understanding the Effectiveness of BERT." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 4143–4152. DOI:

10.18653/v1/D19-1424. URL: https://www.aclweb.org/anthology/D19-1424 (cit. on pp. 92, 94).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." In: *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034 (cit. on p. 90).

– (2016). "Deep residual learning for image recognition." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778 (cit. on p. 91).

Hewitt, John and Percy Liang (Nov. 2019). "Designing and Interpreting Probes with Control Tasks." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2733–2743. DOI: 10.18653/v1/D19-1275. URL: https://www.aclweb.org/anthology/D19-1275 (cit. on p. 66).

Hewitt, John and Christopher D. Manning (June 2019). "A Structural Probe for Finding Syntax in Word Representations." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4129–4138. DOI: 10.18653/v1/N19-1419. URL: https://aclanthology.org/N19-1419 (cit. on pp. 4, 60, 62).

Hochreiter, Sepp (1991). "Untersuchungen zu dynamischen neuronalen Netzen." In: *Diploma, Technische Universität München* 91.1 (cit. on p. 90).

Hoffmann, Jordan et al. (2022). *Training Compute-Optimal Large Language Models*. arXiv: 2203.15556 [cs.CL] (cit. on p. 130).

Houlsby, Neil, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly (2019).

"Parameter-Efficient Transfer Learning for NLP." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 2790–2799. URL: http://proceedings.mlr.press/v97/houlsby19a.html (cit. on p. 29).

Howard, Jeremy and Sebastian Ruder (July 2018). "Universal Language Model Fine-tuning for Text Classification." In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 328–339. DOI: 10.18653/v1/P18-1031. URL: https://aclanthology.org/P18-1031 (cit. on pp. 24, 28, 61, 101, 103, 114, 119).

Hu, Edward J, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen (2022). "LoRA: Low-Rank Adaptation of Large Language Models." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=nZeVKeeFYf9 (cit. on pp. 29, 101, 114, 128).

Hu, Jennifer, Sherry Yong Chen, and Roger Levy (Jan. 2020a). "A closer look at the performance of neural language models on reflexive anaphor licensing." In: *Proceedings of the Society for Computation in Linguistics 2020*. New York, New York: Association for Computational Linguistics, pp. 323–333. URL: https://aclanthology.org/2020.scil-1.39 (cit. on p. 38).

Hu, Jennifer, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy (July 2020b). "A Systematic Assessment of Syntactic Generalization in Neural Language Models." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 1725–1744. URL: https://www.aclweb.org/anthology/2020.acl-main.158 (cit. on p. 38).

Huddleston, Rodney and Geoffrey K. Pullum (2002). *The Cambridge Grammar of the English Language*. Cambridge University Press. DOI: 10.1017/9781316423530 (cit. on p. 35).

Hupkes, Dieuwke, Sara Veldhoen, and Willem Zuidema (2018). "Visualisation and'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure." In: *Journal of Artificial Intelligence Research* 61, pp. 907–926 (cit. on p. 60).

Hupkes, Dieuwke et al. (2022). "State-of-the-art generalisation research in NLP: a taxonomy and review." In: *arXiv* (arXiv). URL: https://arxiv.org/abs/2210.03050 (cit. on pp. 105, 122).

Jia, Robin and Percy Liang (Sept. 2017). "Adversarial Examples for Evaluating Reading Comprehension Systems." In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 2021–2031. DOI: 10.18653/v1/D17-1215. URL: https://aclanthology.org/D17-1215 (cit. on p. 4).

Jiang, Zhengbao, Frank F. Xu, Jun Araki, and Graham Neubig (2020). "How Can We Know What Language Models Know?" In: *Transactions of the Association for Computational Linguistics* 8, pp. 423–438. DOI: 10.1162/tacl_a_00324. URL: https://www.aclweb.org/anthology/2020.tacl-1.28 (cit. on p. 38).

Juneja, Jeevesh, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra (2023). "Linear Connectivity Reveals Generalization Strategies." In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=hY6M0JHl3uL (cit. on pp. 117, 130).

Kaddour, Jean, Oscar Key, Piotr Nawrot, Pasquale Minervini, and Matt J. Kusner (2023). *No Train No Gain: Revisiting Efficient Training Algorithms For Transformer-based Language Models*. arXiv: 2307.06440 [cs.LG] (cit. on p. 24).

Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei (2020).

"Scaling Laws for Neural Language Models." In: *CoRR* abs/2001.08361. arXiv: 2001.08361. URL: https://arxiv.org/abs/2001.08361 (cit. on pp. 24, 130).

Kassner, Nora and Hinrich Schütze (July 2020). "Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 7811–7818. URL: https://www.aclweb.org/anthology/2020.acl-main.698 (cit. on p. 38).

Khot, Tushar, Ashish Sabharwal, and Peter Clark (2018). "SciTaiL: A Textual Entailment Dataset from Science Question Answering." In: URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17368 (cit. on p. 200).

Kim, Najoung et al. (June 2019a). "Probing What Different NLP Tasks Teach Machines about Function Word Comprehension." In: *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 235–249. DOI: 10.18653/v1/S19-1026. URL: https://www.aclweb.org/anthology/S19-1026 (cit. on pp. 35, 38).

– (June 2019b). "Probing What Different NLP Tasks Teach Machines about Function Word Comprehension." In: *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 235–249. DOI: 10.18653/v1/S19-1026. URL: https://aclanthology.org/S19-1026 (cit. on p. 59).

Kingma, Diederik and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization." In: *International Conference on Learning Representations (ICLR)*. San Diega, CA, USA (cit. on p. 23).

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization." In: *International Conference on Learning Representations*. URL: http://arxiv.org/abs/1412.6980 (cit. on p. 90).

Kirkpatrick, James et al. (2017). "Overcoming catastrophic forgetting in neural networks." In: *Proceedings of the National Academy of Sciences* 114.13, pp. 3521–3526. ISSN: 1091-6490. DOI: 10.1073/pnas.1611835114. URL: http://dx.doi.org/10.1073/pnas.1611835114 (cit. on p. 84).

Krasnowska-Kieraś, Katarzyna and Alina Wróblewska (July 2019). "Empirical Linguistic Study of Sentence Embeddings." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 5729–5739. DOI: 10.18653/v1/P19-1573. URL: https://aclanthology.org/P19-1573 (cit. on pp. 60, 67).

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2020). "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=H1eA7AEtvS (cit. on pp. 27, 37, 59, 65, 79, 83, 84).

Le Scao, Teven and Alexander Rush (June 2021). "How many data points is a prompt worth?" In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, pp. 2627–2636. DOI: 10.18653/v1/2021.naacl-main.208. URL: https://aclanthology.org/2021.naacl-main.208 (cit. on p. 29).

Lee, Cheolhyoung, Kyunghyun Cho, and Wanmo Kang (2020). "Mixout: Effective Regularization to Finetune Large-scale Pretrained Language Models." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HkgaETNtDB (cit. on pp. 9, 77, 79–86, 95, 96, 134, 195).

Levene, Howard (1960). "Robust tests for equality of variances." In: *Contributions to probability and statistics: Essays in honor of Harold Hotelling*, pp. 278–292 (cit. on p. 96).

Li, Hao, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein (2018). "Visualizing the Loss Landscape of Neural Nets." In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Curran Associates, Inc., pp. 6389–6399. URL: http://papers.nips.cc/paper/7875-visualizing-the-loss-landscape-of-neural-nets.pdf (cit. on p. 92).

Lin, Yongjie, Yi Chern Tan, and Robert Frank (Aug. 2019). "Open Sesame: Getting inside BERT's Linguistic Knowledge." In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Florence, Italy: Association for Computational Linguistics, pp. 241–253. DOI: 10.18653/v1/W19-4825. URL: https://aclanthology.org/W19-4825 (cit. on p. 60).

Liu, Haokun, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel (2022). "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning." In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho. URL: https://openreview.net/forum?id=rBCvMG-JsPd (cit. on pp. 120, 129).

Liu, Liyuan, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han (2020). "On the Variance of the Adaptive Learning Rate and Beyond." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rkgz2aEKDr (cit. on p. 81).

Liu, Nelson F., Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith (June 2019). "Linguistic Knowledge and Transferability of Contextual Representations." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 1073–1094. DOI: 10.18653/v1/

N19-1112. URL: https://www.aclweb.org/anthology/N19-1112 (cit. on pp. 4, 38, 60, 61, 66, 68).

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019a). "RoBERTa: A Robustly Optimized BERT Pretraining Approach." In: *CoRR* abs/1907.11692. arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (cit. on pp. 1, 27, 102).

– (2019b). "Roberta: A robustly optimized bert pretraining approach." In: *arXiv preprint arXiv:1907.11692* (cit. on pp. 26, 37, 59, 65, 79, 83, 84).

Logan IV, Robert, Ivana Balazevic, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel (May 2022). "Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models." In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, pp. 2824–2835. DOI: 10.18653/v1/2022.findings-acl.222. URL: https://aclanthology.org/2022.findings-acl.222 (cit. on pp. 28, 104, 106).

Loshchilov, Ilya and Frank Hutter (2019). "Decoupled Weight Decay Regularization." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Bkg6RiCqY7 (cit. on pp. 24, 83).

Lu, Yao, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp (May 2022). "Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity." In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, pp. 8086–8098. DOI: 10.18653/v1/2022.acl-long.556. URL: https://aclanthology.org/2022.acl-long.556 (cit. on pp. 4, 5, 101, 106).

Martins, Pedro Henrique, Zita Marinho, and Andre Martins (May 2022). "∞-former: Infinite Memory Transformer." In: *Proceedings of the 60th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, pp. 5468–5485. DOI: 10.18653/v1/2022.acl-long.375. URL: https://aclanthology.org/2022.acl-long.375 (cit. on p. 119).

Marvin, Rebecca and Tal Linzen (2018). "Targeted Syntactic Evaluation of Language Models." In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 1192–1202. DOI: 10.18653/v1/D18-1151. URL: https://www.aclweb.org/anthology/D18-1151 (cit. on p. 38).

McCloskey, Michael and Neal J Cohen (1989). "Catastrophic interference in connectionist networks: The sequential learning problem." In: *Psychology of learning and motivation*. Vol. 24. Elsevier, pp. 109–165 (cit. on p. 84).

McCoy, R. Thomas, Junghyun Min, and Tal Linzen (Nov. 2020). "BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance." In: *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Online: Association for Computational Linguistics, pp. 217–227. DOI: 10.18653/v1/2020.blackboxnlp-1.21. URL: https://www.aclweb.org/anthology/2020.blackboxnlp-1.21 (cit. on p. 196).

McCoy, Tom, Ellie Pavlick, and Tal Linzen (July 2019). "Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 3428–3448. DOI: 10.18653/v1/P19-1334. URL: https://www.aclweb.org/anthology/P19-1334 (cit. on pp. 4, 79, 105, 106).

Merchant, Amil, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney (2020). "What Happens To BERT Embeddings During Fine-tuning?" In: *arXiv preprint arXiv:2004.14448* (cit. on pp. 61, 62).

Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher (2016). "Pointer sentinel mixture models." In: *arXiv preprint arXiv:1609.07843* (cit. on p. 85).

– (2017). "Pointer Sentinel Mixture Models." In: *ArXiv* abs/1609.07843 (cit. on p. 74).

Mikolov, Tomas, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin (2018). "Advances in Pre-Training Distributed Word Representations." In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)* (cit. on p. 41).

Min, Sewon, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer (Dec. 2022). "Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?" In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, pp. 11048–11064. URL: https://aclanthology.org/2022.emnlp-main.759 (cit. on pp. 5, 101, 119).

Niven, Timothy and Hung-Yu Kao (July 2019). "Probing Neural Network Comprehension of Natural Language Arguments." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 4658–4664. DOI: 10.18653/v1/P19-1459. URL: https://www.aclweb.org/anthology/P19-1459 (cit. on pp. 4, 79).

OpenAI (2023). "GPT-4 Technical Report." In: *arXiv preprint arXiv:2303.08774*. URL: https://arxiv.org/abs/2303.08774 (cit. on pp. 2, 3, 101).

Ouyang, Long et al. (2022). *Training language models to follow instructions with human feedback*. arXiv: 2203.02155 [cs.CL] (cit. on p. 131).

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12, pp. 2825–2830 (cit. on p. 41).

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). "GloVe: Global Vectors for Word Representation." In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://www.aclweb.org/anthology/D14-1162 (cit. on p. 41).

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (June 2018). "Deep Contextualized Word Representations." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: https://aclanthology.org/N18-1202 (cit. on pp. 24, 60, 101).

Petroni, Fabio, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller (Nov. 2019). "Language Models as Knowledge Bases?" In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2463–2473. DOI: 10.18653/v1/D19-1250. URL: https://www.aclweb.org/anthology/D19-1250 (cit. on pp. 4, 38).

Pfeiffer, Jonas, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe (July 2022). "Lifting the Curse of Multilinguality by Pre-training Modular Transformers." In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, pp. 3479–3495. DOI: 10.18653/v1/2022.naacl-main.255. URL: https://aclanthology.org/2022.naacl-main.255 (cit. on p. 119).

Phang, Jason, Thibault Févry, and Samuel R Bowman (2018). "Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks." In: *arXiv preprint arXiv:1811.01088* (cit. on pp. 61, 79, 81–84, 86, 87, 195).

Press, Ofir, Noah Smith, and Mike Lewis (2022a). "Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=R8sQPpGCv0 (cit. on p. 119).

Press, Ofir, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis (2022b). "Measuring and Narrowing the Compositionality Gap in Language Models." In: *arXiv preprint arXiv:2210.03350* (cit. on p. 101).

Pruksachatkun, Yada, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman (July 2020a). "Intermediate-Task Transfer Learning with Pretrained Language Models: When and Why Does It Work?" In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 5231–5247. URL: https://www.aclweb.org/anthology/2020.acl-main.467 (cit. on p. 8).

– (July 2020b). "Intermediate-Task Transfer Learning with Pretrained Language Models: When and Why Does It Work?" In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 5231–5247. DOI: 10.18653/v1/2020.acl-main.467. URL: https://www.aclweb.org/anthology/2020.acl-main.467 (cit. on pp. 61, 62, 86).

Quirk, Randolph (1957). "Relative Clauses in Educated Spoken English." In: *English Studies* 38, pp. 97–109 (cit. on p. 36).

Radford, Alec and Karthik Narasimhan (2018). "Improving Language Understanding by Generative Pre-Training." In: (cit. on p. 25).

Radford, Alec, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever (2019). "Language Models are Unsupervised Multitask Learners." In: *OpenAI Blog* 1.8, p. 9 (cit. on pp. 26, 60).

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang (Nov. 2016). "SQuAD: 100,000+ Questions for Machine Comprehension of Text." In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 2383–2392. DOI: 10.18653/v1/D16-1264. URL: https://www.aclweb.org/anthology/D16-1264 (cit. on pp. 63, 64, 82).

Rasley, Jeff, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He (2020). "DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters." In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '20. Virtual Event, CA, USA: Association for Computing Machinery, pp. 3505–3506. ISBN: 9781450379984. DOI: 10.1145/3394486.3406703. URL: https://doi.org/10.1145/3394486.3406703 (cit. on p. 208).

Al-Rfou, Rami, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones (2019). "Character-Level Language Modeling with Deeper Self-Attention." In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI'19/IAAI'19/EAAI'19. Honolulu, Hawaii, USA: AAAI Press. ISBN: 978-1-57735-809-1. DOI: 10.1609/aaai.v33i01.33013159. URL: https://doi.org/10.1609/aaai.v33i01.33013159 (cit. on p. 25).

Rogers, Anna, Olga Kovaleva, and Anna Rumshisky (2020). "A Primer in BERTology: What We Know About How BERT Works." In: *Transactions of the Association for Computational Linguistics* 8, pp. 842–866. DOI: 10.1162/tacl_a_00349. URL: https://aclanthology.org/2020.tacl-1.54 (cit. on pp. 35, 38, 54, 59).

Rohdenburg, Günter (Aug. 2014). "Relative Clauses of Reason in British and American English." In: *American Speech* 89.3, pp. 288–311. ISSN: 0003-1283. DOI: 10.1215/00031283-2848978. eprint: https://read.dukeupress.edu/american-speech/article-pdf/89/3/288/395584/ASp89.3E.2Rohdenburg.pdf. URL: https://doi.org/10.1215/00031283-2848978 (cit. on p. 35).

Roller, Stephen et al. (Apr. 2021). "Recipes for Building an Open-Domain Chatbot." In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, pp. 300–325. DOI: 10.18653/v1/2021.eacl-main.24. URL: https://aclanthology.org/2021.eacl-main.24 (cit. on p. 26).

Rubner, Y., C. Tomasi, and L. J. Guibas (1998). "A metric for distributions with applications to image databases." In: *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 59–66 (cit. on p. 72).

Sanh, Victor et al. (2022). "Multitask Prompted Training Enables Zero-Shot Task Generalization." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=9Vrb9D0WI4 (cit. on pp. 120, 131).

Schick, Timo, Helmut Schmid, and Hinrich Schütze (Dec. 2020). "Automatically Identifying Words That Can Serve as Labels for Few-Shot Text Classification." In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 5569–5578. DOI: 10.18653/v1/2020.coling-main.488. URL: https://aclanthology.org/2020.coling-main.488 (cit. on pp. 28, 104).

Schick, Timo and Hinrich Schütze (Apr. 2021). "Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference." In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, pp. 255–269. DOI: 10.18653/v1/2021.eacl-main.20. URL: https://aclanthology.org/2021.eacl-main.20 (cit. on pp. 28, 103, 119).

Schmidt, R. M., F. Schneider, and P. Hennig (July 2021). "Descending through a Crowded Valley - Benchmarking Deep Learning Optimizers." In: *Proceedings of 38th International Conference on Machine Learning (ICML)*. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 9367–9376. URL: https://proceedings.mlr.press/v139/schmidt21a.html (cit. on p. 23).

Sharma, Lakshay, L. Graesser, Nikita Nangia, and Utku Evci (2019). "Natural Language Understanding with the Quora Question Pairs Dataset." In: *ArXiv* abs/1907.01041 (cit. on p. 106).

Shi, Xing, Inkit Padhi, and Kevin Knight (Nov. 2016). "Does String-Based Neural MT Learn Source Syntax?" In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 1526–1534. DOI: 10.18653/v1/D16-1159. URL: https://aclanthology.org/D16-1159 (cit. on p. 60).

Shwartz-Ziv, Ravid, Micah Goldblum, Hossein Souri, Sanyam Kapoor, Chen Zhu, Yann LeCun, and Andrew Gordon Wilson (2022). "Pre-Train Your Loss: Easy Bayesian Transfer Learning with Informative Priors." In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho. URL: https://openreview.net/forum?id=YCniF6_3Jb (cit. on p. 117).

Si, Chenglei, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang (2023). "Prompting GPT-3 To Be Reliable." In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=98p5x51L5af (cit. on pp. 5, 10, 102, 104, 109, 120, 147, 210).

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts (Oct. 2013). "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank." In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle,

Washington, USA: Association for Computational Linguistics, pp. 1631–1642. URL: https://aclanthology.org/D13-1170 (cit. on p. 63).

Su, Jianlin, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu (2022). *RoFormer: Enhanced Transformer with Rotary Position Embedding*. arXiv: 2104.09864 [cs.CL] (cit. on p. 26).

Talmor, Alon, Yanai Elazar, Yoav Goldberg, and Jonathan Berant (2019). "oLMpics–On what Language Model Pre-training Captures." In: *arXiv preprint arXiv:1912.13283* (cit. on p. 61).

Tam, Derek, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel (Nov. 2021). "Improving and Simplifying Pattern Exploiting Training." In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 4980–4991. DOI: 10.18653/v1/2021.emnlp-main.407. URL: https://aclanthology.org/2021.emnlp-main.407 (cit. on pp. 28, 104).

Tenney, Ian, Dipanjan Das, and Ellie Pavlick (July 2019a). "BERT Rediscovers the Classical NLP Pipeline." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 4593–4601. DOI: 10.18653/v1/P19-1452. URL: https://aclanthology.org/P19-1452 (cit. on pp. 4, 59, 68).

Tenney, Ian et al. (2019b). "What do you learn from context? Probing for sentence structure in contextualized word representations." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SJzSgnRcKX (cit. on pp. 38, 60–62, 66, 68).

Touvron, Hugo et al. (2023a). *Llama 2: Open Foundation and Fine-Tuned Chat Models*. arXiv: 2307.09288 [cs.CL] (cit. on p. 128).

Touvron, Hugo et al. (2023b). *LLaMA: Open and Efficient Foundation Language Models*. arXiv: 2302.13971 [cs.CL] (cit. on p. 128).

Trinh, Trieu H. and Quoc V. Le (2019). *A Simple Method for Commonsense Reasoning*. arXiv: 1806.02847 [cs.AI] (cit. on p. 26).

Utama, Prasetya, Nafise Sadat Moosavi, Victor Sanh, and Iryna Gurevych (Nov. 2021). "Avoiding Inference Heuristics in Few-shot Prompt-based Finetuning." In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 9063–9074. DOI: 10.18653/v1/2021.emnlp-main.713. URL: https://aclanthology.org/2021.emnlp-main.713 (cit. on pp. 105, 112, 121).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need." In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf (cit. on pp. 24, 25).

Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman (2019a). "SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems." In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., pp. 3266–3280. URL: http://papers.nips.cc/paper/8589-superglue-a-stickier-benchmark-for-general-purpose-language-understanding-systems.pdf (cit. on p. 79).

– (2019b). "SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems." In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf (cit. on p. 2).

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman (Nov. 2018). "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding." In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, pp. 353–355. DOI: 10.18653/v1/W18-5446. URL: https://www.aclweb.org/anthology/W18-5446 (cit. on pp. 2, 59, 61, 63, 79, 81, 82).

Wang, Alex et al. (July 2019c). "Can You Tell Me How to Get Past Sesame Street? Sentence-Level Pretraining Beyond Language Modeling." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 4465–4476. DOI: 10.18653/v1/P19-1439. URL: https://aclanthology.org/P19-1439 (cit. on p. 59).

Wang, Ben and Aran Komatsuzaki (May 2021). *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model*. https://github.com/kingoflolz/mesh-transformer-jax (cit. on p. 26).

Wang, Yizhong et al. (Dec. 2022). "Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks." In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, pp. 5085–5109. URL: https://aclanthology.org/2022.emnlp-main.340 (cit. on p. 131).

Warstadt, Alex and Samuel R. Bowman (2019). *Linguistic Analysis of Pretrained Sentence Encoders with Acceptability Judgments*. arXiv: 1901.03438 [cs.CL] (cit. on pp. 35, 39).

Warstadt, Alex, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman (2020a). "BLiMP: The Benchmark of Linguistic Minimal Pairs for English." In: *Transactions of the Association for*

Computational Linguistics 8, pp. 377–392. DOI: 10.1162/tacl_a_00321. URL: https://www.aclweb.org/anthology/2020.tacl-1.25 (cit. on p. 38).

Warstadt, Alex, Amanpreet Singh, and Samuel R Bowman (2018). "Neural Network Acceptability Judgments." In: *arXiv preprint arXiv:1805.12471* (cit. on pp. 63, 82).

Warstadt, Alex, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman (Nov. 2020b). "Learning Which Features Matter: RoBERTa Acquires a Preference for Linguistic Generalizations (Eventually)." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 217–235. DOI: 10.18653/v1/2020.emnlp-main.16. URL: https://aclanthology.org/2020.emnlp-main.16 (cit. on pp. 4, 121).

Warstadt, Alex et al. (Nov. 2019). "Investigating BERT's Knowledge of Language: Five Analysis Methods with NPIs." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2877–2887. DOI: 10.18653/v1/D19-1286. URL: https://www.aclweb.org/anthology/D19-1286 (cit. on p. 38).

Webson, Albert and Ellie Pavlick (July 2022). "Do Prompt-Based Models Really Understand the Meaning of Their Prompts?" In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, pp. 2300–2344. DOI: 10.18653/v1/2022.naacl-main.167. URL: https://aclanthology.org/2022.naacl-main.167 (cit. on pp. 4, 29, 106, 107, 117).

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou (2022a). "Chain of Thought Prompting Elicits Reasoning in Large Language Models." In: *Advances in*

Neural Information Processing Systems. Ed. by Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho. URL: https://openreview.net/forum?id=_VjQlMeSB_J (cit. on pp. 5, 30, 101, 117, 119, 129).

Wei, Jason et al. (2022b). "Emergent Abilities of Large Language Models." In: *Transactions on Machine Learning Research*. Survey Certification. ISSN: 2835-8856. URL: https://openreview.net/forum?id=yzkSU5zdwD (cit. on pp. 3, 5).

Wilcox, Ethan, Peng Qian, Richard Futrell, Miguel Ballesteros, and Roger Levy (June 2019). "Structural Supervision Improves Learning of Non-Local Grammatical Dependencies." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 3302–3312. DOI: 10.18653/v1/N19-1334. URL: https://www.aclweb.org/anthology/N19-1334 (cit. on p. 38).

Williams, Adina, Nikita Nangia, and Samuel Bowman (June 2018). "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 1112–1122. DOI: 10.18653/v1/N18-1101. URL: https://www.aclweb.org/anthology/N18-1101 (cit. on p. 105).

Wolf, Thomas et al. (Oct. 2020). "Transformers: State-of-the-Art Natural Language Processing." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6 (cit. on pp. 41, 208).

Wong, Eric, Leslie Rice, and J. Zico Kolter (2020). "Fast is better than free: Revisiting adversarial training." In: *International Conference on Learning Rep-*

resentations. URL: https://openreview.net/forum?id=BJx040EFvH (cit. on p. 91).

Xiong, Ruibin, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu (2020). "On layer normalization in the transformer architecture." In: *arXiv preprint arXiv:2002.04745* (cit. on p. 81).

You, Yang, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh (2020). "Large Batch Optimization for Deep Learning: Training BERT in 76 minutes." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Syx4wnEtvH (cit. on p. 90).

Zhang, Susan et al. (2022). *OPT: Open Pre-trained Transformer Language Models*. arXiv: 2205.01068 [cs.CL] (cit. on pp. 26, 101, 103, 105, 121, 147, 210).

Zhang, Tianyi, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi (2021). "Revisiting Few-sample BERT Fine-tuning." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=cO1IH43yUF (cit. on p. 92).

Zhang, Yuan, Jason Baldridge, and Luheng He (June 2019). "PAWS: Paraphrase Adversaries from Word Scrambling." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 1298–1308. DOI: 10.18653/v1/N19-1131. URL: https://www.aclweb.org/anthology/N19-1131 (cit. on p. 106).

Zhao, Zihao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh (2021). "Calibrate Before Use: Improving Few-shot Performance of Language Models." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning

Research. PMLR, pp. 12697–12706. URL: https://proceedings.mlr.press/v139/zhao21c.html (cit. on pp. 117, 123).

Zhu, Chen, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu (2020). "FreeLB: Enhanced Adversarial Training for Natural Language Understanding." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=BygzbyHFvB (cit. on p. 86).

Zhu, Dawei, Xiaoyu Shen, **Mosbach, Marius**, Andreas Stephan, and Dietrich Klakow (July 2023). "Weaker Than You Think: A Critical Look at Weakly Supervised Learning." In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Best paper award 🏆. Toronto, Canada: Association for Computational Linguistics, pp. 14229–14253. URL: https://aclanthology.org/2023.acl-long.796 (cit. on p. 113).

Zhu, Yukun, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler (2015). *Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books*. arXiv: 1506.06724 [cs.CV] (cit. on p. 26).

A

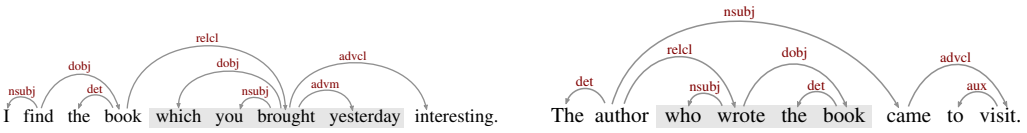
Probing Pre-trained Models for Linguistic Knowledge

Contents

A.1 Probing dataset

Figure A.1 shows how to identify sentences containing RCs using a dependency parse tree, which were obtained using SpaCy. A sentence with an RC can be identified if the antecedent has an outgoing edge with the tag RELCL. The RC in the sentence can be further extracted since the main verb in the RC would have an incoming edge with the tag RELCL, and the subtree where RC main verb is the head constitutes the RC. This procedure enables us to extract sentences with RCs that are grammatical from text.

However, probing classifiers for grammatical acceptability require both grammatical and ungrammatical sentences. Since grammaticality of RCs in English



(a) **(b)**

Figure A.1: Visualized dependency parse trees extracted by SpaCy: (a) object RC, and (b) subject RC.

depends on restrictiveness, animacy of the head noun, and whether the relativizer occupies the subject or the object position in the sentence, we populate three meta-data variables for each grammatical sentence with respect to these aspects of RCs: ANIMATE, RESTRICTIVE, and SUBJRC. The variables ANIMATE and RESTRICTIVE are populated using a set of hand-crafted rules for the usage of RCs in American English (illustrated in Figure A.2), while the variable SUBJRC is populated using the incoming edge to the relativizer in the parse tree (e.g. NSUBJ for subject RC vs. DOBJ for object RC). We discard all sentences where at least one meta-data variable cannot be populated with certainty.

Using the aforementioned annotation procedure, and given the values of the three meta-data variables, each sentence can be manipulated to create an ungrammatical sentence that forms a minimal pair with the original sentence. The resulting paradigms from the three meta-data variables and the set of all possible modifications that be applied to each paradigm are presented in Table A.1. Each grammatical sentence in the dataset is manipulated using all possible modifications of the paradigm to generate a "bag-of-sentences" associated with each sentence (where the original grammatical sentence is a member of). To create the final dataset, we sample one sentence from each bag and construct a balanced dataset of 48,060 sentences with grammatical acceptability labels. Tables A.2 and A.3 show summary statistics of the dataset.

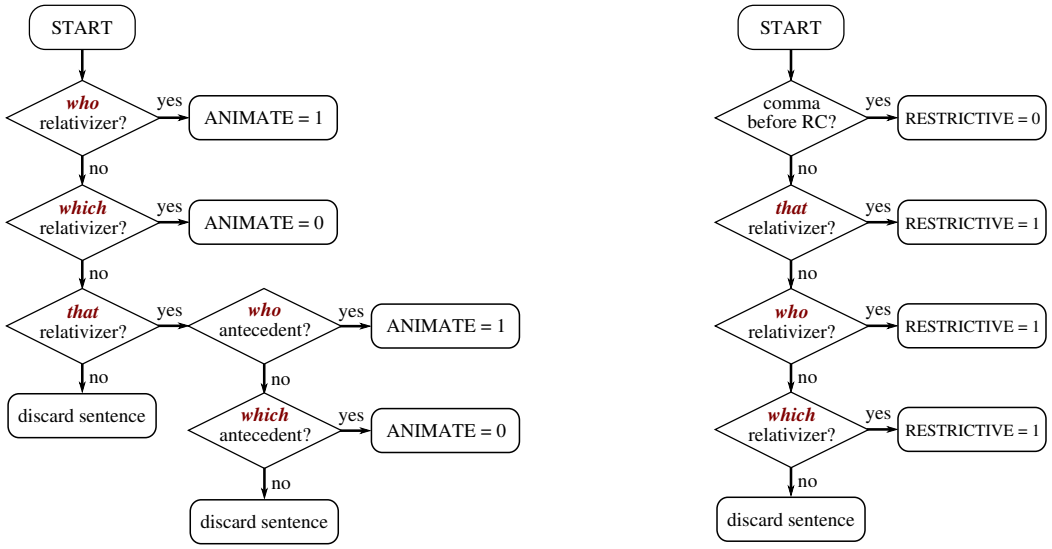


Figure A.2: Annotation decision process for meta-data variables: (a) *Animate*, relativizer **who** and **which** are directly categorized as animate and non-animate, respectively. The relativizer **that** can be either way; thus, we categorize these sentences based on the antecedent. We compile two disjoints sets for antecedents that exclusively occur either with **who** or **which**, and the decision is made based on the membership of the antecedent. If the antecedent is not a member of either set, the sentence is discarded. (b) *Restrictive* can be easily identified since non-restrictive RCs in American English are always preceded by comma ",".

A.2 Probing results

Figure A.3 shows layer-wise probing accuracies for all models introduced in §3.1.1. CLS-pooling clearly leads to worse sentence representations performing on par with the non-contextualized GloVe and fasttext mean-embeddings for most layers.

Animate	Restrictive	Subject RC	Modification	Label	Example
1	1	1	None	1	Katrina Haus was a woman **who** sought to attract stares, not turn them away.
			Relativizer omission	0	*Katrina Haus was a woman sought to attract stares, not turn them away.
			who → which	0	*Katrina Haus was a woman **which** sought to attract stares, not turn them away.
1	1	0	None	1	Two people **who** I loved very much have left me .
			Relativizer omission	1	Two people I loved very much have left me .
			who → which	0	*Two people **which** I loved very much have left me .
1	0	1	None	1	Buck , **who** was snoozing over in the corner , woke up and looked around.
			Relativizer omission	0	*Buck , was snoozing over in the corner , woke up and looked around.
			who →which	0	*Buck , **which** was snoozing over in the corner , woke up and looked around.
1	0	0	None	1	Sally turned with a welcoming grin , expecting to see Gus , **whom** she liked a lot.
			Relativizer omission	0	*Sally turned with a welcoming grin , expecting to see Gus , she liked a lot
			who →which	0	*Sally turned with a welcoming grin , expecting to see Gus , **which** she liked a lot
0	1	1	None	1	One is a rather, um, disconcerting bit of information **which** has reached my ears.
			Relativizer omission	0	*One is a rather, um, disconcerting bit of information has reached my ears.
			which → who	0	*One is a rather, um, disconcerting bit of information **who** has reached my ears.
			which → that	1	One is a rather, um, disconcerting bit of information **that** has reached my ears.
0	1	0	None	1	Pulls out a course catalog, various forms, and a letter **which** she hands to Kevin.
			Relativizer omission	0	*Pulls out a course catalog, various forms, and a letter she hands to Kevin.
			which → who	0	*Pulls out a course catalog, various forms, and a letter **who** she hands to Kevin.
			which → that	1	Pulls out a course catalog, various forms, and a letter **that** she hands to Kevin.
0	0	1	None	1	I never saw a penny in royalties, **which** was all right with me.
			Relativizer omission	0	*I never saw a penny in royalties, was all right with me.
			which → who	0	*I never saw a penny in royalties, **who** was all right with me.
0	0	0	None	1	Lyric clips her Walkman to her fanny pack, **which** she wears pouch forward.
			Relativizer omission	0	*Lyric clips her Walkman to her fanny pack, she wears pouch forward.
			which → who	0	*Lyric clips her Walkman to her fanny pack, **who** she wears pouch forward.

Table A.1: Examples of the generated paradigms and the possible modifications for each paradigm. It is worth pointing out that not all sentence modifications yield an unacceptable sentence. For example, the modification *relativizer omission* keeps the sentence grammatical in restrictive object RCs.

A.2.1 ALBERT-base-v1 vs. ALBERT-xxlarge-v1

Figure A.4 shows layer-wise probing accuracies for ALBERT-base-v1 (ALBERT) and ALBERT-xxlarge-v1. ALBERT-xxlarge-v1 contains 235M parameters and hence roughly 10x as many as ALBERT-base-v1 and 2x as many as BERT and RoBERTa. As the results in Figure A.4 show, the larger number of parameters indeed results in a significantly higher probing accuracy, outperforming all other models by a large margin. However, it should be noted that the larger number

	Total	**Acceptable**		**Animate**		**Restrictive**	
		0	1	0	1	0	1
Training	42720	21360	21360	11106	31614	25947	16773
Test	5340	2670	2670	1439	3901	3229	2111

Table A.2: Summary statistics of the dataset splits used in Chapter 3] The dataset is balanced with respect to acceptability judgment (the probing label **Acceptable**). **Animate** and **restrictive** are meta-data variables, they are only used for creating the data and the modifications, not for probing.

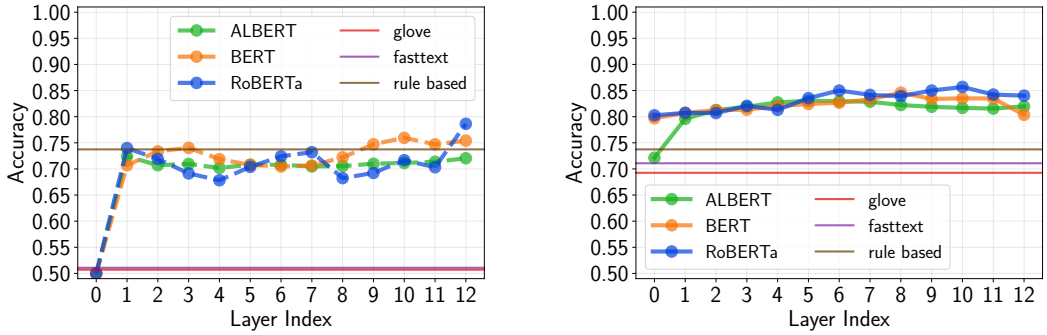
	No modif.		Rel. omission		who → which		which → who		which → that	
Acceptability	0	1	0	1	0	1	0	1	0	1
Training	–	20170	10512	116	8291	–	2557	–	–	1074
Test	–	2670	1299	19	1025	–	346	–	–	125

Table A.3: Summary statistics of the dataset with respect to the modifications. Note that modifications are not balanced.

of parameters is the results of a larger dimensionality of ALBERT-xxlarge-v1's embeddings space, which is 5x larger that that of ALBERT-base-v1, BERT, and RoBERTa (4096 vs. 768). This makes the obtained probing results not directly comparable.

A.2.2 Qualitative analysis for predicted type of antecedent

Table A.5 shows percentage for predicted types of antecedents: (a) identical to target, (b) synonym, (c) hypernym, general noun, determiner, pronoun, (d) hyponym (i.e. semantically lower in hierarchy and thus more specific), or (e) not directly related (i.e. no direct hierarchical relationship) or completely unrelated.



(a) CLS-pooling **(b)** mean-pooling

Figure A.3: Side-by-side comparison of layer-wise probing accuracy on the test set for pre-trained transformer and baseline models using (a) CLS-pooling and (b) mean-pooling.

Results show ALBERT to perform worst as the majority falls into not directly related/unrelated antecedents or hypernyms (more general words). RoBERTa is quite good at predicting *identical* targets outperforming the other two models, especially in object RCs with *who*. While the *which* case in object RCs is harder for all models, RoBERTa still makes best predictions considering identical and synonym prediction. All models perform less well on subject RCs, especially for *which* and *that* (40% to $>$ 50% of unrelated/not directly related targets). Interestingly, for animate antecedents (*who* relativizer), the majority for all models falls into hypernyms, i.e. more general variants. This is especially the case for BERT and ALBERT.

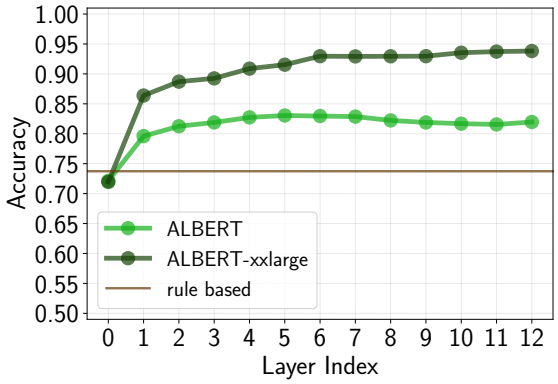


Figure A.4: Comparison of ALBERT-base-v1 and ALBERT-xxlarge-v1 using mean-pooling. Table A.4 shows the results of the best ALBERT-xxlarge-v1 probing classifier compared to all other models grouped by modification.

Modification	GloVE	fasttext	rule based	BERT	RoBERTa	ALBERT	ALBERT-xxlarge
no modification	67.3	69.2	100	83.7 (78.4)	85.3 (79.7)	83.6 (71.5)	95.0 (70.6)
relativizer omission	77.2	85.3	97.6	96.0 (98.3)	95.1 (98.2)	96.4 (94.8)	96.2 (93.5)
who → which	88.0	81.7	0.00	84.9 (88.4)	87.3 (89.6)	78.8 (68.3)	94.6 (69.0)
which → who	18.2	21.3	0.00	47.9 (1.73)	50.0 (0.02)	44.2 (18.7)	80.6 (21.3)
which → that	12.8	8.0	100	80.0 (52.0)	77.6 (44.8)	72.0 (22.4)	73.6 (33.6)
total	69.2	71.0	73.7	84.6 (79.7)	85.5 (80.1)	83.0 (72.1)	93.8 (71.9)

Table A.4: Test accuracy (in %) grouped by modification type (cf. Table A.3 for statistics). For BERT, RoBERTa, ALBERT-base-v1 (ALBERT) and ALBERT-xxlarge-v1 we select the best model according to the probing results shown in Figure 3.2. Numbers in parenthesis show the accuracy of the non-contextualized baseline (layer 0) for each model. ALBERT-xxlarge-v1 performs especially well on the `which` → `who` modification.

	type	**objRC**			**subjRC**		
		who	which	that	who	which	that
BERT	identical	**0.38**	0.22	**0.38**	0.27	0.31	0.29
	synonym	0.03	0.08	0.10	0.10	0.06	0.02
	hypernym	0.07	0.19	0.14	**0.40**	0.17	0.16
	hyponym	0.17	0.00	0.10	0.15	0.06	0.08
	unrelated/not directly related	0.34	**0.50**	0.29	0.10	**0.40**	**0.45**
RoBERTa	identical	**0.48**	0.22	**0.52**	0.31	0.27	0.29
	synonym	0.00	0.11	0.00	0.10	0.15	0.08
	hypernym	0.21	0.19	0.05	**0.31**	0.08	0.08
	hyponym	0.17	0.00	0.10	0.12	0.08	0.08
	unrelated/not directly related	0.14	**0.33**	0.47	0.16	**0.47**	**0.42**
ALBERT	identical	0.28	0.14	0.29	0.20	0.15	0.14
	synonym	0.03	0.06	0.00	0.02	0.08	0.00
	hypernym	0.21	0.19	0.24	**0.43**	0.10	0.16
	hyponym	0.17	0.03	0.14	0.08	0.08	0.10
	unrelated/not directly related	**0.31**	**0.58**	**0.33**	0.27	**0.58**	**0.59**

Table A.5: Predicted **types of antecedents** by relativizer in percentage. The type hypernym encompasses also general nouns, determiners, and pronouns.

B

On the Interplay Between Fine-tuning and Probing

Contents

B.1 Hyperparameters and task statistics

Table B.1 shows hyperparamters used when fine-tuning BERT, RoBERTa, and ALBERT on CoLA, SST-2, RTE, and SQuAD. On SST-2 training for a single epoch was sufficient and we didn't observe a significant improvement when training for more epochs.

Table B.2 shows number of training and development samples for each of the fine-tuning datasets considered in our experiments. Additionally, we report the metric used to evaluate performance for each of the tasks.

Hyperparameter	**Value**
Learning rate	2e−5
Warmup steps	10%
Learning rate schedule	warmup-constant
Batch size	32
Epochs	3 (1 for SST-2)
Weight decay	0.01
Dropout	0.1
Attention dropout	0.1
Classifier dropout	0.1
Adam ϵ	1e−8
Adam β_1	0.9
Adam β_2	0.99
Max. gradient norm	1.0

Table B.1: Hyperparamters used when fine-tuning.

B.2 Additional results

Table B.3 shows the effect of fine-tuning on RTE and SQuAD on the layer-wise accuracy for all three encoder models across the three probing tasks.

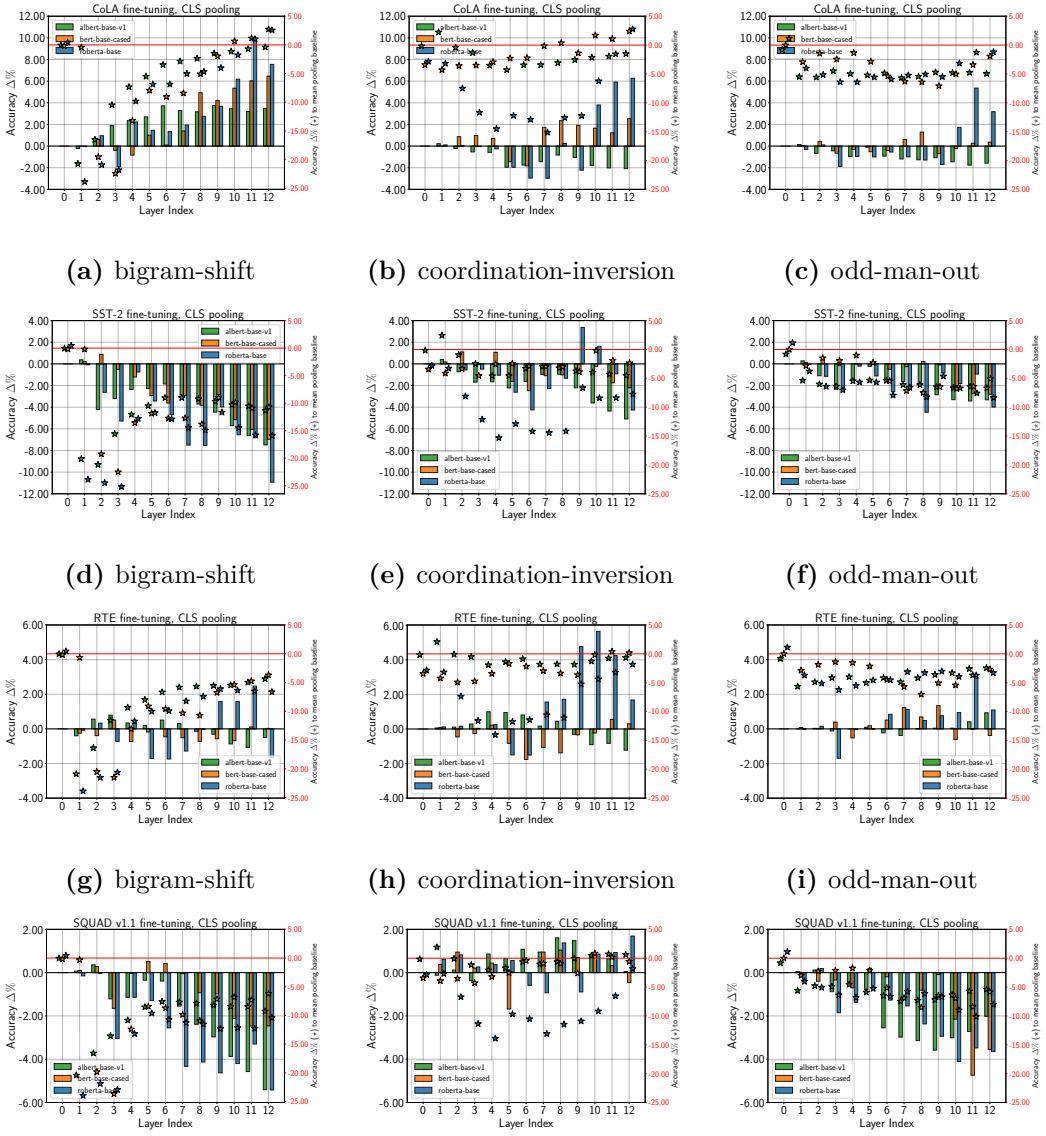
Figure B.1 and Figure B.2 show the change in probing accuracy (Δ in %) across all probing tasks when fine-tuning on CoLA, SST-2, RTE, and SQuAD using CLS-pooling and mean-pooling, respectively. The second y-axis in Figure B.1 shows the layer-wise difference after fine-tuning compared to the mean-pooling baseline. Note that only in very few cases this differences is larger than zero.

Statistics	**Task**			
	CoLA	SST-2	RTE	SQuAD
training	8.6k	67k	2.5	87k
validation	1,043	874	278	10k
metric	MCC	Acc.	Acc.	EM/F_1

Table B.2: Fine-tuning task statistics.

Probing Task	BERT-base-cased							
	CLS-pooling				mean-pooling			
	RTE		SQuAD		RTE		SQuAD	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	−0.21	−0.39	−0.05	−1.50	−0.07	−0.31	−0.54	−1.66
coordinate-inversion	−0.43	−0.36	0.04	0.56	0.05	0.13	−0.03	0.10
odd-man-out	0.09	0.38	−0.21	−1.89	0.09	0.01	−0.28	−1.73
Probing Task	RoBERTa-base							
	CLS-pooling				mean-pooling			
	RTE		SQuAD		RTE		SQuAD	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	−0.51	0.44	−1.17	−4.33	−0.09	−1.32	−0.28	−3.09
coordinate-inversion	−0.35	3.27	0.29	0.50	0.30	−0.48	0.20	0.05
odd-man-out	−0.11	1.22	−0.76	−3.01	−0.04	−1.96	−0.21	−3.58
Probing Task	ALBERT-base-v1							
	CLS-pooling				mean-pooling			
	RTE		SQuAD		RTE		SQuAD	
	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12	0 – 6	7 – 12
bigram-shift	0.29	−0.43	−0.38	−3.46	−0.13	−0.82	−0.60	−3.11
coordinate-inversion	0.46	−0.44	0.32	0.92	0.13	−0.38	0.04	−0.27
odd-man-out	−0.03	0.17	−0.65	−2.91	−0.17	−0.85	−0.55	−3.18

Table B.3: Change in probing accuracy (Δ in %) of **RTE** and **SQuAD** fine-tuned models compared to the pre-trained models when using CLS and mean-pooling. We average the difference in probing accuracy over two different layers groups: layers 0 to 6 and layers 7 to 12.



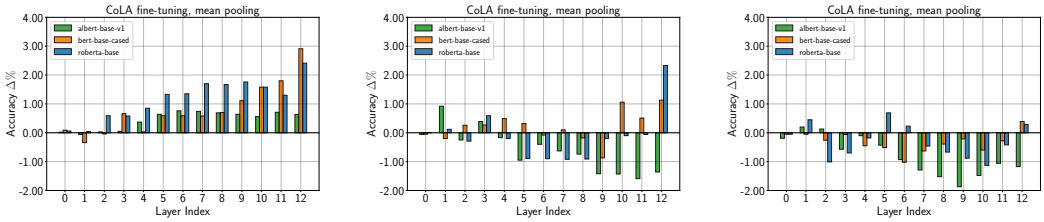
(a) bigram-shift **(b)** coordination-inversion **(c)** odd-man-out

(d) bigram-shift **(e)** coordination-inversion **(f)** odd-man-out

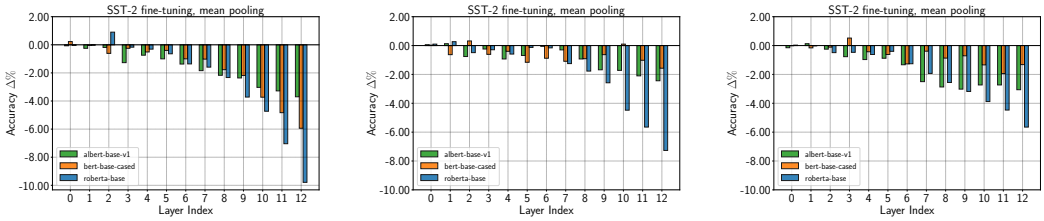
(g) bigram-shift **(h)** coordination-inversion **(i)** odd-man-out

(j) bigram-shift **(k)** coordination-inversion **(l)** odd-man-out

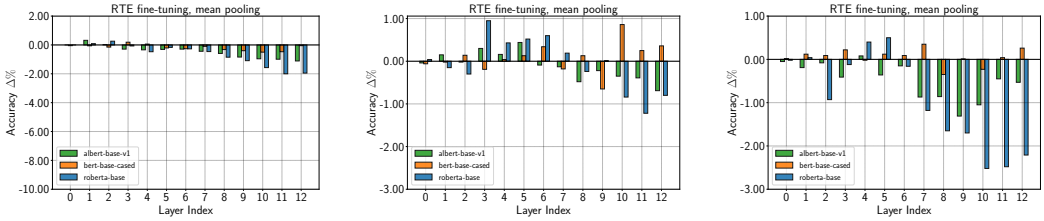
Figure B.1: Difference in probing accuracy (Δ in %) when using CLS-pooling after fine-tuning on **CoLA**, **SST-2**, **RTE**, and **SQuAD** for all three encoder models BERT, RoBERTa, and ALBERT across all probing taks considered in Chapter 4. The second y-axis shows layer-wise improvement over the mean-pooling baselines (stars) on the respective task.



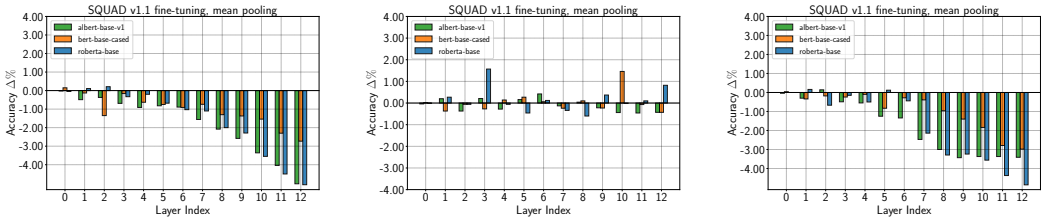
(a) bigram-shift **(b)** coordination-inversion **(c)** odd-man-out



(d) bigram-shift **(e)** coordination-inversion **(f)** odd-man-out



(g) bigram-shift **(h)** coordination-inversion **(i)** odd-man-out



(j) bigram-shift **(k)** coordination-inversion **(l)** odd-man-out

Figure B.2: Difference in probing accuracy (Δ in %) when using mean-pooling after fine-tuning on **CoLA**, **SST-2**, **RTE**, and **SQuAD** for all three encoder models BERT, RoBERTa, and ALBERT across all probing tasks considered in Chapter 4.

C

Investigating Fine-tuning Stability

Contents

C.1 Alternative notions of stability

Here, we elaborate on other possible definitions of fine-tuning stability. The definition that we use throughout Chapter 5 follows previous work (Phang et al., 2018; Dodge et al., 2020a; Lee et al., 2020). For example, while Dodge et al., 2020a do not directly define fine-tuning stability, they report and analyze the *standard deviation* of the validation performance (e.g., see Section 4.1 of their paper). Along the same lines, an earlier work of Phang et al., 2018, which studies the influence of intermediate fine-tuning, discusses the *variance* of the validation performance

(see Section 4: Results, paragraph Fine-Tuning Stability therein) and shows the *standard deviation* over multiple random seeds in Figure 1.

For simplicity, let us assume that the performance metric is *accuracy* and we have two classes. Let A be a randomized fine-tuning algorithm that produces a classifier f_A, and let us denote data points as $(x, y) \sim \mathcal{D}$ where $\mathcal{D}$ is the data-generating distribution. Our definition of fine-tuning stability can be formalized as follows:

$$S_{\text{ours}}(A) = \text{Var}_A \left[\text{E}_{x,y} \left[\mathbb{1}_{f_A(x)=y} \right] \right] = \text{Var}_A \left[\text{Accuracy}(f_A) \right].$$

This definition directly measures the variance of the performance metric and aims to answer the question: *If we perform fine-tuning multiple times, how large will the difference in performance be?*

An alternative definition of fine-tuning stability that could be considered is *per-point stability* where the expectation and variance are interchanged:

$$S_{\text{per-point}}(A) = \text{E}_{x,y} \left[\text{Var}_A \left[\mathbb{1}_{f_A(x)=y} \right] \right].$$

This definition captures a different notion of stability. Namely, it captures stability per data point by measuring how much the classifiers f_A differ on the same point x given label y. Studying the per-point fine-tuning stability can be useful to better understand the properties of fine-tuned models and we refer to R. T. McCoy et al. (2020) for a study in this direction.

C.2 Task statistics

Statistics for each of the datasets studied in Chapter 5 are shown in Table C.1. All datasets are publicly available. The GLUE datasets can be downloaded here: https://github.com/nyu-mll/jiant. SciTail is available at https://github.com/allenai/scitail.

	RTE	**MRPC**	**CoLA**	**QNLI**	**SciTail**
Training	2491	3669	8551	104744	23596
Development	278	409	1043	5464	1304
Majority baseline	0.53	0.75	0.0	0.50	0.63
Metric	Acc.	F1 score	MCC	Acc.	Acc.

Table C.1: Dataset statistics and majority baselines.

C.3 Hyperparameters

Hyperparameters for BERT, RoBERTa, and ALBERT used for all our experiments are shown in Table C.2.

C.4 Ablation studies

Figure C.1 shows the results of fine-tuning on RTE with different combinations of learning rate, number of training epochs, and bias correction. We make the following observations:

- When training for only 3 epochs, disabling bias correction clearly hurts performance.
- With bias correction, training with larger learning rates is possible.
- Combining the usage of bias correction with training for more epochs leads to the best performance.

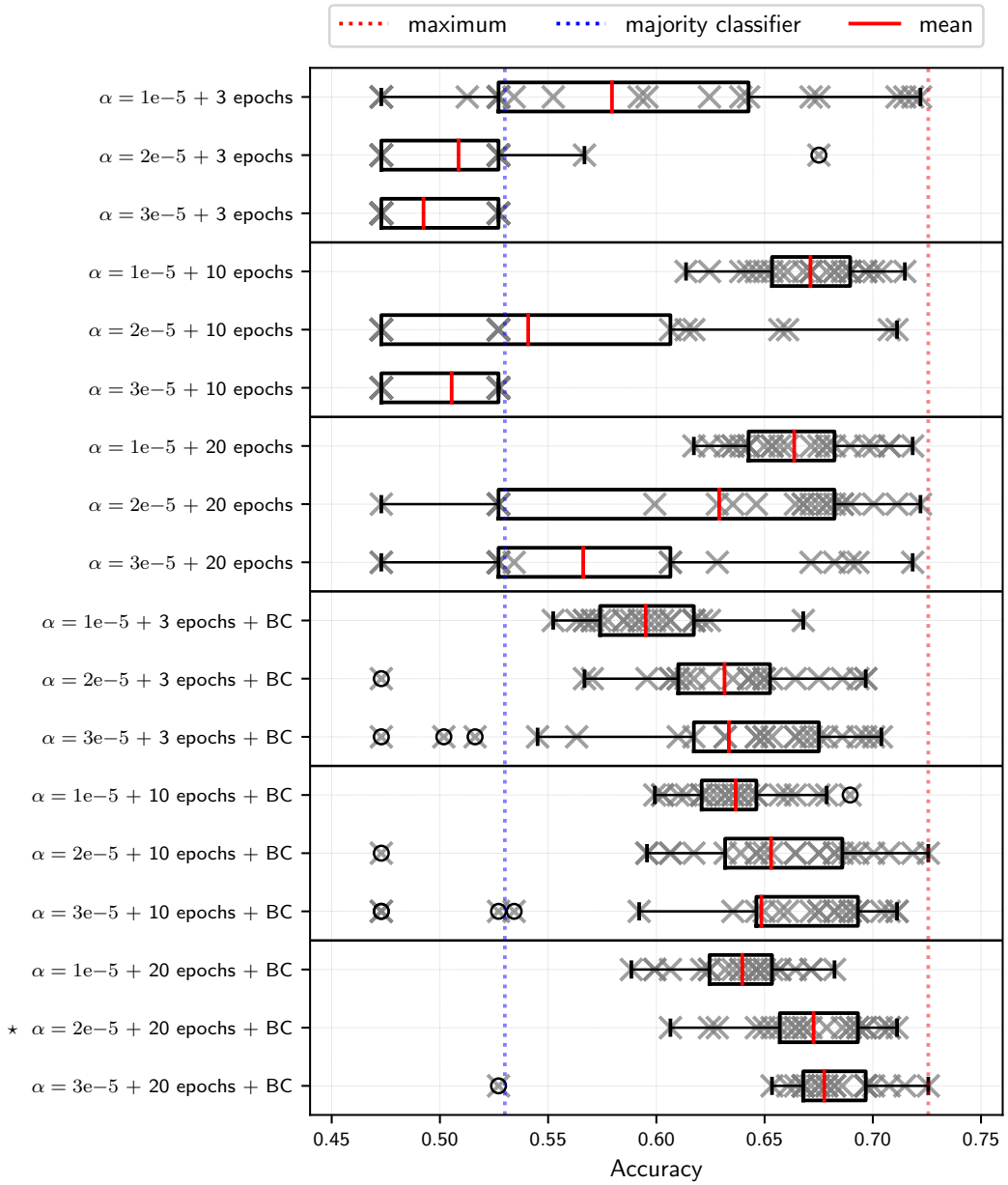


Figure C.1: Full ablation of fine-tuning BERT on RTE. For each setting, we vary only the number of training steps, learning rate, and usage of bias correction (BC). All other hyperparameters are unchanged. We fine-tune 25 models for each setting. ⋆ shows the setting which we recommend as a new baseline fine-tuning strategy.

Hyperparam	BERT	RoBERTa	ALBERT
Epochs	$3, 10, 20$	3	3
Learning rate	1e−5 − 5e−5	1e−5 − 3e−5	1e−5 − 3e−5
Learning rate schedule	warmup-linear	warmup-linear	warmup-linear
Warmup ratio	0.1	0.1	0.1
Batch size	16	16	16
Adam ϵ	1e−6	1e−6	1e−6
Adam β_1	0.9	0.9	0.9
Adam β_2	0.999	0.98	0.999
Adam bias correction	{True, False}	{True, False}	{True, False}
Dropout	0.1	0.1	–
Weight decay	0.01	0.1	–
Clipping gradient norm	1.0	–	1.0
Number of random seeds	25	25	25

Table C.2: Hyperparameters used for fine-tuning.

C.5 Additional gradient norm visualizations

We provide additional visualizations for the vanishing gradients observed when fine-tuning BERT, RoBERTa, and ALBERT in Figures C.2, C.3, and C.4. Note that for ALBERT besides the pooler and classification layers, we plot only the gradient norms of a single hidden layer (referred to as `layer0`) because of weight sharing.

Gradient norms and MLM perplexity We can see from the gradient norm visualizations for BERT in Figures 5.4 and C.2 that the gradient norm of the pooler and classification layer remains large. Hence, even though the gradients on most layers of the model vanish, we still update the weights on the top layers. In fact, this explains the large increase in MLM perplexity for the failed models

which is shown in Figure 5.2a. While most of the layers do not change as we continue training, the top layers of the network change dramatically.

C.6 Loss surfaces

For Figure 5.7, we define the range for both α and β as $[-1.5, 1.5]$ and sample 40 points for each axis. We evaluate the loss on 128 samples from the training dataset of each task using *all* model parameters, including the classification layer. We disabled dropout for generating the surface plots.

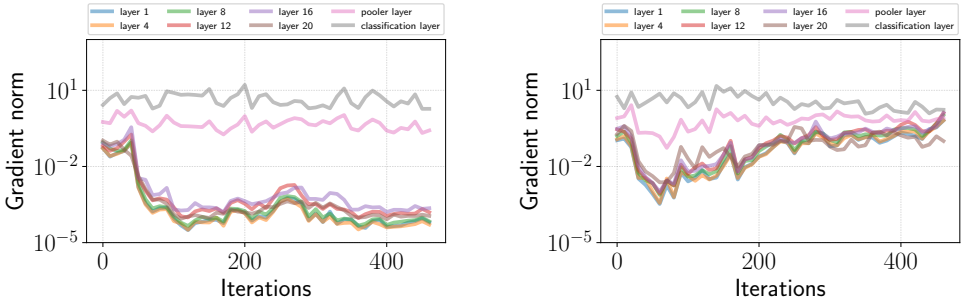
Figure C.5 shows contour plots of the total gradient norm. We can again see that the point to which the failed model converges to ($\boldsymbol{\theta}_f$) is separated from the point the successful model converges to ($\boldsymbol{\theta}_s$) by a barrier. Moreover, on all the three datasets we can clearly see the valley around $\boldsymbol{\theta}_f$ with a small gradient norm.

C.7 Training curves

Figure C.6 shows training curves for 10 successful and 10 failed fine-tuning runs on RTE. We can clearly observe that all 10 failed runs have a common pattern: throughout the training, their training loss stays close to that at initialization. This implies an optimization problem and suggests to reconsider the optimization scheme.

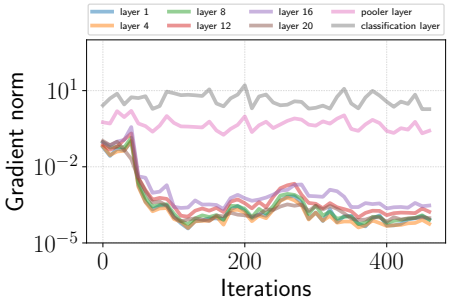
C.8 Additional fine-tuning results

We report additional fine-tuning results on the SciTail dataset (Khot et al., 2018) in Table C.3 to demonstrate that our findings generalize to datasets from other domains.

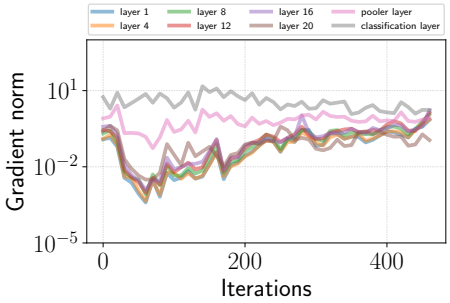


(a) Failed run: attention.key

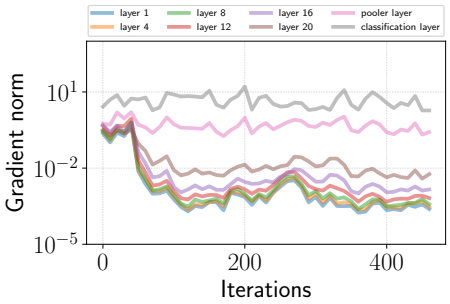
(b) Successful run: attention.key



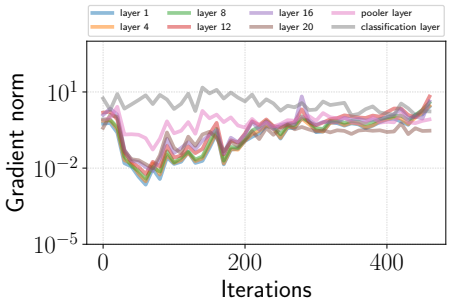
(c) Failed run: attention.query



(d) Successful run: attention.query

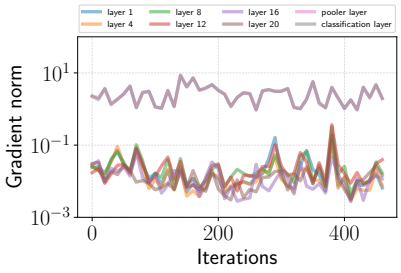


(e) Failed run: attention.value

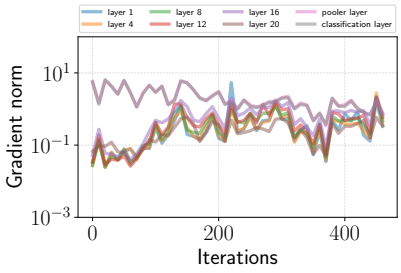


(f) Successful run: attention.value

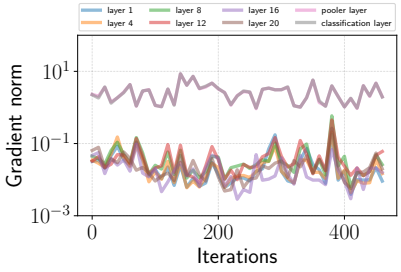
Figure C.2: Gradient norms (plotted on a *logarithmic scale*) of additional weight matrices of **BERT** fine-tuned on RTE. Corresponding layer names are in the captions. We show gradient norms corresponding to a single failed and single successful, respectively.



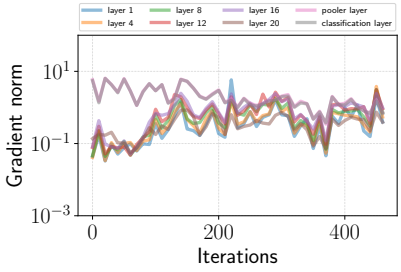
(a) Failed run: attention.key



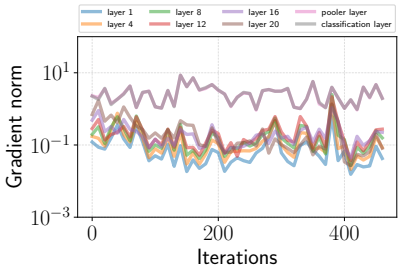
(b) Successful run: attention.key



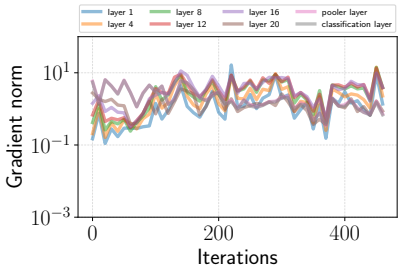
(c) Failed run: attention.query



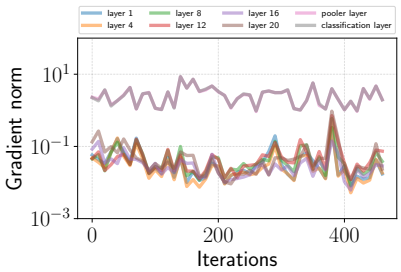
(d) Successful run: attention.query



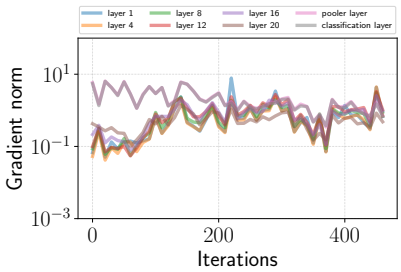
(e) Failed run: attention.value



(f) Successful run: attention.value

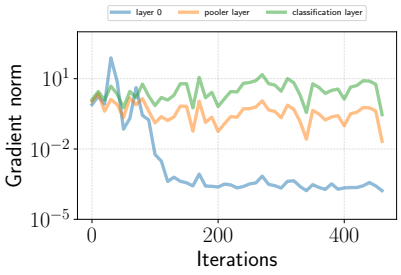


(g) Failed run: attention.output.dense

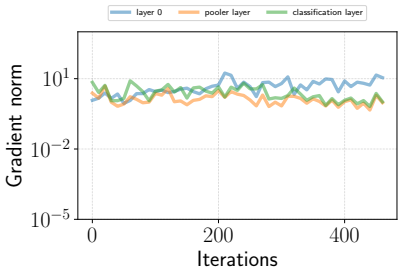


(h) Successful run: attention.output.dense

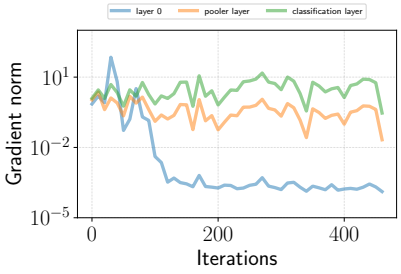
Figure C.3: Gradient norms (plotted on a *logarithmic scale*) of additional weight matrices of **RoBERTa** fine-tuned on RTE. Corresponding layer names are in the captions. We show gradient norms corresponding to a single failed and single successful, respectively.



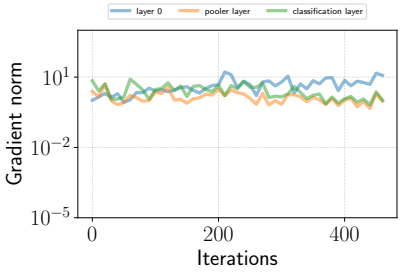
(a) Failed run: attention.key



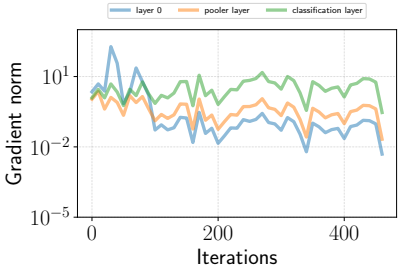
(b) Successful run: attention.key



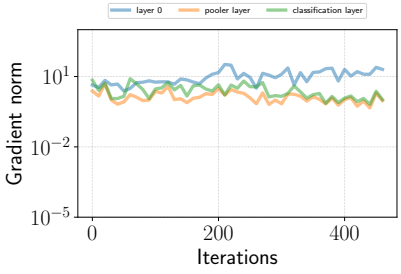
(c) Failed run: attention.query



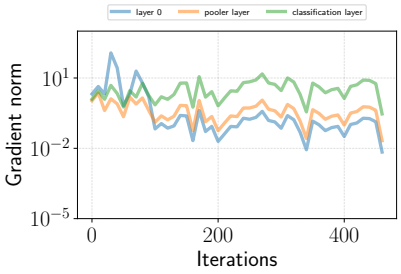
(d) Successful run: attention.query



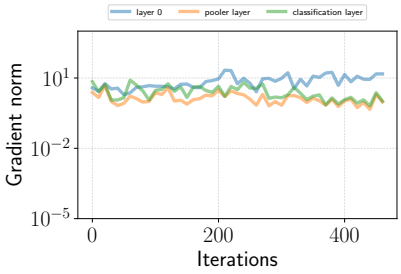
(e) Failed run: attention.value



(f) Successful run: attention.value

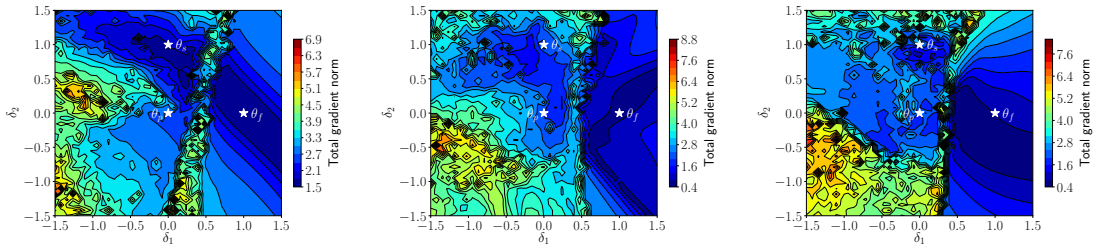


(g) Failed run: attention.dense



(h) Successful run: attention.dense

Figure C.4: Gradient norms (plotted on a *logarithmic scale*) of additional weight matrices of **ALBERT** fine-tuned on RTE. Corresponding layer names are in the captions. We show gradient norms corresponding to a single failed and single successful, respectively.

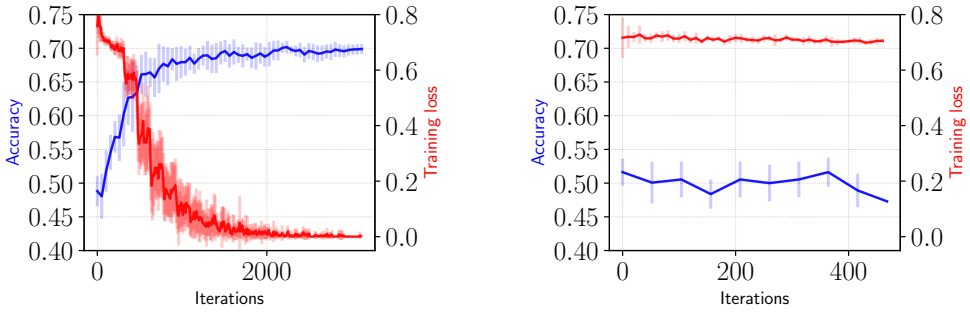


(a) RTE **(b)** MRPC **(c)** CoLA

Figure C.5: 2D gradient norm surfaces in the subspace spanned by $\delta_1 = \boldsymbol{\theta}_f - \boldsymbol{\theta}_p$ and $\delta_2 = \boldsymbol{\theta}_s - \boldsymbol{\theta}_p$ for BERT fine-tuned on RTE, MRPC and CoLA. $\boldsymbol{\theta}_p$, $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_s$ denote the parameters of the pre-trained, failed, and successfully trained model, respectively.

Due to its comparatively large size (28k training samples) fine-tuning on SciTail with the Devlin et al. (2019) scheme is already very stable even when trained for 3 epochs. This is comparable to what we find for QNLI in Section 5.5.1. When applying our fine-tuning scheme to SciTail, the results are very close to that of Devlin et al. (2019). On the other hand, when training on a smaller subset of SciTail (1k training samples) we can clearly see the same results as also observed in Figure 5.3 for MRPC, CoLA, and QNLI, i.e. using more training iterations improves the fine-tuning stability.

We conclude from this experiment that our findings and guidelines generalize to datasets from other domains as well. This gives further evidence that the observed instability is not a property of any particular dataset but rather a result of too few training iterations based on the common fine-tuning practice of using a fixed number of epochs (and not iterations) independently of the training data size.



(a) Successful runs **(b)** Failed runs

Figure C.6: The test accuracy and training loss of (a) 10 successful runs with our fine-tuning scheme and (b) 10 failed runs with fine-tuning scheme Devlin on RTE. Solid line shows the mean, error bars show ±1std.

Approach	**SciTail**		
	std	mean	max
Devlin et al. (2019), full train set, 3 epochs	0.4	95.1	96.0
Devlin et al. (2019), 1k samples, 3 epochs	17.9	69.8	89.4
Devlin et al. (2019), 1k samples, 71 epochs	0.9	87.5	89.1
Ours, full train set	0.6	95.1	96.8

Table C.3: Standard deviation, mean, and maximum performance on the development set of SciTail when fine-tuning BERT over 25 random seeds. Standard deviation: lower is better, i.e. fine-tuning is more stable.

D

Investigating the Generalization of Task-adapted Models

Contents

D.1 Experimental details

We access all models via huggingface transformers (Wolf et al., 2020) and use its DeepSpeed (Rasley et al., 2020) integration for efficient distributed training and evaluation.

D.1.1 Hardware

We run our experiments on 8x A100 GPUs with 80GB of memory.

D.1.2 Label distribution

Table D.1 shows the accuracy of the majority class label on each of the datasets. Note that MNLI (when merging the neutral and contradiction classes) and PAWS-QQP are highly unbalanced.

Dataset	**Majority class**
MNLI (remove neutral)	0.512
MNLI (merge neutral and contradiction)	0.645
RTE	0.527
QQP	0.632
HANS	0.500
PAWS-QQP	0.718

Table D.1: Accuracy of the majority class label for each dataset.

D.1.3 In-context learning: Additional details

Patterns We present the patterns used for in-context learning in Table D.2. We obtain the `GPT-3` pattern from Brown et al. (2020). The `eval-harness` pattern is based on L. Gao et al. (2021).

Dataset(s)	Pattern	Text	Answer prefix	Target tokens
MNLI, RTE	`minimal`	{premise} {hypothesis} ?	–	Yes, No
MNLI, RTE	`gpt-3`	{premise} question: {hypothesis} Yes or No?	answer:	Yes, No
MNLI, RTE	`eval-harness`	{premise} \n Question: {hypothesis} True or False?	\n Answer:	True, False
QQP	`minimal`	{question 1} {question 2} ?	–	Yes, No
QQP	`eval-harness`	Question 1: {question 1} \n Question 2:{question 2} \n Question: Do both questions ask the same thing?	Answer:	Yes, No

Table D.2: Patterns used for in-context learning. The minimal patterns are used for pattern-based fine-tuning as well.

D.1.4 In-context learning: Comparison with previous work

Table D.3 compares our in-context learning results to results from previous work. On RTE and MNLI we achieve comparable performance to previous work. On QQP, our in-context learning results are much worse (and very close to the majority class classifier). We hypothesize that this is due to the difference in model size (GPT-3 with 175B parameters vs. OPT with 30B parameters) and hence focus on MNLI and RTE for most of our experiments.

D.1.5 Fine-tuning: Additional details

Hyperparameters Table D.4 provides an overview of all hyperparameters used during fine-tuning.

Model	Dataset	In-domain	Out-of-domain
GPT-3 175B	MNLI	77.6	75.3
OPT 30B	RTE	62.0	–
GPT-3 175B	QQP	83.5	73.7
OPT 30B	MNLI	71.4 (74.9)	56.7 (72.3)
OPT 30B	RTE	61.7 (66.8)	60.5 (65.4)
OPT 30B	QQP	42.0 (63.1)	49.8 (53.3)

Table D.3: Comparing in-context learning results from previous work (first three rows) with ours (last three rows). In our results we report average and maximum performance (in parentheses) of the largest model. Previous results are from Si et al. (2023) for GPT-3 and S. Zhang et al. (2022) for OPT.

D.2 Additional results for OPT models

D.2.1 Significance tests

Tables 6.1 and D.5 to D.7 show the results of a Welch's t-test comparing the average in-domain and out-of-domain performance of fine-tuning and pattern-based fine-tuning on RTE and MNLI. We use 16 samples and 10 different seeds for each approach and consider a p-value of 0.05 to be statistically significant. For fine-tuning, we compare two different approaches of model selection: (1) based on in-domain performance and (2) based on out-of-domain performance (note that these are the same models as those shown in the first row of Figure 6.1).

For RTE, our results show that in-context learning outperforms fine-tuning only when comparing large models to smaller models. However, when comparing models of the same size, fine-tuning performs at least equally well to in-context learning, and in some cases even significantly better. For MNLI, for larger models (6.7B onwards) in-context learning outperforms fine-tuning in terms of in-domain performance. Looking at out-of-domain generalization performance, however, we

Hyperparameter	**Value**
Optimizer	AdamW
Learning rate	10^{-5}
Learning rate schedule	linear warmup then constant
Warmup ratio	10% of total steps
Weight decay	0.0
Dropout	0.1
Batch size	32
Epochs	40
Total steps	$\frac{\#\text{samples}}{\text{batch size}} * \text{epochs}$

Table D.4: fine-tuning hyperparameters.

again see that in-context learning only outperforms fine-tuning when comparing large models to much smaller models.

D.2.2 In-context learning

Figures D.1, D.2, and D.3 show in-context learning results on MNLI, RTE, and QQP for all OPT model sizes grouped by number of demonstrations and patterns.

Sensitivity to pattern choice and number of examples On MNLI and RTE, we find that only the largest model benefits from the instructive `gpt-3` and `eval-harness` patterns. Moreover, on all datasets and for all patterns, models are sensitive to the number of demonstrations and do not necessarily improve with more demonstrations.

D.2.3 Fine-tuning

We provide all fine-tuning results in Figures D.4, D.5, and D.6. When comparing results across rows, we see the impact of the number of training examples on the

ICL \ FT	125M	350M	1.3B	2.7B	6.7B	13B	30B
125M	0.03	0.04	0.08	0.11	0.10	0.09	0.10
350M	0.03	0.05	0.08	0.11	0.10	0.10	0.10
1.3B	0.03	0.04	0.08	0.10	0.10	0.09	0.09
2.7B	0.00	0.02	0.05	0.08	0.07	0.07	0.07
6.7B	−0.06	−0.04	−0.01	0.02	0.01	0.01	0.01
13B	−0.06	−0.05	−0.01	0.02	0.01	0.00	0.01
30B	−0.06	−0.04	−0.01	0.02	0.01	0.01	0.01

(a) RTE

ICL \ FT	125M	350M	1.3B	2.7B	6.7B	13B	30B
125M	0.07	0.09	0.13	0.14	0.12	0.17	0.13
350M	0.05	0.07	0.11	0.13	0.11	0.16	0.11
1.3B	−0.02	−0.00	0.03	0.05	0.03	0.08	0.03
2.7B	0.01	0.03	0.07	0.08	0.06	0.11	0.06
6.7B	−0.06	−0.04	−0.00	0.01	−0.01	0.04	−0.00
13B	−0.13	−0.11	−0.07	−0.06	−0.08	−0.03	−0.08
30B	−0.11	−0.09	−0.05	−0.04	−0.06	−0.01	−0.06

(b) MNLI

Table D.5: Difference between average **in-domain performance** of in-context learning and fine-tuning on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

		FT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	−0.01	0.02	0.05	0.05	0.07	0.07	0.07
	350M	−0.01	0.02	0.05	0.05	0.08	0.08	0.07
	1.3B	−0.01	0.01	0.04	0.04	0.07	0.07	0.06
	2.7B	−0.04	−0.01	0.02	0.02	0.05	0.05	0.04
	6.7B	−0.09	−0.07	−0.04	−0.04	−0.01	−0.01	−0.02
	13B	−0.10	−0.07	−0.04	−0.04	−0.02	−0.02	−0.02
	30B	−0.10	−0.07	−0.04	−0.04	−0.01	−0.01	−0.02

(a) RTE

		FT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	0.03	0.05	0.09	0.10	0.07	0.13	0.08
	350M	0.01	0.03	0.07	0.09	0.05	0.12	0.06
	1.3B	−0.07	−0.04	−0.01	0.01	−0.02	0.04	−0.01
	2.7B	−0.03	−0.01	0.02	0.04	0.01	0.07	0.02
	6.7B	−0.10	−0.08	−0.04	−0.03	−0.06	0.00	−0.05
	13B	−0.17	−0.15	−0.11	−0.10	−0.13	−0.07	−0.12
	30B	−0.16	−0.13	−0.10	−0.08	−0.11	−0.05	−0.10

(b) MNLI

Table D.6: Difference between average **in-domain performance** of ICL and FT on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

		FT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	0.01	0.02	0.03	0.11	0.16	0.18	0.16
	350M	0.01	0.02	0.03	0.11	0.16	0.18	0.16
	1.3B	0.01	0.02	0.03	0.11	0.16	0.18	0.16
	2.7B	0.00	0.02	0.03	0.11	0.15	0.18	0.16
	6.7B	0.01	0.02	0.03	0.11	0.15	0.18	0.16
	13B	−0.03	−0.01	0.00	0.08	0.12	0.14	0.13
	30B	−0.10	−0.08	−0.07	0.01	0.05	0.07	0.06

(a) RTE

		FT						
		125M	**350M**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**
ICL	**125M**	0.00	0.01	0.05	0.04	0.13	0.14	0.17
	350M	0.00	0.01	0.05	0.04	0.13	0.14	0.17
	1.3B	0.00	0.01	0.05	0.04	0.13	0.14	0.16
	2.7B	0.00	0.01	0.05	0.04	0.13	0.14	0.16
	6.7B	−0.01	−0.00	0.04	0.03	0.12	0.13	0.16
	13B	−0.03	−0.02	0.02	0.01	0.10	0.11	0.13
	30B	−0.06	−0.06	−0.01	−0.02	0.06	0.08	0.10

(b) MNLI

Table D.7: Difference between average **out-of-domain performance** of in-context learning and fine-tuning on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

results. Comparing results across columns demonstrates the importance of model selection for in-domain and out-of-domain performance.

Figures D.7 and D.8 show a comparison between two different ways of binarizing MNLI. For our main experiments, we remove the neutral class entirely. Merging it with the contradiction class instead leads to an even better relationship between in-domain and out-of-domain generalization performance.

D.3 Additional results for Pythia models

Figure D.9 compares fine-tuning and in-context learning of Pythia models ranging from 410M to 12B parameters (Biderman et al., 2023). Similar to OPT, the Pythia models differ only in their size and have all been trained on exactly the same data (even in the exact same order). We focus on RTE and report results using 16 examples. For in-context learning, we use three different patterns (`minimal`, `gpt-3`, `eval-harness`). For fine-tuning, we report results using 16 and 128 examples and three different model selection strategies (best in-domain, last checkpoint, best out-of-domain). Significance tests are provided in Tables 6.2 and D.8 to D.10.

For in-context learning, all models perform poorly when using the `minimal` pattern. With the `gpt-3` pattern, we can observe a clear impact of model size on in-domain and out-of-domain performance. On the other hand, with the `eval-harness` pattern, for Pythia models, only in-domain performance improves with model size.

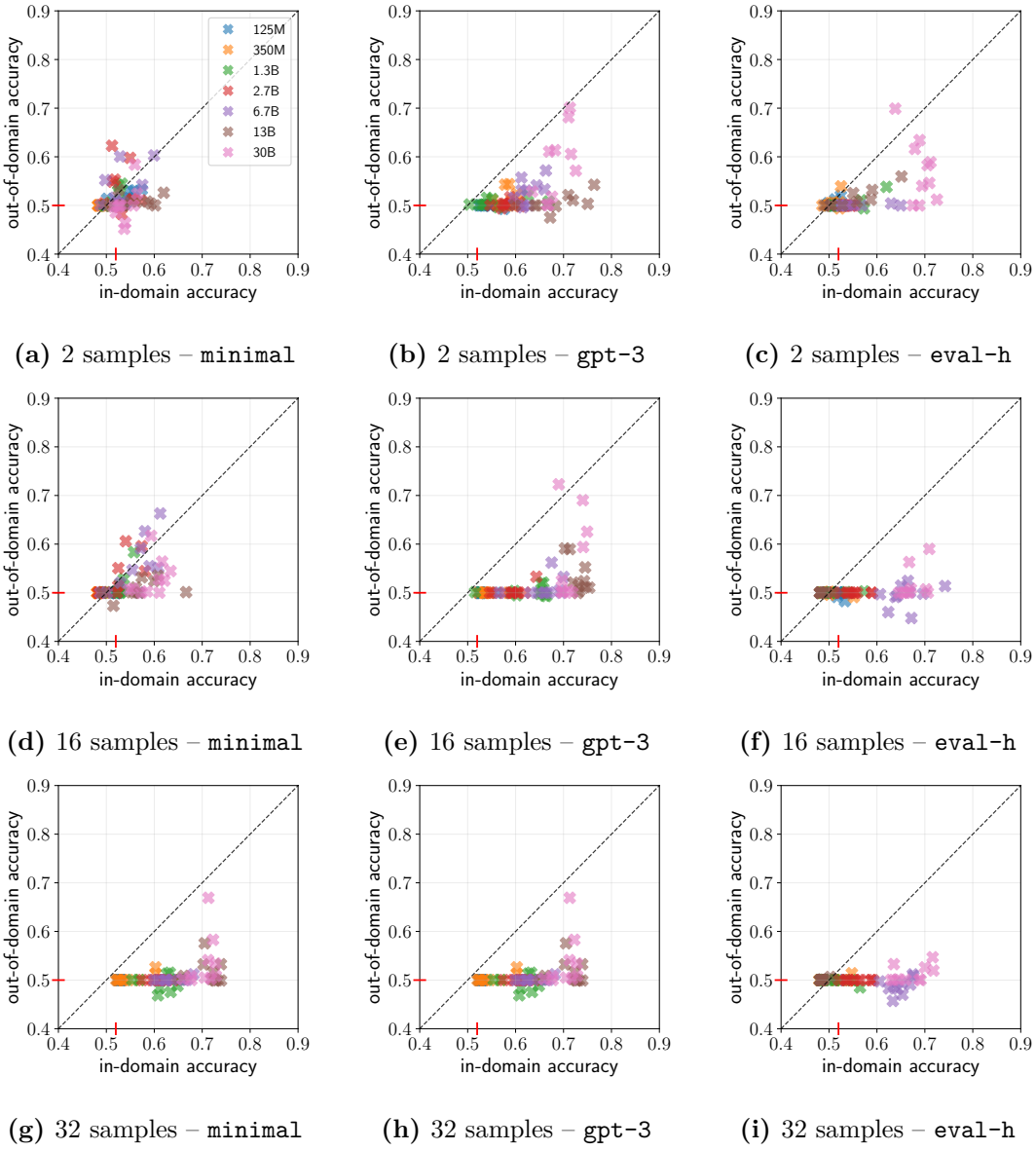
For fine-tuning, when using 16 samples and selecting checkpoints according to out-of-domain performance, almost all checkpoints lead to better out-of-domain than in-domain performance. Moreover, almost all fine-tuned models perform significantly better OOD than models adapted via in-context learning. When fine-tuning with 128 examples, we can see a very clear effect of model size on both in-domain and out-of-domain performance. In particular, when selecting checkpoints according to out-of-domain performance, almost all models perform better out-of-domain than in-domain.

		FT				
		410M	**1.4B**	**2.8B**	**6.9B**	**12B**
ICL	**410M**	0.05	0.06	0.06	0.09	0.07
	1.4B	0.03	0.04	0.04	0.07	0.05
	2.8B	−0.02	−0.00	−0.01	0.02	0.01
	6.9B	−0.03	−0.02	−0.02	0.01	−0.01
	12B	−0.04	−0.03	−0.03	−0.00	−0.02

Table D.8: Difference between average **in-domain performance** of in-context learning and fine-tuning with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For fine-tuning, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: in-context learning performs significantly better than fine-tuning, fine-tuning performs significantly better than in-context learning. For cells without color, there is no significant difference between in-context learning and fine-tuning.

D.4 Analyzing individual OPT fine-tuning runs

Looking at the in-domain and out-of-domain performance for individual checkpoints does not reveal the generalization behavior of individual FT runs during training. In particular, this view does not tell us how stable the generalization of individual runs is during FT. Therefore, in Figures D.10 and D.11 we visualize both in-domain and out-of-domain performance throughout FT on MNLI and RTE when using 128 examples. We observe that out-of-domain performance varies considerably across seeds and even during fine-tuning.

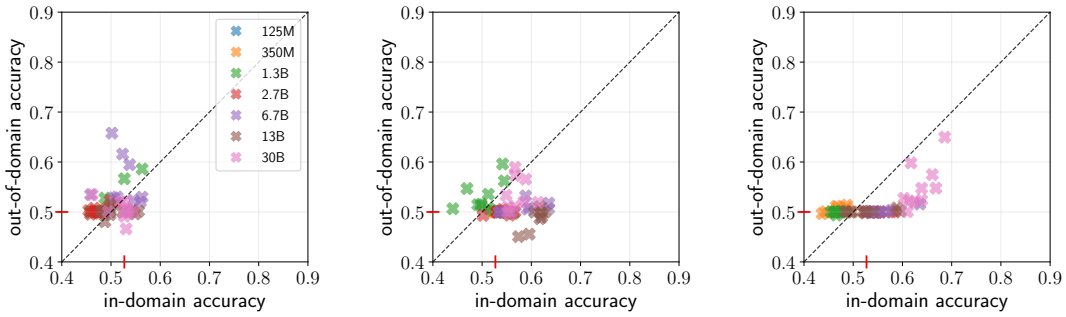


(a) 2 samples – `minimal` **(b)** 2 samples – `gpt-3` **(c)** 2 samples – `eval-h`

(d) 16 samples – `minimal` **(e)** 16 samples – `gpt-3` **(f)** 16 samples – `eval-h`

(g) 32 samples – `minimal` **(h)** 32 samples – `gpt-3` **(i)** 32 samples – `eval-h`

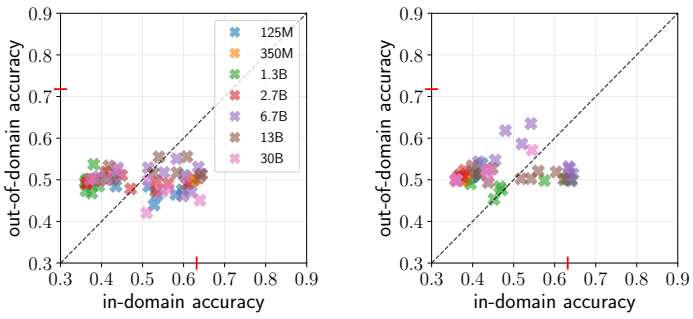
Figure D.1: Relationship between in-domain and out-of-domain performance of in-context learning on MNLI for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



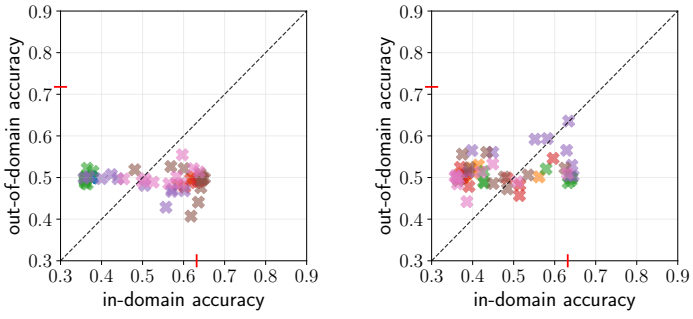
(a) 2 samples – `minimal` **(b)** 2 samples – `gpt-3` **(c)** 2 samples – `eval-h`

(d) 16 samples – `minimal` **(e)** 16 samples – `gpt-3` **(f)** 16 samples – `eval-h`

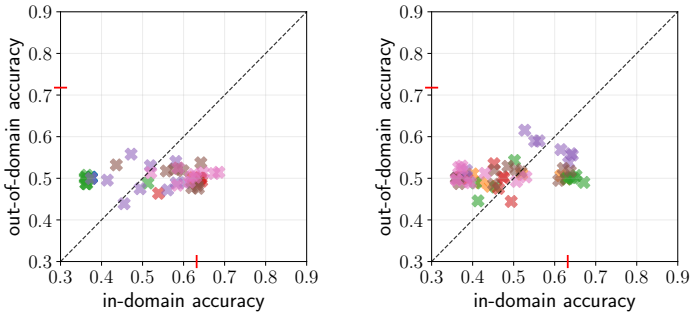
Figure D.2: Relationship between in-domain and out-of-domain performance of in-context learning on RTE for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



(a) 2 samples – `minimal` **(b)** 2 samples – `eval-h`

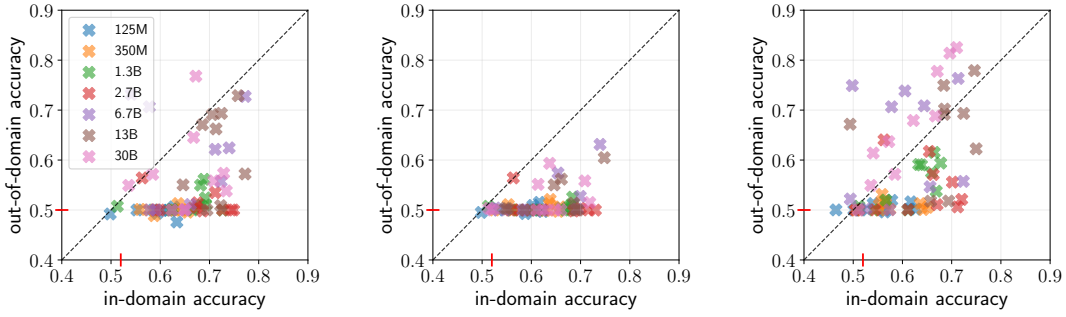


(c) 16 samples – `minimal` **(d)** 16 samples – `eval-h`

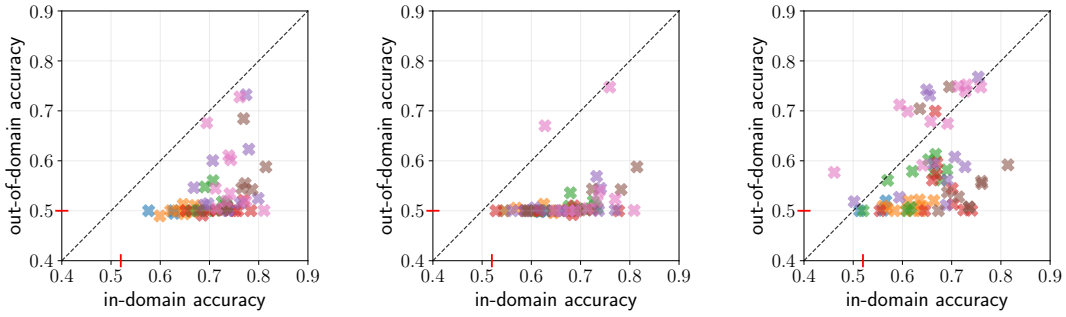


(e) 32 samples – `minimal` **(f)** 32 samples – `eval-h`

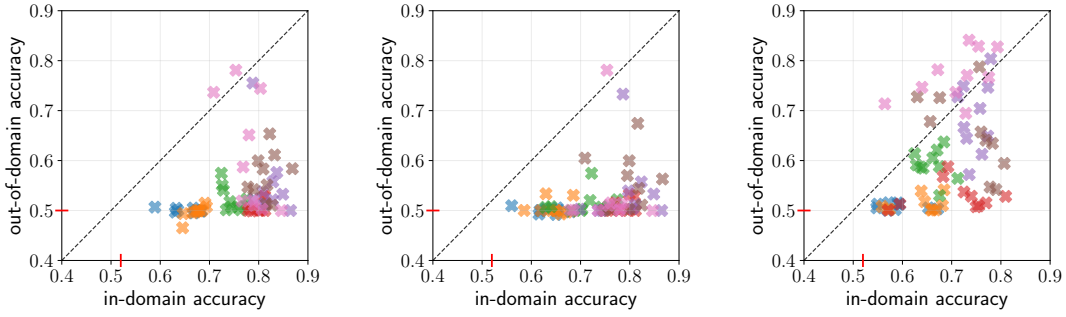
Figure D.3: Relationship between in-domain and out-of-domain performance of in-context learning on QQP for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



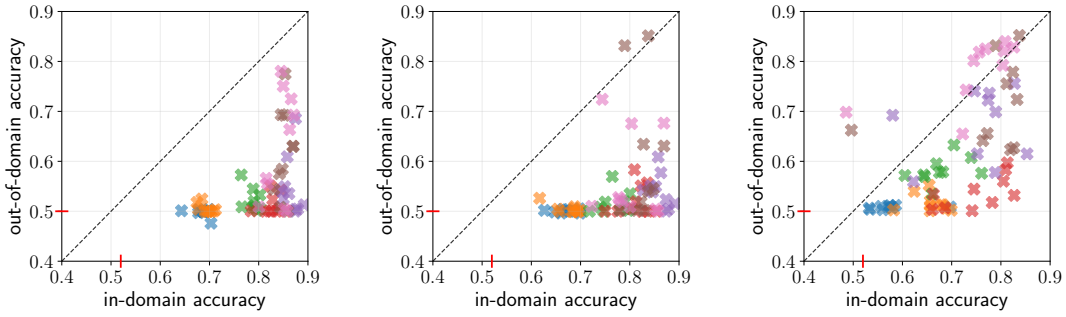
(a) 16 – in-domain **(b)** 16 – last checkpoint **(c)** 16 – out-of-domain



(d) 32 – in-domain **(e)** 32 – last checkpoint **(f)** 32 – out-of-domain

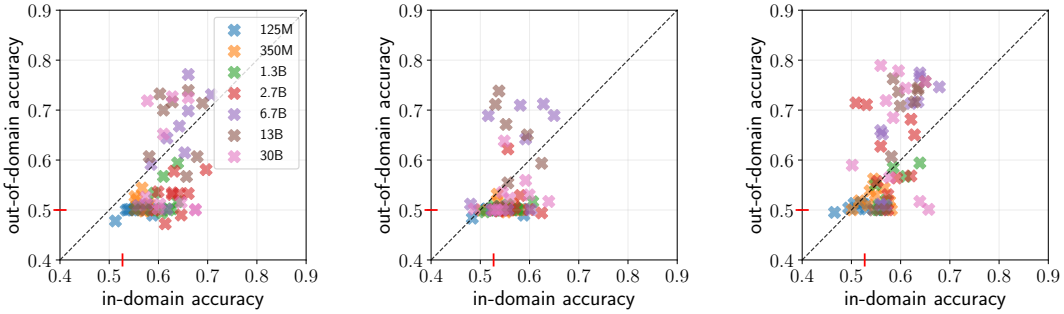


(g) 64 – in-domain **(h)** 64 – last checkpoint **(i)** 64 – out-of-domain

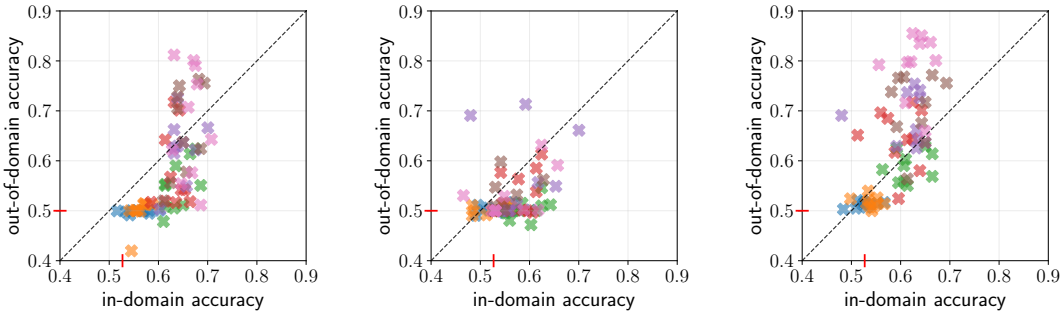


(j) 128 – in-domain **(k)** 128 – last checkpoint **(l)** 128 – out-of-domain

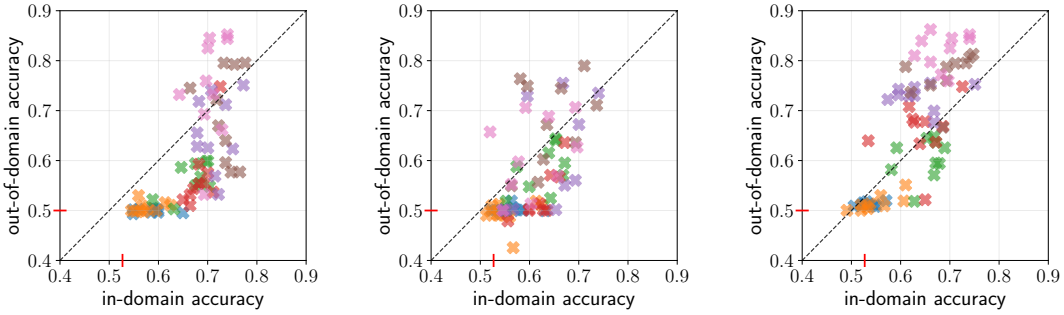
Figure D.4: Relationship between in-domain and out-of-domain performance of pattern-based fine-tuning on MNLI for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



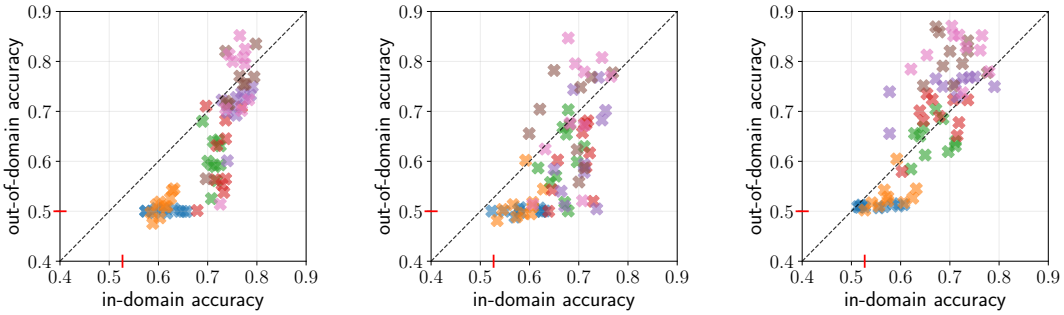
(a) 16 – in-domain **(b)** 16 – last checkpoint **(c)** 16 – out-of-domain



(d) 32 – in-domain **(e)** 32 – last checkpoint **(f)** 32 – out-of-domain

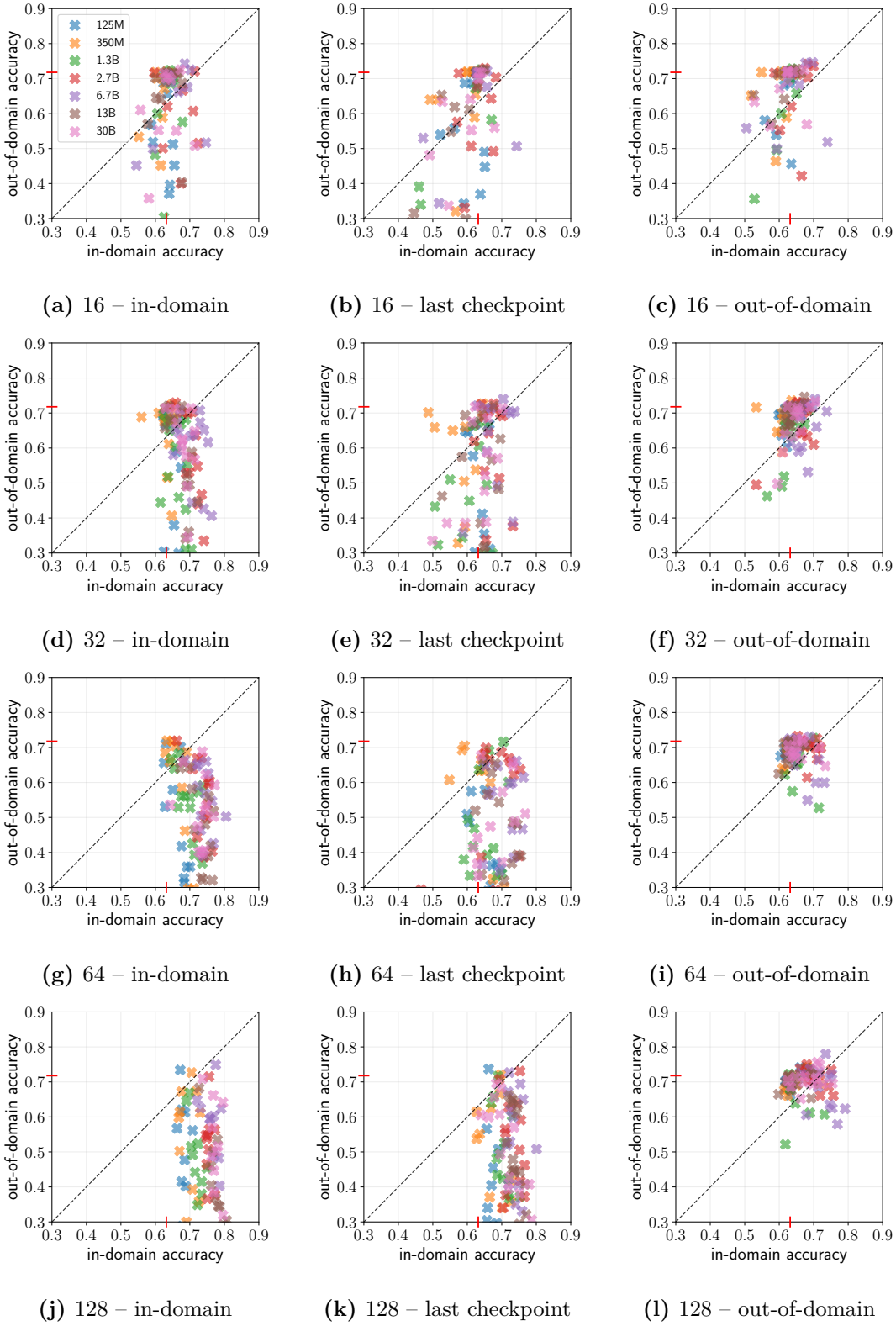


(g) 64 – in-domain **(h)** 64 – last checkpoint **(i)** 64 – out-of-domain



(j) 128 – in-domain **(k)** 128 – last checkpoint **(l)** 128 – out-of-domain

Figure D.5: Relationship between in-domain and out-of-domain performance of pattern-based fine-tuning on RTE for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



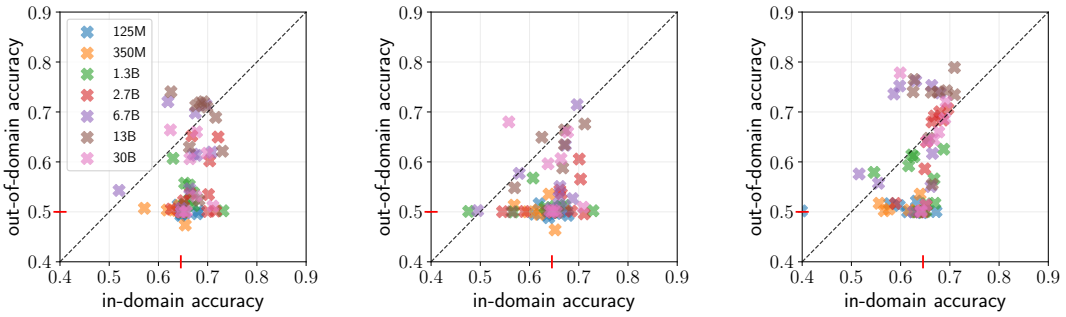
(a) 16 – in-domain **(b)** 16 – last checkpoint **(c)** 16 – out-of-domain

(d) 32 – in-domain **(e)** 32 – last checkpoint **(f)** 32 – out-of-domain

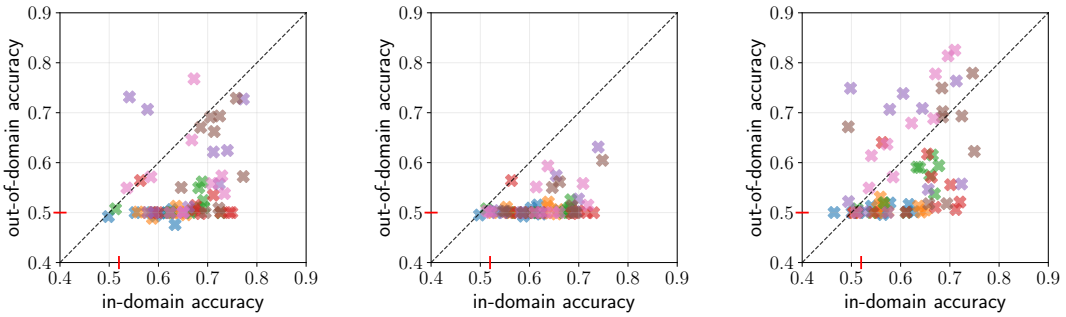
(g) 64 – in-domain **(h)** 64 – last checkpoint **(i)** 64 – out-of-domain

(j) 128 – in-domain **(k)** 128 – last checkpoint **(l)** 128 – out-of-domain

Figure D.6: Relationship between in-domain and out-of-domain performance of pattern-based fine-tuning on QQP for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.

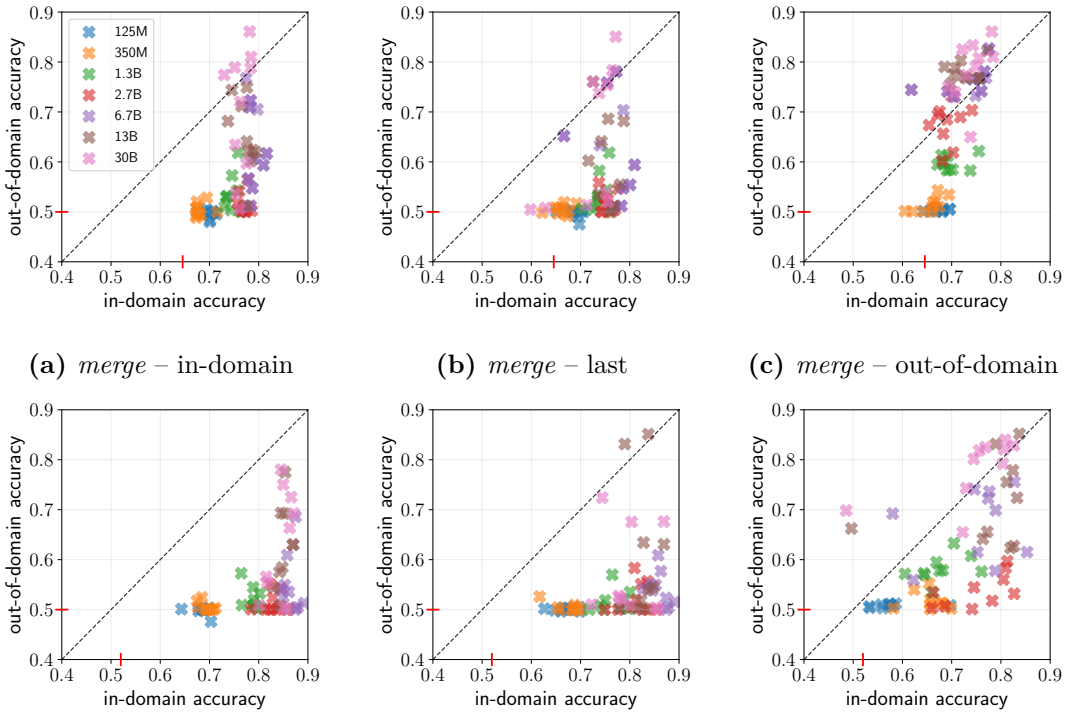


(a) *merge* – in-domain **(b)** *merge* – last **(c)** *merge* – out-of-domain



(d) *remove* – in-domain **(e)** *remove* – last **(f)** *remove* – out-of-domain

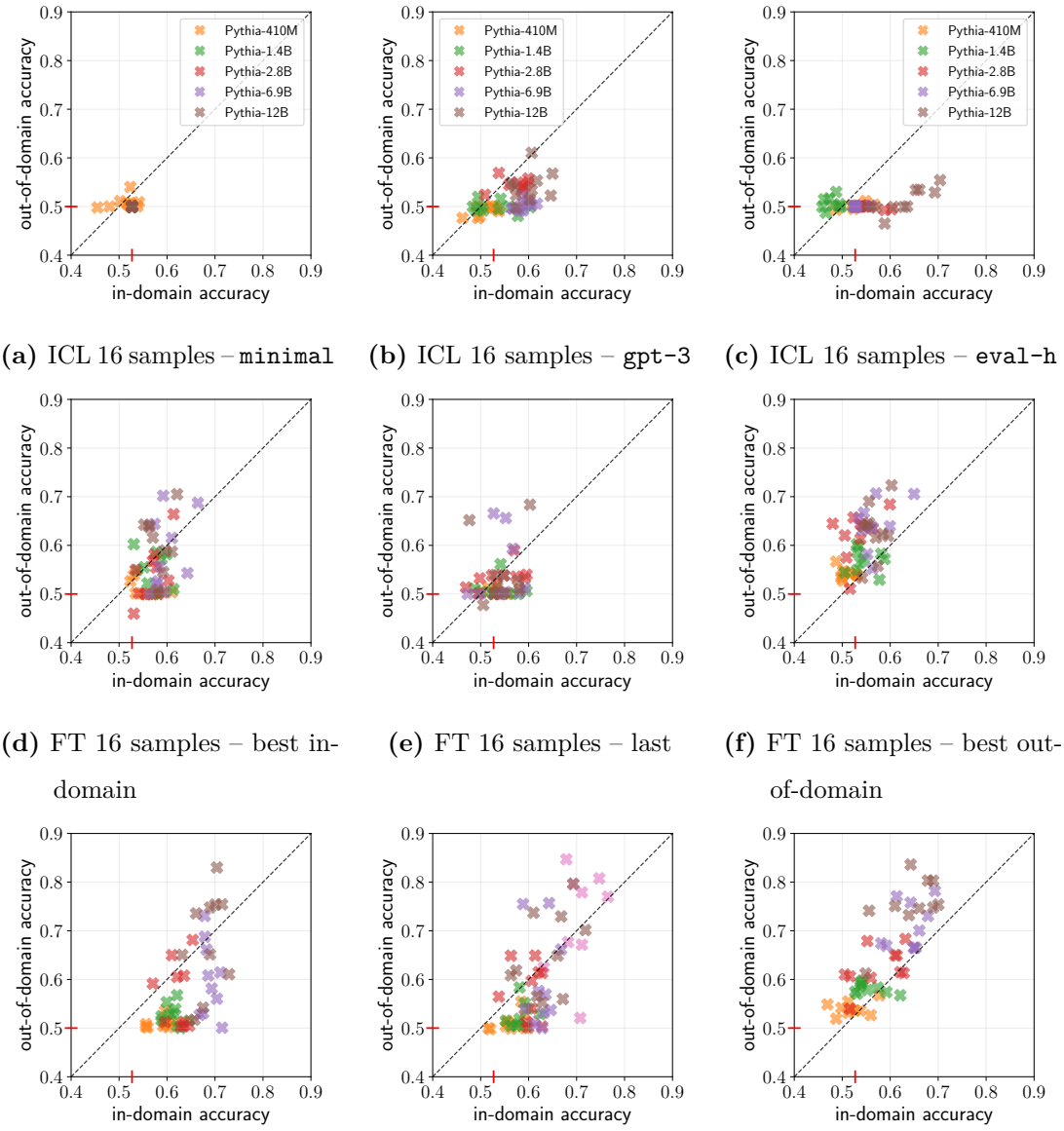
Figure D.7: Relationship between in-domain and out-of-domain performance of pattern-based fine-tuning on MNLI for OPT models of various sizes when **merging** the neutral and contradiction classes **vs. removing** the neutral examples altogether. We fine-tune on **16 examples** using 10 different seeds. — in the x- and y-axis indicates the performance of the majority class label.



(a) *merge* – in-domain **(b)** *merge* – last **(c)** *merge* – out-of-domain

(d) *remove* – in-domain **(e)** *remove* – last **(f)** *remove* – out-of-domain

Figure D.8: Relationship between in-domain and out-of-domain performance of pattern-based fine-tuning on MNLI for OPT models of various sizes when **merging** the neutral and contradiction classes **vs. removing** the neutral examples altogether. We fine-tune on **128 examples** using 10 different seeds. — in the x- and y-axis indicates the performance of the majority class label.



(a) ICL 16 samples – `minimal` **(b)** ICL 16 samples – `gpt-3` **(c)** ICL 16 samples – `eval-h`

(d) FT 16 samples – best in-domain **(e)** FT 16 samples – last **(f)** FT 16 samples – best out-of-domain

(g) FT 128 samples – best in-domain **(h)** FT 128 samples – last **(i)** FT 128 samples – best out-of-domain

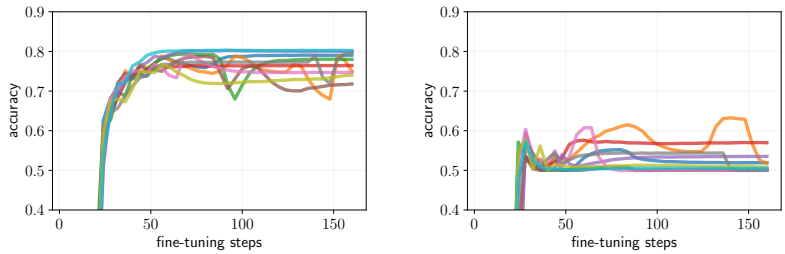
Figure D.9: ICL and FT results for Pythia models of different size. For in-context learning, we report results using 16 examples and three different patterns (`minimal`, `gpt-3`, `eval-harness`). For FT, we report results using 16 and 128 examples using three different model selection strategies (best in-domain, last checkpoint, best out-of-domain). In all cases, we show results for 10 different random seeds. — in the x- and y-axis indicates the performance of the majority class label.

		FT				
		410M	**1.4B**	**2.8B**	**6.9B**	**12B**
ICL	**410M**	−0.00	0.04	0.02	0.06	0.06
	1.4B	−0.02	0.02	0.00	0.04	0.04
	2.8B	−0.06	−0.03	−0.04	−0.01	−0.01
	6.9B	−0.08	−0.04	−0.06	−0.02	−0.02
	12B	−0.09	−0.05	−0.07	−0.03	−0.03

Table D.9: Difference between average **in-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

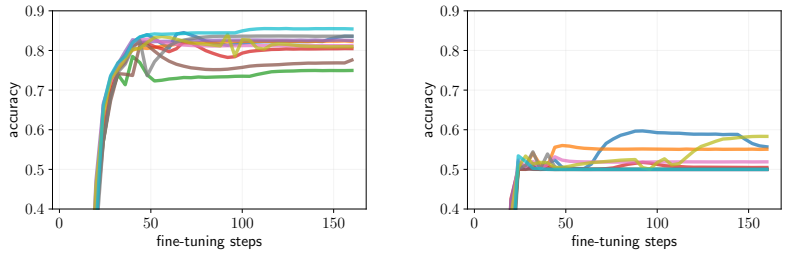
		FT				
		410M	**1.4B**	**2.8B**	**6.9B**	**12B**
ICL	**410M**	0.05	0.08	0.13	0.15	0.14
	1.4B	0.04	0.07	0.12	0.14	0.13
	2.8B	−0.00	0.03	0.08	0.10	0.09
	6.9B	0.04	0.07	0.12	0.14	0.13
	12B	0.00	0.03	0.08	0.10	0.09

Table D.10: Difference between average **out-of-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.



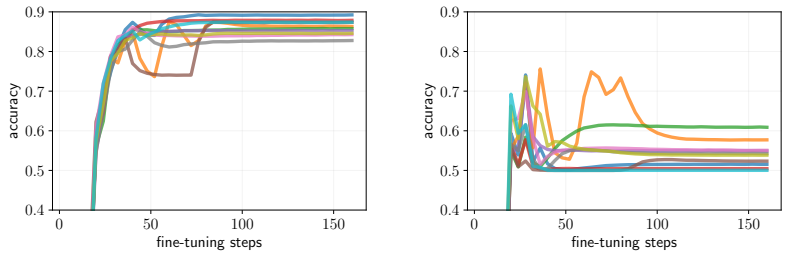
(a) 1.3B – in-domain

(b) 1.3B – out-of-domain



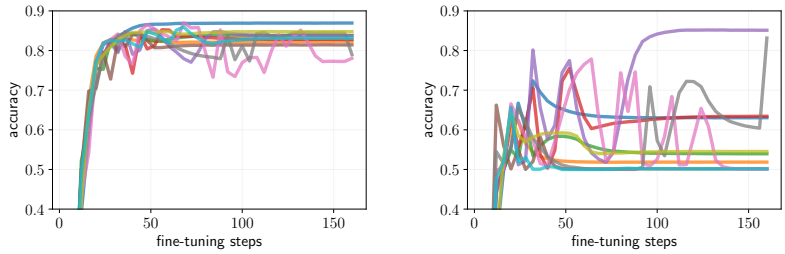
(c) 2.7B – in-domain

(d) 2.7B – out-of-domain



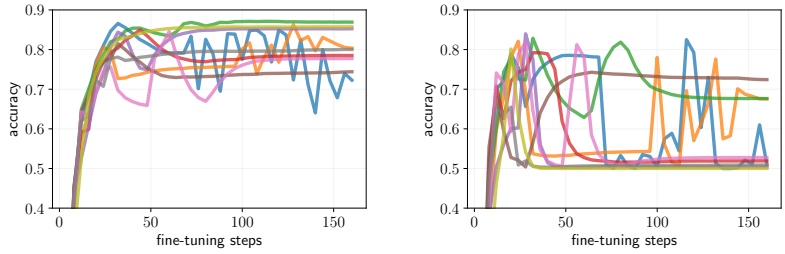
(e) 6.7B – in-domain

(f) 6.7B – out-of-domain



(g) 13B – in-domain

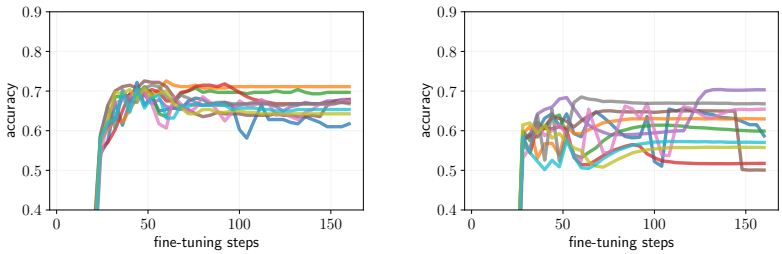
(h) 13B – out-of-domain



(i) 30B – in-domain

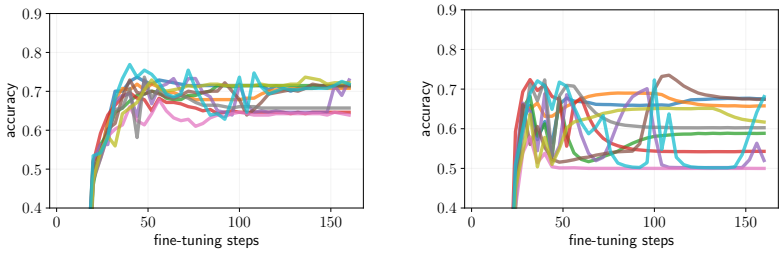
(j) 30B – out-of-domain

Figure D.10: Generalization throughout PBFT on MNLI for OPT models of various sizes. We train on 128 examples. Colors denote different data seeds. First column shows in-domain, second column out-of-domain performance.



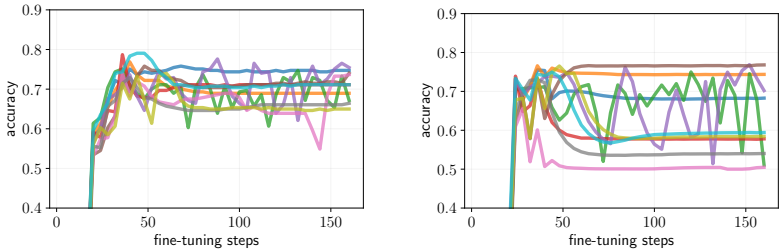
(a) 1.3B – in-domain

(b) 1.3B – out-of-domain



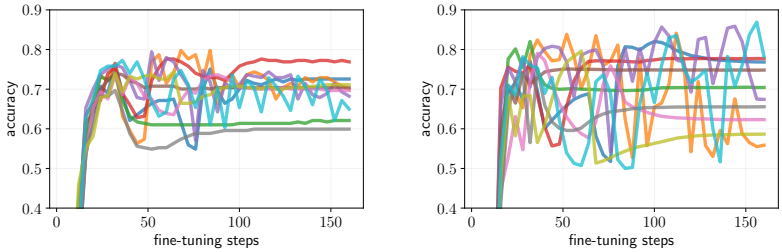
(c) 2.7B – in-domain

(d) 2.7B – out-of-domain



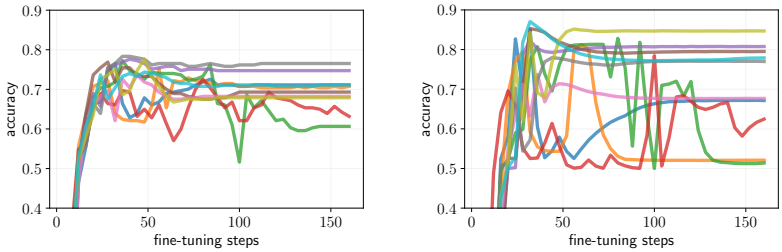
(e) 6.7B – in-domain

(f) 6.7B – out-of-domain



(g) 13B – in-domain

(h) 13B – out-of-domain



(i) 30B – in-domain

(j) 30B – out-of-domain

Figure D.11: Generalization throughout PBFT on RTE for OPT models of various sizes. We train on 128 examples. Colors denote different data seeds. First column shows in-domain, second column out-of-domain performance.

Declaration

I hereby declare that this dissertation is my own original work except where otherwise indicated. All data or concepts drawn directly or indirectly from other sources have been correctly acknowledged. This dissertation has not been submitted in its present or similar form to any other academic institution either in Germany or abroad for the award of any other degree.

Saarbrücken, 2023

Marius Mosbach