DisAI.eu

D1.5
Replication challenge report

Project Title	Improving scientific excellence and creativity in combating disinformation with artificial intelligence and language technologies
Contract Nº.	101079164
Type of Action	HORIZON-CSA
Topic	Disinformation Combating
Project start date	1st Dec 2022
Duration	36 months

Funded by
the European Union

Deliverable title	Replication challenge report
Deliverable number	D1.5
Deliverable version	1.1
Contractual date of delivery	30 Apr 2024
Actual date of delivery	30 Apr 2024, updated 1 July 2024
Nature of deliverable	Document, Report (R)
Dissemination level	Public (PU)
Work Package	WP1
Task(s)	T1.4
Partner responsible	UCPH
Author(s)	Marian Simko (KInIT), Simon Ostermann (DFKI), George Karantaidis (CERTH), Stefanos Papadopoulos (CERTH), Michal Gregor (KInIT), Kamil Burda (KInIT), Qiwei Peng (UCPH)

Abstract	The deliverable describes the organisation and results of the replication challenge targeted at early stage researchers. 11 early stage researchers were involved to get acquainted with the state of the art in their field of study and gain practical research experience by replicating existing research works while being mentored by experienced researchers. The deliverable contains the replication study reports created by students.

Keywords	KInIT, Training, Research, Replication study, Replicability, Reproducibility

History of Changes

Version	Date	Description/Note
1.0	30 Apr 2024	Final version as submitted by the contractual date of delivery
1.1	1 Jul 2024	Incorporated feedback based on the mid-term review: • Section 4 Mentors Conclusions expanded to include *Obstacles and Overcomings* subsection

Table of Contents

1 Introduction

This deliverable reports on the replication challenge activity realised within the DisAI project. Its original aim was to organise activity focusing on early stage researchers (ESRs), primarily focusing on doctoral students. By conducting a replication study under the mentorship of a senior researcher, the aim was to get students better acquainted with the state of the art in their field of study and gain practical research experience.

The replication challenge was organised from October 2023 (M11) to April 2024 (M17). Together, 11 students were invited to participate, with 9 students finishing the challenge. Besides PhD students, also master students were involved, addressing both the actual lower number of PhD students in KInIT and collaborating academic institutions (due to the overall decrease of interest in PhD study) that was anticipated at the time of project proposal writing, and lack of interest in participation.

The activity culminated with a 1-day in-person workshop in Bratislava, during which the ESRs presented the results of their work. The event was enriched by an invited lecture from Rafael Tolosana Calasanz from the University of Zaragoza, Spain, who addressed the overall phenomenon of reproducibility in AI and discussed results achieved in the Horizon Europe AI4Europe project[1]. The participants benefited from feedback and further discussion on their presented work provided by additional invited experts invited as guests to the workshop.

The result of students' work is attached to this deliverable as Appendix A.

[1] https://www.ai4europe.eu/

2 Organisation

The activity organisation started intensively in June 2023 (M7), setting the schedule for autumn and winter 2023/2024, identified as suitable due to the standard academic year organisation. Originally, three mentors were involved, with 12 research papers identified as suitable for replication.

Call for participation was issued in October 2023 at the DisAI project web site[2], with expected duration of replication studies from November 2023 to January 2024. As a lower number of PhD students had been studying in KInIT than expected at the moment of project writing (the original aim was to include at least 10 early stage researchers), we individually approached 7 departments[3] from Slovakia and Czechia with focus on AI to cover and attract a wider audience of potential PhD student participants.

By the end of October 2023, only 4 PhD student participants applied despite the focused promotion. To increase the number of participants, and to fulfil the objective of this activity, we decided to reach out to master students at the Faculty of Mathematics, Physics and Informatics, which is the only faculty in Slovakia, where Natural language processing course is included in the curriculum.

We re-issued the call for participation in November 2023 and 10 more students expressed interest in the replication challenge. We selected 7 of them to participate based on their topic preference, mentoring capacity and motivation letters they supplied. A new schedule was introduced for this run. One mentor was added by CERTH and one was replaced by KInIT due to the contract change.

Finally, the following mentors were involved in the replication challenge:

- Dr. Simon Ostermann, Lab Manager, Senior Researcher, Group Lead "Data and Resources", DFKI, offering 4 papers and a wild-card option[4], covering the topic of Parameter-Efficient Multilingual Natural Language Processing.
- Dr. Stefanos Papadopoulos and Dr. George Karantaidis, AI Researchers at Information Technologies Institute, CERTH, offering 6 papers and a wild-card option, covering the topic of Multimodal AI for Misinformation Detection.
- Dr. Matúš Pikuliak, Senior researcher, NLP Team, KInIT, offering 2 papers and a wild-card option (for Run #1), covering the topic of Multilingual NLP.
- Assoc. prof. Michal Gregor, Expert researcher, NLP Team, KInIT, offering 2 papers (for Run #2), covering the topic of explainability for vision-language models and prompting for passage retrieval in large language models.

[2] https://disai.eu/replication-challenge/

[3] Department Of Computer Graphics And Multimedia, Faculty Of Information Technology, Brno University of Technology, Brno; Division of Cognitive Science and Division of Artificial Intelligence, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava; Department of Electronics and Multimedia, Center of Intelligent Technologies and Telecommunications and Center of Intelligent Technologies, Faculty of Electrical Engineering and Informatics, Technical University of Kosice, Kosice; Institute of Computer Science and Mathematics, Faculty of Electrical Engineering and Information Technology of Slovak University of Technology in Bratislava;

[4] A participant was allowed to select another paper for replication study, if approved by the mentor.

A mentor from the University of Copenhagen was not involved in the activity due to the personnel unavailability. This did not affect the outcomes of activity.

At the student side, the 11 early stage researchers participated in the replication challenge, 4 females and 7 males): 3 PhD students from KInIT, 1 PhD student from Technical University of Kosice, 7 master students from Comenius University, Bratislava. The replication challenge lasted until February 7th (Run #1) and April 11th (Run #2). During the process, two students from Run #2 dropped off. Together, 9 reports were submitted and presented at the final event: The Reproducibility and Replicability Workshop, organised on April 22th in Bratislava (see section 3 Final workshop).

Selected Research Works for Replication

Parameter-Efficient Multilingual Natural Language Processing (Dr. Simon Ostermann)

The papers selected for this track of the replication study evolve around efficient methods for various tasks of natural language understanding, with strong applications and relevance for multilingual use cases. The selected papers present approaches to enhancing the parameter efficiency, data efficiency, and effectiveness of language models across multiple languages without necessitating extensive computational resources or complete model retraining. Each study introduces methods that are especially pertinent to managing the complexity and diversity inherent in multilingual contexts. Two papers present basic methods for parameter-efficient NLP, Adapters and Prompts, and two papers present more advanced modelling techniques based on soft prompts.

- **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**[5] - This paper builds the foundation for most of modern NLP, introducing one of the first pretrained large language models. This study was offered to less experienced students as an alternative for replication.
- **Parameter-Efficient Transfer Learning for NLP**[6] - This study introduces Adapters, a parameter-efficient learning technique, meaning they achieve high performance while modifying a minimal number of parameters in pre-trained models. Such Adapters can for example be used to finetune multilingual models for single languages in an efficient way.
- **The Power of Scale for Parameter-Efficient Prompt Tuning**[7] - This paper discusses the advantages of using prompt tuning, a method where a fixed number of parameters (prompts) are trained while the rest of the model remains

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[6] Houlsby, Neil, et al. "Parameter-efficient transfer learning for NLP." International conference on machine learning. PMLR, 2019.

[7] Lester, Brian, Rami Al-Rfou, and Noah Constant. "The Power of Scale for Parameter-Efficient Prompt Tuning." Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021.

unchanged. The scalability of this approach is shown to be highly effective for parameter efficiency.

- **ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts**[8] - This paper introduces a method that leverages soft prompts combined with attention mechanisms to tune models for multiple tasks simultaneously in a parameter-efficient manner.
- **SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer**[9] - The SPoT technique focuses on adapting frozen models through the use of soft prompts and their transferability across tasks.

Projects 1 and 2 reimplemented the BERT paper. Project 3 reimplemented the paper "Parameter-efficient transfer learning for NLP", Project 4 the ATTEMPT paper and Project 5 the SPoT paper.

Multimodal AI for Misinformation Detection (Dr. Stefanos Papadopoulos, Dr. George Karantaidis)

The papers chosen for this segment of the replication study focus on techniques for detecting multimodal misinformation and automated fact-checking through multimodal deep learning. These selected papers showcase methodologies across different phases of the automated fact-checking process, including evidence retrieval, verification prediction, and explanation generation. They leverage cutting-edge deep learning advancements, particularly large pre-trained neural networks.

- **SpotFake: A Multi-modal Framework for Fake News Detection**[10] - This paper introduces a multimodal framework that utilizes textual and visual features of an article to alleviate the challenging task of fake news detection. It employs language models, such as BERT, and the pretrained model VGG-19 to extract the textual and visual features, respectively.
- **Logically at Factify 2: A multi-modal fact checking system based on evidence retrieval techniques and transformer encoder architecture**[11] - The paper addresses the task of automated multimodal fact-checking by utilising: 1) an evidence retrieval component that selects the most relevant sentences to the claim from the full article, 2) a combination of pre-trained cross-modal and unimodal models, and 3) a Transformer Encoder architecture for cross-modal veracity.

[8] Asai, Akari, et al. "ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts." Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. 2022.
[9] Vu, Tu, et al. "SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer." Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2022.
[10] Singhal, Shivangi, et al. "Spotfake: A multi-modal framework for fake news detection." 2019 IEEE fifth international conference on multimedia big data (BigMM). IEEE, 2019.
[11] Zhang, Y., Tao, Z., Wang, X., & Wang, T. (2023). INO at Factify 2: Structure Coherence based Multi-Modal Fact Verification.

- **INO at factify 2: Structure coherence based multi-modal fact verification**[12] - The paper proposes an ensemble machine learning approach to tackle the task of automated multimodal fact-checking that leverages semantic, lexical and visual similarities among the image-text under verification and the external evidence (images and articles).
- **End-to-end multimodal fact-checking and explanation generation**[13] A challenging dataset and models - The paper proposes an end-to-end framework for three sub-tasks related to automated fact-checking, namely: multimodal evidence retrieval, claim verification, and explanation generation that leverages large pre-trained models such as CLIP, Sentence BERT and BERT and BART.

Project 6 reimplemented the SpotFake paper. Project 7 reimplemented the *Logically at Factify 2* paper. Project 8 reimplemented the *INO at Factify 2* paper.

Explainability for Vision-Language Models and Prompting for Retrieval (Assoc. Prof. Michal Gregor)

This track combines two topics: that of explainability for vision-language models (replicating a method based on a very robust indicator, which is also very agnostic of models' architectures) and of prompting/fine-tuning large language models for the task of retrieval.

- **MM-SHAP: A Performance-agnostic Metric for Measuring Multimodal Contributions in Vision and Language Models & Tasks**[14] – This paper explores a robust method for assessing in what proportions a multimodal model uses the individual modalities. The approach is based on Shapley values, but adapted for vision-language architectures. One considerable advantage of the method is that, unlike the majority of explainability methods in deep learning, it is relatively agnostic of the model's particular architecture.
- **Scaling Sentence Embeddings with Large Language Models**[15] – The paper considers the use of large language models (LLMs) in computing embeddings usable for passage retrieval. It presents ways to implement both: a fully prompt-based approach, and an approach, which also leverages fine-tuning. This work is very relevant in the sense that modern LLMs exhibit significantly improved understanding of natural language when compared to more traditional sentence embedding models (generally based on BERT-like architectures). Furthermore, modern LLMs also tend to have much larger context lengths, which can be crucial in some contexts.

---

[12] Verschuuren, P. J., Gao, J., van Eeden, A., Oikonomou, S., & Bandhakavi, A. (2023). Logically at Factify 2: A Multi-Modal Fact Checking System Based on Evidence Retrieval techniques and Transformer Encoder Architecture.

[13] Yao, B. M., Shah, A., Sun, L., Cho, J. H., & Huang, L. (2023, July). End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 2733-2743).

[14] Parcalabescu, L., & Frank, A. (2022). MM-SHAP: A Performance-agnostic Metric for Measuring Multimodal Contributions in Vision and Language Models & Tasks. In ACL 2023.

[15] Jiang, T., Huang, S., Luan, Z., Wang, D., & Zhuang, F. (2023). Scaling sentence embeddings with large language models.

In project 10, a participant was working on replicating the MM-SHAP paper. A participant was working on the scaling sentence embeddings paper in project 11.

Final Schedule

Run #I

- October 4th – Call for Participation
- October 4th – October 20th – Application and ESR-Mentor Matching
- November 2nd – February 7th – Replication Study Realisation
- February 7th – Report submission
- April 15th 9:00 – Review, Camera-ready submission

Run #II

- November 28th – Call for Participation
- November 28th – December 10th – Application and ESR-Mentor Matching
- December 11th – April 11th – Replication Study Realisation
- April 11th – Report submission
- April 18th 12:00 – Review, Camera-ready submission

April 22nd – Final Workshop in Bratislava

3 Final Workshop

The replication challenge culminated with the 1-day workshop titled "*The Reproducibility and Replicability Workshop*" organised on April 22nd 2024 in Bratislava as a hybrid event.

The overall schedule of the event was as follows:

- 10:00-10:15 Welcome
- 10:15-11:15 **Rafael Tolosana Calasanz & Andrea Hrčková: Reproducibility in AI** (AI4Europe)
- 11:15-12:15 Lunch
- 12:15-13:45 **Replication Studies: Part I – PhD students**
- 13:45-14:15 Coffee Break
- 14:15-16:15 **Replication Studies: Part II – master's students**
- 16:15-16:30 Goodbye

The programme for the event was set up to attract a wider audience and included the invited talk on Reproducibility in AI presented by Rafael Tolosana Calasanz, a distinguished researcher from the University of Zaragoza, Spain, who addressed the overall phenomenon of reproducibility in AI and discussed results achieved in the Horizon Europe AI4Europe project[1]. Together with Andrea Hrčková from KInIT, they delved into the impact of reproducibility in AI research. Andrea identified and addressed key challenges encountered by PhD students during replication and reproducibility studies, drawing from her own research findings. Rafael introduced essential reproducibility tools within the AI on Demand (AIoD) architecture, such as RAIL, REANA, and the catalog. He explored actionable strategies for fostering reproducibility within the AI community through the AIoD reproducibility process. The aim was to equip researchers with practical insights and tools to enhance reproducibility in their work, thereby advancing scientific integrity and progress in the field of AI.

The following sessions were dedicated to presentations of replication challenge participants. In their presentations, they were instructed to cover the problem statement, the original work they were replicating, results and lessons learned. In the discussion (8 min) following their presentation (12 min), they were provided with additional feedback, benefiting of the presence of additional experienced researchers in the auditorium:

- Claudia Borg, expert researcher and senior lecturer from University of Malta
- Marek Suppa, lecturer of Natural Language Processing course at the Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, Head of Data at Sli.do
- Michal Kompan, chief research officer and expert researcher at KInIT
- Martin Tamajka, lead research engineer and researcher at KInIT
- Robert Moro, senior researcher at KInIT
- Sebastian Kula, senior researcher at KInIT (online)
- with mentors: Simon Ostermann (DFKI), Michal Gregor (KInIT), George Karantaidis (CERTH, online), Stefanos Papadopoulos (CERTH, online)

4 Mentors Conclusions

Assessment of Participants

Especially the PhD students taking part in the study showed an impressive involvement in the tasks set out in the scope of the replication study.

In particular, students implementing projects 4 and 5 were not only involved in the replication, but also engaged in further activities to extend the research done as part of the challenge, as well as creative and motivated to experiment beyond what was necessary for the study.

The participant implementing project 3 found it harder to commit the necessary time for extensions on the replication study, but in general achieved a satisfactory performance during the study.

The two students that took part and that implemented projects 1 and 2, found it hard to commit a larger amount of time into the study due to other duties at university. They also found it difficult to engage with the presented papers because most of them were too advanced for them to reimplement the approaches presented. They thus fell back to implementing more straightforward experiments on pretrained models. Most of the work was done in cooperation between the students, which additionally limits the amount of work that each of them was able to put in.

The participant implementing project 6 committed themself to completing their replication project despite facing time management issues due to other duties at university and some personal issues. The participant replicated the assigned paper, compiled the report, and prepared the final presentation. Participant's determination and dedication to overcoming obstacles were evident, showcasing their strong work ethic and commitment to the assigned paper. The final outcomes were satisfactory, though there was potential for improvement had there been more efficient time management.

The participants of project 7 and 8 displayed notable dedication in the tasks set out in the scope of the replication study. Despite encountering initial challenges, primarily stemming from ambiguities in the manuscripts, they swiftly adapted their approaches and achieved results that closely mirrored those of the original papers.

Regrettably, the participant of project 9 withdrew from the replication challenge soon after confirming their participation, citing personal reasons but also difficulty to keep up with the demands of the replication study.

The participant of project 10 worked on replicating the MM-SHAP paper. They actually managed to progress quite far with replicating the implementation of the method itself, but then unfortunately decided to abandon the challenge to keep up with their workload. Consequently, they did not produce a comparison with the results in the original paper, and did not attend the final workshop.

The participant of project 11 replicated the “Scaling Sentence Embeddings” paper. The participant were committed to the task from the beginning. While they required a significant amount of assistance in the initial phase of the challenge, in the later phase, having acquired the relevant knowledge and practised the associated practical skills, they worked in a completely independent way and managed to achieve good results.

Obstacles and Overcomings

Hardware limitations. We sometimes faced a situation in which hardware was not sufficiently available, especially for participants not working at KInIT, such that short-notice workarounds had to be identified, a non-ideal situation. We circumvent restrictions on hardware on short notice by rolling back to smaller models and smaller datasets, concentrating the replication on such simpler parts of the works to be replicated. In some cases, the difficulty and complexity of papers was too high for some of the participants. In that case, we flexibly assigned less challenging works for the replication and also rolled back the complexity of the replication in terms of which toolkits were allowed to be used (less low-level, more ready-made Hugging Face methods).

Student motivation. Decreasing student motivation was noticeable in case of master participants, primarily due to the less flexible semester organisation. In one such case, the participant was willing to replicate the study, but it was perceived as part of a semester's coursework and not their top priority. When they faced difficulties, they often allowed time to pass without addressing the difficulties promptly. The solution involved active engagement from the mentor, who scheduled calls to discuss the code line by line and explain the peculiarities of both the paper and the code. This guidance enabled the participant to successfully complete the replication. For future replications involving students, it is advisable to arrange a more structured and tight schedule. This approach can help participants feel more confident and ensure steady progress throughout the replication process.

Skill and experience disparity. Due to skill and experience disparities of participants, there were gaps in understanding even after discussing the task in detail. This led to the first results being highly divergent from the original papers due to misunderstandings and mistakes in implementation. To address this, we established a shared code repository allowing the mentor to monitor progress or detect mistakes in greater detail and we took the time to clarify and ensure that students were fully comprehending various technical terms, methodologies and programming frameworks.

Process Retrospective

For the future, it will be helpful to select a range of papers of varying difficulty and to openly communicate such difficulty estimations. It will also be helpful to directly discuss hardware requirements and possibilities for the participants to access hardware during the study. It will also be helpful to formulate clearer suggestions for the mentoring, such that it can be ensured that the mentoring schemes work similar across mentors.

5 Students Conclusions (Lessons Learned)

Students presentations from the final workshop are available online[16]:

- Session I (PhD students)
- Session II (Master students)

When preparing final presentations, participants were instructed to include at least one slide on lessons learned. We include content from lessons learned slides provided by students to illustrate the perception of replication challenge by students.

Participant #1

- Insight into the topic of Adapters
  - new valuable experience
  - knowledge about possible applications of Adapters into future research
- Difficulties with using the paper's code
  - the paper was accompanied by hardly legible code for an outsider
  - the original implementation was challenging to reproduce
  - the easier solution was to use publicly available library
- The need for proper documentation for future reproducibility
  - insufficient documentation and code explanation hampers possible future reproduction
  - the coherency of the article as a whole is partially undermined by the lack of extensively explained methodology in the form of code

Participant #2

- Don't trust the code
  - Authors usually submit the code in a hurry and usually just copy everything to a single repo and provide none or only a little documentation
  - What is in the paper is sacred (even if it contradicts with the code)
- Custom implementations over public libraries?
  - Gets us a better understanding and a better control
  - But can divide our attention to not so important tasks
  - Overriding methods and contributing to open libraries may be the solution
- Think about reproducibility when doing research
  - "Would you enjoy doing replication study of your own work?"
  - What makes our results reproducible? (stability, environment)

Participant #3

- Evidence retrieval
  - Interesting idea, but there is no direct comparison between other approaches such as models for long texts or summarization

[16] https://disai.eu/replication-challenge/

- Not uniformly beneficial
- Explain the architecture of model in detail and not omit anything or let it for the readers to decode
  - For example layers, their parameters, random seed and much more
  - Paper should present the model, so the architecture is understandable without the code
- Publishing the code should be part of publishing the paper
  - Robustness of models
  - Importance of transparency in research
  - Provides a deeper understanding of the intricacies involved

Participant #4

- Insight into the PEFT methods
  - PEFT methods can outperform full model fine-tuning
- Paper will put up with anything that is written on it
  - Is the information in the paper correct?
  - The differences in the paper and in the source code
- Refactoring is important for code understanding, but even more important is to remember to publish all the code for other researchers to reproduce the results!

Participant #5

- My code writing actually isn't as bad as I thought
- Bachelor thesis was good for something
- It isn't that bad to submit assignments ahead of time
- This semester would be easier if we got more 2 for 1 deals like this from our teachers
- Jokes aside, it was nice to see how people in real world work
- I learned yet a couple new things that I can use in ML
- And I learned that this thing called replicability in science is not for granted

Participant #6

- Timing in life is important
- More communication -> fewer misunderstandings
- GPU is still a scarce resource in 2024
- If you're running a program that's supposed to run for 10+ hours, make sure it doesn't stop after any kind of error
- Replication tends to be successful when the paper has 97859 citations on Google Scholar

Participant #7

- Importance of reproducibility in science
- Replicating something already done is not as easy as it may seem

- New knowledge about creating machine learning models
- That I need to work on my time management

Participant #8

- replicating can be really strenuous and hard
- it should not be taken for granted
- GPU - really scarce resource
- really like this type of assignments
- it is great to participate in challenges similar to this
- cooperation is the key for better results
- try to work on my time management and submitting on time

6 Conclusions

The replication challenge was a very interesting experience for all the parties involved. Despite several obstacles (low interest, partially caused by lower number of PhD students in KInIT than originally expected), we managed to involve 11 young people at the early stage of their career to get familiar with research works related to the topics of the DisAI project. The activity successfully culminated in a one-day workshop with a participation of many experienced researchers. The activity contributed to:

- improved scientific capacity in KInIT – three PhD students from KInIT were involved
- increased number of Slovak research institutions dealing with DisAI topic
- increased awareness on the topics of disinformation combating
- increased visibility of KInIT and the DisAI project
- Increased research competences of 11 participants

An important outcome of the activity is experience with organising replication studies. Improvement points identified in the process retrospective will make organisation of such activity in the future even better.

Appendix A: The Proceedings of Replication Studies

The replication study reports submitted by replication challenge participants are included.

Table of Contents

Parameter-Efficient Transfer Learning for NLP: A Replication Study

Viliam Balara[1], **Simon Ostermann**[2] and **Kristína Machová**[1]
[1] Technical University Košice, Košice, Slovakia
[2] German Research Institute for Artificial Intelligence (DFKI), Saarland Informatics Campus, Germany
viliam.balara@tuke.sk, simon.ostermann@dfki.de, kristina.machova@tuke.sk

Abstract

With the onset of the widespread use of large language models, fine-tuning the pre-trained models presents an effective transfer mechanism for NLP. However, in the case of downstream tasks, fine-tuning performs unsatisfactorily due to being parameter inefficient, the main reason being that an entirely new model trained from scratch has to be created for every particular task. To counteract this undesirable behaviour, the proposed solution by Houlsby et al. (2019), the adapter modules, yields increased performance while providing a compact and extensible model, with the advantageous feature being the need to add only a minuscule sample of trainable parameters. Thus, new tasks can be effortlessly added without retraining the whole existing model. The original parameters remain untouched, therefore a high degree of parameter sharing is present. In the original paper, the effectiveness was demonstrated with the BERT Transformer model accompanied by the GLUE benchmark as well as 26 diverse text classification tasks. In the original article, Adapters attained almost state-of-the-art performance in regard to the original timeframe, while minimally increasing the number of parameters per individual task. On GLUE, the attained results were within 0.4% of the performance of full fine-tuning, adding only 3.6% parameters per task. By contrast, fine-tuning necessitates the training of 100% of parameters per task. In our replication of this research, we will focus on the GLUE benchmark and simultaneously add additional variations of the BERT model to assess the performance of adapters.

1 Introduction

Transferring already acquired knowledge from pre-trained models has the benefit of strong performance, especially in the case of NLP tasks (Radford et al., 2018). In the time of the conception of the original article (Houlsby et al., 2019), BERT, a Transformer network which was trained on substantial text corpora with an unsupervised loss, attained state-of-the-art performance on text classification as well as extractive question answering (Devlin et al., 2019). Therefore, our goal is to reproduce the results of the original paper and include additional experiments to prove the versatility of adapter modules. Sharing of parameters proves to be beneficial in a wide range of applications, such as large language models, cloud services or Generative AI. Therefore, it is important to convey as much as possible of already gained knowledge from existing model to another. The method of adapter modules benefits from the preservation of existing knowledge while maintaining low retraining costs.

The two most common transfer learning techniques in NLP are feature-based transfer and fine-tuning. On contrary, the original paper proposes the use of a novel solution – adapter modules (Rebuffi et al., 2018). Features-based transfer involves pre-training real-valued embeddings vectors. The embedding are either at the word, sentence, or paragraph level (Le and Mikolov, 2014). Afterwards, the embeddings are passed to custom downstream models. During the fine-tuning, the copying of the weights from a pre-trained network takes place, which are in the next step tuned on the downstream task. However, it is important to point out the fact that Both feature-based transfer and fine-tuning require a new set of weights for every new task. In the terms of efficiency, fine-tuning is considered more parameter efficient if the lower layers of a network are shared between tasks. On the other hand, the proposed adapter tuning method achieves even higher rate of efficiency in the number of parameters. This difference is portrayed on Figure 1. Adapters can be described as new segments which are put additionally between the layers of a pre-trained network.

To combat the drawbacks of costly retraining and the loss of already learned information, the adapters bring the idea of freezing the shared parameters. Therefore, the model is enabled to retain the memory

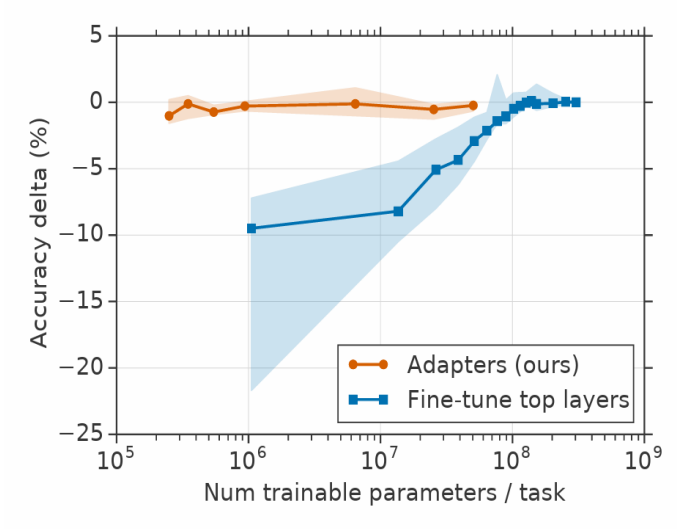

Figure 1: Trainable parameters comparison (Houlsby et al., 2019)

of previous tasks. The key innovation of the original paper was the design of an effective adapter module and its integration with the base model. The proposes bottleneck architecture performed satisfactorily on GLUE benchmark, almost matching the fully fine-tuned BERT while utilizing only 3% of task-specific parameters. Similar results were attained on further datasets and SQuAD extractive question answering. Our results did not achieve the levels of efficiency of the original paper, however, they support the efficiency of adapters for NLP tasks.

2 Adapter tuning for NLP

The presented strategy of adapters contained three key properties (Houlsby et al., 2019): (i) it attained good performance, (ii) it permitted training on tasks sequentially, implicating, it did not require simultaneous access to all datasets, and (iii) it added only a minuscule fraction of additional parameters for each task. The aforementioned properties are generally useful in environments where a significant number of tasks are related to each other, therefore mutual sharing of already learned knowledge is highly desirable. Prime examples are tasks present in GLUE or SQUaD benchmark dataset.

To achieve those features, the original paper contained a proposition of a new bottleneck adapter module. Tuning of such a module was done by adding a negligible amount of parameters to a model, which were subsequently trained on a downstream task. During vanilla fine-tuning of deep neural networks, a modifications are made to the top layer of the network. Contrasting to this approach, adapter modules perform architectural modifications that aim to re-purpose a pre-trained network for a downstream task. Such a case is the adapter tuning strategy which involves injecting new layers into the original network. The weights of the original network remain unmodified, while on the other hand, the new adapter layers are randomly initialized. In the case of standard fine-tuning, the new top layer and the original weights are co-trained, thus resulting in the loss of already gathered information. Contrary to this, during adapter tuning, the parameters of the original network remain frozen and therefore may be subsequently shared by a multitude of various tasks. Adapter modules have two main features: a small number of parameters, and a near-identity initialization. The adapter-based tuning is used with the transformer architecture, which is known to achieve state-of-the-art (SoTA) performance in a wide multitude of NLP tasks.

3 Experiments

In this chapter, the methodology and used data are described in a more elaborate manner. We replicate the experiments performed in the original paper, describe the dataset, and use methods in accordance with the original paper. Contrary to the original paper, we have omitted the use of SQuAD and additional datasets and have focused entirely on testing on the GLUE dataset.

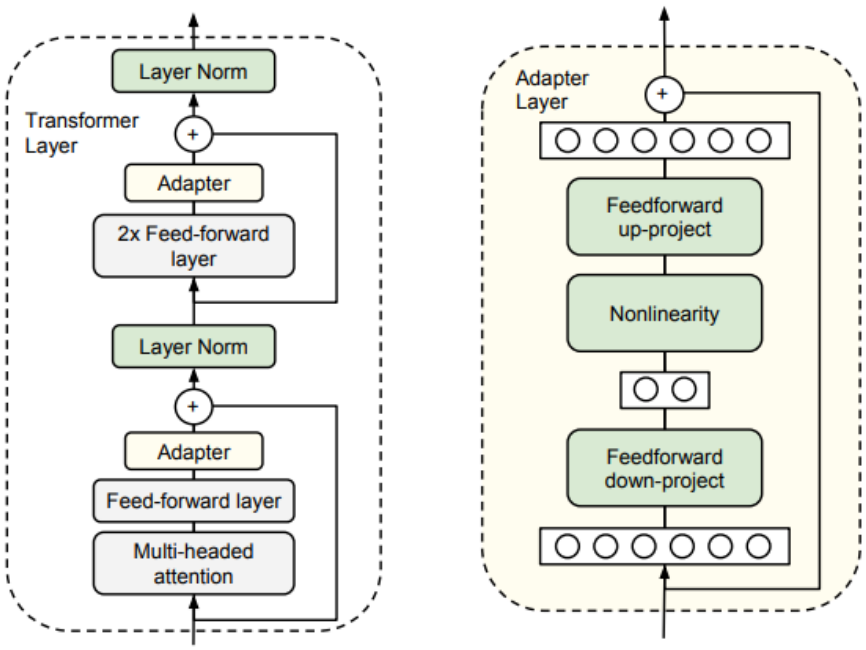

Figure 2: Adapter module (Houlsby et al., 2019)

3.1 Experimental settings

We have utilized a publicly available pre-trained version of BERT BASE model as our base model. In order to perform classification tasks of GLUE with BERT BASE model, we have tokenized the first token in each sequence as a special "classification token". Subsequently, we have proceeded with modifying the following parameters to assess the best setting for optimal results: learning rate, number of epochs, weight decay, and batch size. To further elaborate on the efficacy and versatility of the adapter modules, we have selected additional architectures to be equipped with adapter modules and then compared them with the original BERT BASE model. The models were trained with the use of Google Colab GPUs.

3.2 GLUE Benchmark

The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating, and analyzing natural language understanding systems. GLUE benchmark consists of: A benchmark of nine sentence- or sentence-pair language understanding tasks(datasets) built on established existing datasets and selected to cover a diverse range of dataset sizes, text genres, and

Corpus	|Train|	|Test|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	**1k**	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	**391k**	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	**20k**	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	**146**	coreference/NLI	acc.	fiction books

Figure 3: GLUE Benchmark

degrees of difficulty, A diagnostic dataset designed to evaluate and analyze model performance with respect to a wide range of linguistic phenomena found in natural language, and A public leaderboard for tracking performance on the benchmark and a dashboard for visualizing the performance of models on the diagnostic set. The format of the GLUE benchmark is model-agnostic, so any system capable of processing sentence and sentence pairs and producing corresponding predictions is eligible to participate. The benchmark tasks are selected so as to favor models that share information across tasks using parameter sharing or other transfer learning techniques. The ultimate goal of GLUE is to drive research in the development of general and robust natural language understanding systems. The structure and the constituent sub-datasets of glue accompanied by their evaluation metrics can be seen in Figure 3.

3.3 Additional architectures

The DistilBERT model was proposed in the blog post Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT, and the paper DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. DistilBERT is a small, fast and light Transformer model trained by distilling the BERT BASE model. It contains 40% less parameters compared to BERT-BASE, runs with the 60% increase in speed while preserving over 95% of BERT's performances as measured on the GLUE benchmark. DeBERTa (Decoding-enhanced BERT with disentangled attention) is a model that improves the BERT and RoBERTa models using two novel techniques. The first one is disentangled attention mechanism, where each word is represented with the use of two vectors that encode the content and position of the word and the attention weights among words are computed using disentangled matrices of their contents and relative positions. The second technique, an enhanced mask decoder is used to replace the output softmax layer to predict the masked tokens for model pretraining. Compared to RoBERTa-Large, a DeBERTa model trained on half of the training data performed consistently better on a wide range of NLP tasks, achieving increased efficiency levels on MNLI by +0.9% (90.2% vs. 91.1%), on SQuAD v2.0 by +2.3% (88.4% vs. 90.7%) and RACE by +3.6% (83.2% vs. 86.8%). ELECTRA is a pretraining approach that trains two transformer models: the generator and the discriminator. The generator's role is to replace tokens in a sequence and is therefore trained as a masked language model. The discriminator tries to identify which tokens that were replaced by the generator in the sequence. Masked language modeling (MLM) pretraining methods such as BERT corrupt the input by replacing some tokens with masks and then training a model to reconstruct the original tokens. While they produce good results when transferred to downstream NLP tasks, they generally require large amounts of computation to be effective. Instead of masking the input, the ELECTRA corrupts it by replacing certain tokens with plausible alternatives sampled from a small generator network. Afterward, instead of training a model that predicts the original identities of the corrupted tokens, a discriminative model is trained that predicts whether each token in the corrupted input was replaced by a generator sample or not.

4 Results

The achieved results are portrayed in the Tables 1, 2 and 3. Results of our replication are displayed on Table 1. Generally, our results did not achieve the performance that was stated in the original paper. However, the attained performance is close to the original and supports the claim of effectivity of adapter modules. To further support the claim, the additional tested models, depicted in Table 3, have achieved satisfactory performance, thus validating the versatility of adapter modules. However, significantly decreased was achieved in the case of ELECTRA. DeBERTa and DISTILBERT have achieved lower performance than BERT BASE with adapter module. The closest performance to the BERT BASE with adapter module was achieved by DeBERTa.

Table 1: Replicated results

	CoLA	SST	MRPC	STS-B	QQP	MNLIm	MNLImm	QNLI	RTE
BERT(Base)	58.2	92.5	90.6	69.3	70.7	84.8	85.3	89.9	69.3
Adapters 64	59.6	91.8	91.0	89.5	70.5	83.4	83.9	88.3	68.9

Table 2: Original results

	CoLA	SST	MRPC	STS-B	QQP	MNLIm	MNLImm	QNLI	RTE
BERT(Base)	60.5	94.9	89.3	87.6	72.1	86.7	85.9	91.1	70.1
Adapters 64	56.9	94.2	89.6	87.3	71.8	85.3	84.6	91.4	68.8

Table 3: Replicated results

Dataset	CoLA
BERT(Base)	58.2
BERT(Adapters 64)	59.6
DISTILBERT(Adapters 64)	54.1
DeBERTa(Adapters 64)	58.2
ELECTRA(Adapters 64)	49.7

5 Conclusion

We have replicated the research with the use of the original dataset. We have utilized various settings for the models that were used and selected the best performing ones. We have summarized the results and made a comparison to the original paper, with the addition of additional models. The adapter modules have showcased significant improvements over fine-tuning while maintaining the required level of performance.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2018. Efficient parametrization of multi-domain deep neural networks.

ATTEMPT – Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts: A Replication Study

Róbert Belanec[1,2], **Simon Ostermann**[3], **Ivan Srba**[2] and **Mária Bieliková**[2]
[1] Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic
[2] Kempelen Institute of Intelligent Technologies, Bratislava, Slovakia
[3] German Research Institute for Artificial Intelligence (DFKI),
Saarland Informatics Campus, Germany
{robert.belanec, ivan.srba, maria.bielikova}@kinit.sk, simon.ostermann@dfki.de

Abstract

Generative language models gained an increase in popularity shortly after the introduction of the transformer architecture (Vaswani et al., 2017), which resulted in a fast increase in the number of model parameters. Currently, large language models contain billions of trainable parameters, which makes them power and cost inefficient. Large language models also require significant amounts of training data, which especially benefits well-resourced languages. To address these problems, parameter-efficient fine-tuning methods have emerged. Parameter-efficient fine-tuning methods aim to fine-tune generally pre-trained language models while training only a fraction of parameters. In the scientific and academic community, authors often compare with the current state-of-the-art. However, for the results to be relevant and trustworthy, both of the works (state-of-the-art and compared) must be reproducible. In our work, we present the methodology and the results of our replication study of a parameter-efficient fine-tuning method introduced in the paper ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts. To replicate the results provided by the authors, we have conducted a series of experiments and we show that better-performing source prompts may contribute more to the overall results. We also point out a stability issue and provide examples of results that have a better score but are harder to replicate due to the randomness factors. Finally, we compare our results to the results provided by the authors and derive a conclusion based on a discussion.

1 Introduction

In recent years, generative language models have experienced a steady increase in popularity. After the introduction of the transformer architecture (Vaswani et al., 2017) for natural language processing, there has been a fast increase in the number of model parameters. The first widely-used transformer models contained millions of trainable parameters (e.g. BERT-Large having 340 million parameters (Devlin et al., 2019) and GPT having 117 million parameters (Radford et al., 2018)). Recent architectures contain billions of trainable parameters (e.g. GPT-2 having 1.5 billion parameters (Radford et al., 2019) and GPT-3 having 175 billion parameters (Brown et al., 2020)). With the rising trend of increasing the number of parameters to achieve better results, models often require a vast amount of computational resources for training. Besides their parameter hunger, large language models also require significant amounts of training data, which especially benefits well-resourced languages. The newest language models often perform sub-par for low-resourced languages, decreasing the exhibited trust in such models. Significant trust decrease is also caused by the loss of interpretability that correlates with the size of the newest language models.

Consequently, there is a strong motivation in the natural language processing research community to decrease the number of trained parameters and the need for large amounts of training data, while maintaining the results on downstream tasks. To address these problems, parameter-efficient and data-efficient fine-tuning methods have emerged. Parameter-efficient fine-tuning methods aim to fine-tune generally pre-trained language models while training only a fraction of the model parameters. Data-efficient finetuning methods aim to leverage the power of large pre-trained models and try to adapt them to specific tasks or domains with only minimal amounts of training data. Many state-of-the-art parameter-efficient models have been shown also to require less training data, which is why both problems can often be alleviated with a single method (Yu et al., 2022; Gu et al., 2022).

In the scientific and academic community, authors often compare with the current state-of-the-art. However, for the results to be relevant and trustworthy, both of the works (state-of-the-art and compared)

must be reproducible. This means, that by following the authors' publication, we should be able to derive the same results as provided by the authors. In our work, we present the methodology and the results of our replication study of a parameter-efficient fine-tuning method presented in the paper ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts (Asai et al., 2022).

2 Related Work

Language models. Recently, generative language models have experienced a breakthrough that started by introducing the transformer architecture (Vaswani et al., 2017), which was preceded by the introduction of novel methods in machine learning translation like Sequence to Sequence models and attention (Sutskever et al., 2014; Bahdanau et al., 2014). For natural language generation, the GPT (Radford et al., 2018) model was introduced. Shortly after, BERT (Kenton and Toutanova, 2019) architecture was introduced, replacing the bi-directional LSTM (Peters et al., 2018) with a bidirectional transformer architecture pre-trained to de-mask masked parts of a text sequence.

Recently, with the introduction of large language models (Radford et al., 2018, 2019; Touvron et al., 2023a,b; Jiang et al., 2023), the number of model parameters has risen from millions of trainable parameters to billions. These models require large amounts of computational resources and large amounts of data to train.

Large language models have also been trained to solve multiple natural language processing tasks. For example, authors of the T5 model (Raffel et al., 2020) pre-trained their model on a set of a multi-task mixture of unsupervised and supervised tasks. The T5 model is designed to solve text-to-text denoising problems by training on text-to-text format datasets with span corruption. Building upon T5 versatility, its improved version Flan-T5 (Chung et al., 2022) was introduced shortly after. Nevertheless, fine-tuning large language models (e.g. to perform in a multi-task setting) is also a parameter-heavy task, therefore, new methods of training have been introduced.

Parameter-efficient fine-tuning. The core idea of parameter-efficient fine-tuning is to train a neural network model while backpropagating only over a small fraction of parameters. One of the first works towards more efficient fine-tuning of the language models was a work introducing sequential *adapters* (Houlsby et al., 2019), which are small trainable feedforward neural network modules, that are inserted into transformer architecture layers, while keeping the rest of the model frozen. Adapters and their variations (Pfeiffer et al., 2021; He et al., 2022; Chronopoulou et al., 2023) are still heavily used since they provide flexibility for multi-task problems (e.g. by training multiple adapters for each task separately and swapping between them on demand).

Some parameter-efficient fine-tuning methods focus on the reparameterization of the original weights by introducing a smaller matrix that is then transformed into a bigger matrix that represents the δW that will be added to the base model weights. For example, the Intrinsic SAID (Aghajanyan et al., 2020) method uses a Fastfood transform to transform from the low-rank decomposing, but it is not that effective due to the high memory complexity of the Fastfood transform (Le et al., 2013). Building on top of the Intrinsic SAID method LoRA (Hu et al., 2021) introduced two separate matrices that form the resulting δW matrix. After the introduction of LoRA, other methods based on LoRA appeared. For example, QLoRA (Dettmers et al., 2023) uses quantization of model parameters to 4-bit NormalFloat and uses a paged optimizer to deal with the memory spikes.

Another parameter-efficient fine-tuning method that adds modules to the base models is *prompt-tuning* (Lester et al., 2021). Prompt-tuning trains embeddings (in a separate embedding module) that are prepended to the input embeddings before inserting them into the base model. Prompt tuning requires only less than 0.01% of the original parameters to train the model to a specific task. In parallel with prompt-tuning, *prefix-tuning* (Li and Liang, 2021) was developed. Instead of prepending a single matrix of weights to the first layer, in prefix-tuning, a matrix is prepended to each separate layer of a transformer architecture. Therefore, it requires around 1% of the original parameters, still a relatively small number. These methods can be classified as *soft prompt fine-tuning* methods (Liu et al., 2023; Vu et al., 2022; Asai et al., 2022; Hambardzumyan et al., 2021; Wang et al., 2023), as they are fine-tuning parameter-efficient soft prompts (i.e., which are not made by humans, when compared to hard prompts). These methods provide more significant parameter reduction than some methods incorporating adapters but also sacrifice a portion of the model input context.

Some recent works also focus on transferring soft prompt information like SPoT (Vu et al., 2022) and ATTEMPT (Asai et al., 2022). The SPoT method investigates the transferability of soft prompts on 160 task combinations. ATTEMPT focuses on fine-tuning the model using prompt tuning on multiple tasks.

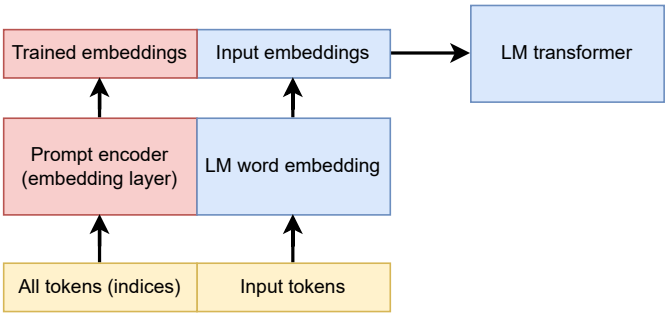

Figure 1: Diagram representing training process of the prompt tuning method. The blue color represents components with frozen parameters and the red color represents components with trainable parameters. The yellow color represents components without any weights (i.e. model inputs and outputs or utility functions).

After training the source soft prompts for each source task, ATTEMPT then trains a target soft prompt to solve the target tasks using a trainable attention layer to incorporate the source prompts accordingly. This method comes from the idea that learning to solve different tasks may contribute to solving other tasks. ATTEMPT is still very parameter-efficient as it trains only 0.4% of the original model parameters.

3 Replicated Methods

We have divided the ATTEMPT parameter-efficient fine-tuning method replication into two main parts: 1) the prompt tuning (Lester et al., 2021) replication, which we will use to train the source soft prompts and target soft prompts for prompt transfer, and 2) the ATTEMPT method replication. ATTEMPT is built on top of the prompt tuning and heavily relates to it. At the time of execution of this replication study, the prompt tuning parameter-efficient fine-tuning method is implemented in the publicly available parameter-efficient fine-tuning module (Mangrulkar et al., 2022). Regardless, we have decided to replicate the prompt tuning method, since we can build on it when implementing ATTEMPT. In this chapter, we will further describe both replicated methods.

3.1 Parameter-Efficient Prompt Tuning

The first parameter-efficient fine-tuning we will describe is prompt tuning (Lester et al., 2021). Prompt tuning is a parameter-efficient fine-tuning method that prepends a trainable embedding (prompt embedding) to the input embeddings to be forwarded as input to the base model. When training, the prompt embedding guides the language model to produce better results. Prompt tuning can therefore be seen as automatic prompt generation (which is also similar to adversarial reprogramming that can be seen in computer vision tasks (Elsayed et al., 2019)). This automatically trained prompt embedding is called a *soft prompt*.

Soft prompts are often compared with hard prompts. Hard prompts are prompts, that are made by a human (prompt engineer) to improve the results of already trained language models without any weight updates. This comparison of soft prompts and hard prompts can sometimes mislead the reader as it suggests that soft prompts are interpretable and readable in human language. Interpretation of a soft prompt in human language is not straight forward as the prompt embedding is trained separately. Therefore it has its own set of tokens (indices of the embedding) which is not a subset of the base model tokens and, therefore cannot be detokenized to the base model's vocabulary.

In a text-to-text approach using T5 (Raffel et al., 2020) as a base model we can interpret the language model as a conditional probability $Pr_\theta(Y|X)$ where Y is a sequence of tokens conditioned by a sequence of input tokens X parametrized by models weights θ. Prompting is a method that incorporates creating a hard prompt P which is a set of tokens prepended to input tokens $[P; X]$. Prompt tuning builds upon this idea and introduces parametrization of P with its weights θ_P. The conditional probability of generating Y is now $Pr_{\theta;\theta_P}(Y|[P; X])$. T5 embeds the set of input tokens into a matrix $X_e \in \mathbb{R}^{n\times e}$ where n is the length of the input token sequence e is the dimension of T5 embeddings. Prompt tuning represents soft prompts as a matrix of parameters $P \in \mathbb{R}^{p\times e}$ where p is the length of the soft prompt.

To gain a better overview of the prompt tuning parameter-efficient fine-tuning method, we provide a method diagram that can be seen in Figure 1. After the training, soft prompts include information about

the tasks that they were trained on. This can also mean that combining multiple soft prompts benefits in solving multi-task problems. The ATTEMPT method further builds upon this idea.

3.2 ATTEMPT – Attentional Mixtures of Soft Prompts

The ATTEMPT method takes advantage of the prompt tuning and builds on top of the method by introducing an attention module to create a mixture of pre-trained soft prompts based on how much they contribute to the result. The main hypothesis of the ATTEMPT method is when transferring the information from one soft prompt trained on a specific task, it can also contribute to solving other tasks, which is parallel to the language model transfer learning (e.g. model trained to summarize texts in the English language has already learned to understand the English language grammar and therefore can be trained easily to solve other English language tasks).

ATTEMPT can be trained in multiple ways – in a single-task setting (training each dataset separately) and in multi-task training on multiple concatenated datasets with an option to share the attention module across multiple tasks. In both of these settings, ATTEMPT trains a set of **target prompts** for each task (i.e. in a single task setting and a multi-task setting without a shared attention module the number of target prompts is 1) and uses a set of soft prompts to calculate the addition to the target prompt. We will further describe each of these training settings in the following paragraphs. The overview of the ATTEMPT method can be seen in Figure 2.

Single-task training setting. To train ATTEMPT in a single-task setting, we first need a set of pre-trained soft prompts (that authors call **source prompts**) and choose a soft prompt to initialize the target prompt (the target prompt can be also initialized with random weights, but authors used one of the pre-trained source prompts to initialize the target prompt). The target prompt is trained similarly to the prompt-tuning prompt (a matrix of parameters that is prepended to the matrix of input embeddings). What ATTEMPT does on top of that is to add a weighted sum of source prompts to the target prompt to produce an **instance prompt**. The weighted sum is calculated using attention scores from the attention module.

Multi-task training setting. To train ATTEMPT in a multi-task setting, we can train the target prompt similarly to the single-task setting, but concatenate multiple datasets into a single training dataset. We can then train the target prompt on a single train set and evaluate it on multiple evaluation sets separately. Multi-task ATTEMPT can be also trained with a shared attention module for multiple tasks. This means that for each dataset, we have a separate target prompt identified by a task ID. We then assign a task ID to each dataset before training. During training, we then retrieve the right target prompt based on the task ID of the input data. The process of retrieving the right target prompt is depicted in Figure 3. After we retrieved the right prompts for every input in the batch, we can continue with the instance prompt calculation as mentioned in the single-task training setting. This will increase the overall trained parameters, but the usage of only a single shared attention module for multiple target prompts compensates for the increase.

Attention module. The role of the attention module is to determine a score for the contribution of each source prompt based on the model input X, source prompts P, and the target prompts P_{target}. Since $X \in \mathbb{R}^{n \times e}$ and $P_j \in \mathbb{R}^{p \times e}$ have different sequence lengths, the attention module first does max pooling over the model input and source prompts to get $\hat{X} \in \mathbb{R}^e$ and $\hat{P}_j \in \mathbb{R}^e$. After the max pooling of a sub-network $\mathcal{G}$ projects the input $\hat{X}$ into the space of source prompts. The sub-network $\mathcal{G}$ consists of one downsampling fully connected input layer $H_{down} = W_{down}^T(\hat{X})$ and one upsampling fully connected layer with a SiLU (Elfwing et al., 2018) non-linear activation function $H_{up} = W_{up}^T(SiLU(H_{down}))$. As an output layer, there is a layer norm layer $H_{out} = LayerNorm(H_{up})$ after the upsampling layer. Finally, the attention module computes the attention score a_j by multiplying the $\hat{P}_j$ and H_{out} and applies a softmax over the scores as follows.

ATTEMPT also scales the attention scores with temperature T (Radford et al., 2021) to avoid making the attention over-confident. To calculate an instance prompt ATTEMPT adds a weighted sum to the target prompt as follows:

$$P_{instance} = P_{target} + \sum_{j=1}^{t+1} a_j P_j \tag{1}$$

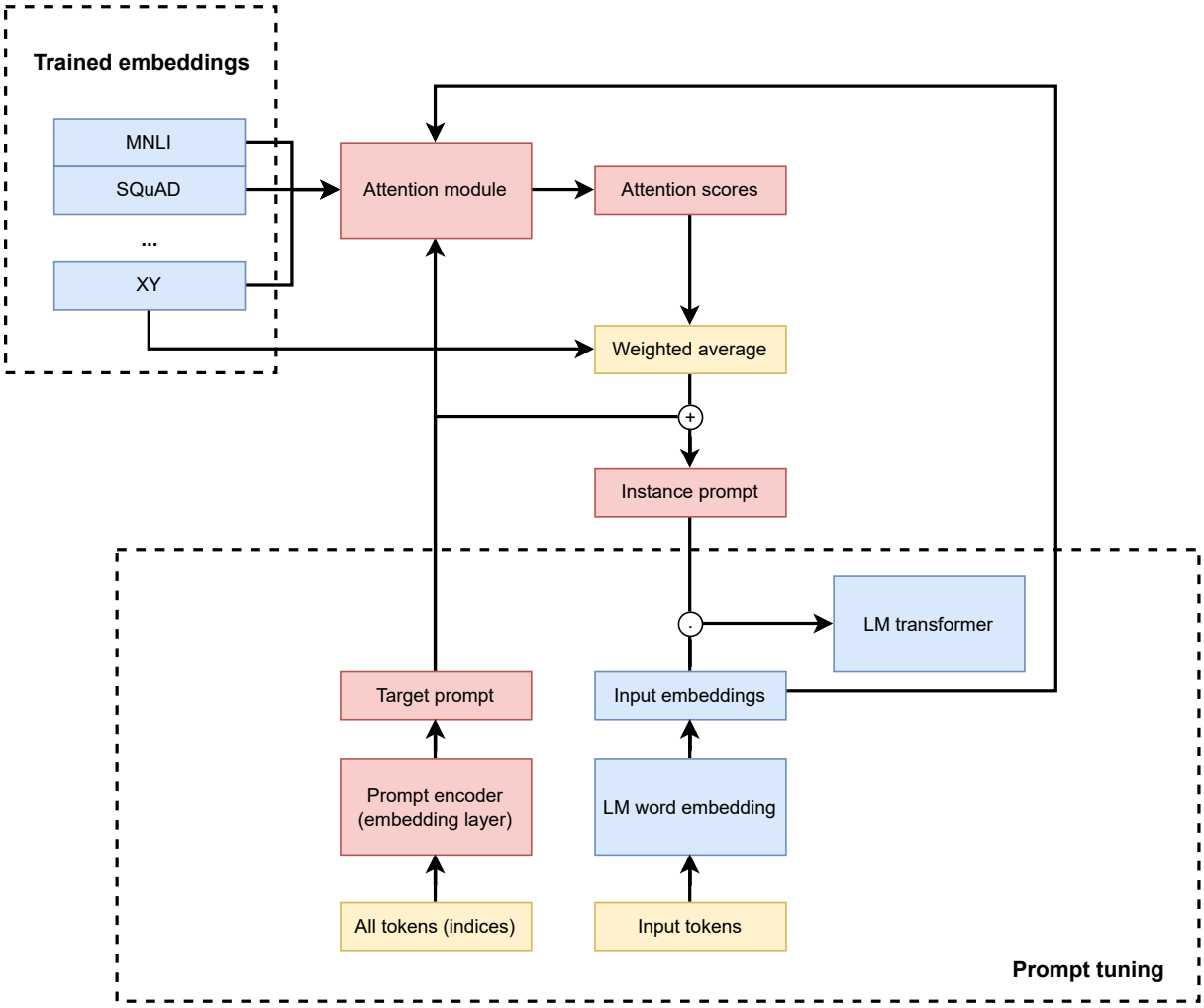

Figure 2: Diagram representing training process of the ATTEMPT method. The blue color represents components with frozen parameters and the red color represents components with trainable parameters. The yellow color represents components without any weights (i.e. model inputs and outputs or utility functions). The dot sign operation represents prepending the instance prompt to the input embeddings. The plus sign operation represents the addition of weighted average interpolation and target prompt from eq. 1.

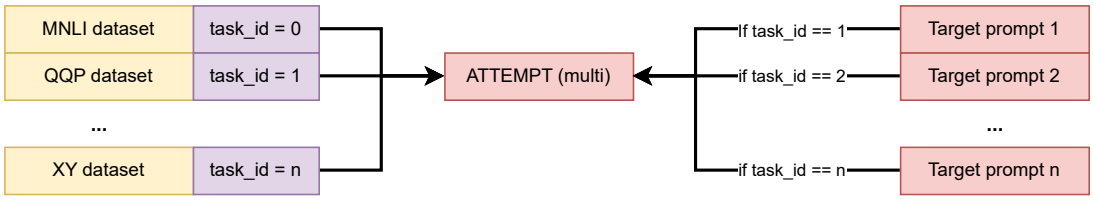

Figure 3: Diagram representing the process of target prompt selection when using shared attention across multiple target prompts. The red color represents components with trainable parameters and the yellow color represents components without any weights (i.e. model inputs and outputs or utility functions). The purple color is to represent added information to datasets.

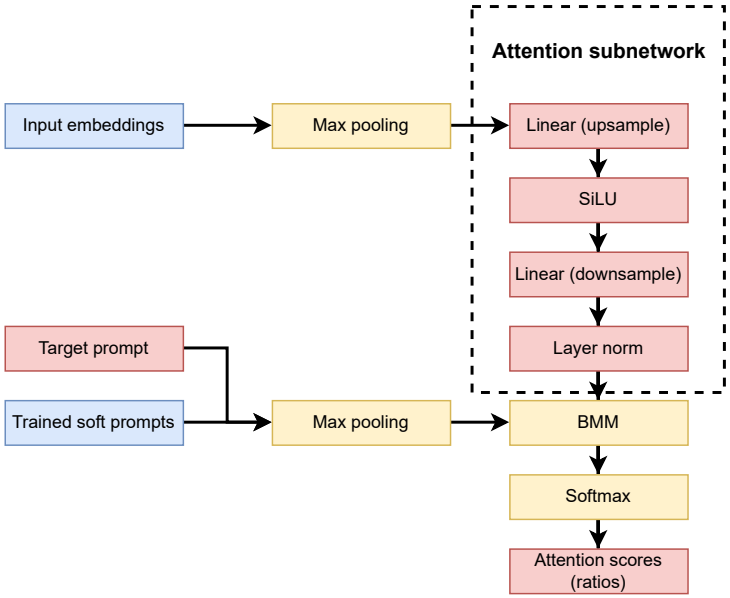

Figure 4: Diagram representing the architecture of attention module. The blue color represents components with frozen parameters and the red color represents components with trainable parameters.

Where P_{target} represents the task-specific part and the weighted sum represents a composition from different tasks that differs from different instances of the same task. As shown from eq. 1, the selection of $1 + a_{t+1}$ weights for the target task enables the ATTEMPT to use the knowledge from the target prompt when the knowledge from soft prompts is insufficient. These are some of the theoretical details from the ATTEMPT article, that we have built upon in the implementation phase of our replication study.

4 Implementation

Implementing the replicated method is an important part of our replication study. We implement the prompt tuning parameter-efficient fine-tuning method as well as the ATTEMPT parameter-efficient fine-tuning method. We use Python with deep learning modules (i.e. PyTorch, Transformers). All of our source code can be found in our GitHub repository[1]. Required Python packages are in the *requirements.txt* file. The original ATTEMPT implementation can be found in the authors' repository[2].

The *run.py* script creates a *PeftTraning* object that contains all the information about a single training run and handles the data pre-processing and training in the *run* method. During the dataset pre-processing, each dataset has a specified preprocessor function (in *tasks/tasks.py* file) to transform data into text-to-text setting and a formater function to put the transformed data into seq-2-seq format. The ATTEMPT authors use the preprocessing available in the T5 implementation, but instead of using words for classification (i.e. entailment, neutral, contradiction), the authors used numbers for classification (i.e. 0, 1, 2). We suspect that this change may result in pre-trained T5 model sub-optimal performance[3]. However, to achieve similar results as ATTEMPT, we used the same preprocessing as the authors did. The datasets are then split into training, validation, and test sets. Large datasets (over 10k samples) have a validation set split into 1000 validation samples and the rest for the test set; the small datasets (less or equal to 10k samples) have a validation set split into two halves, which are validation and test sets. We have used seed 42 to match the authors' seed for the dataset shuffle.

The *PeftTraning* also creates a *PeftConfig* and initializes a pre-trained version of the T5 model (Raffel et al., 2020). The *PeftConfig* is then inserted into the *get_peft_model* method, which creates and initializes the *PeftModel* based on the config. The config contains the information about the type of task (in our case seq-2-seq language model) and the type of parameter-efficient fine-tuning method (in our case prompt tuning or ATTEMPT). We also implement *save_pretrained*, *from_pretrained*, *forward* and *generate* methods. Since we have decided to implement prompt tuning from scratch, we built a parameter-efficient

[1] `https://github.com/DisAI-Replication-Challenge/ATTEMPT`

[2] `https://github.com/AkariAsai/ATTEMPT`

[3] We have also approached the authors to discuss this (and other) possible issues, but unfortunately, we did not receive an answer at the time of writing this report.

fine-tuning framework called **CPEFT** for **C**ustom **PEFT**, which is a custom remake of Huggingface parameter-efficient fine-tuning module (Mangrulkar et al., 2022) that we took inspiration from.

4.1 Prompt Tuning Implementation

Prompt tuning introduces a new variable to set the length of the prompt to be prepended to the input and a new variable to set the initialization of the prompt. The prompt can be initialized with random numbers, with random embeddings from the model vocabulary, and with single or multiple pre-trained prompt embeddings. During initialization, the prompt encoder Pytorch module is created and appended to the base model. The prompt encoder for the seq-2-seq language model is initialized with double the size of the prompt since the seq-2-seq architecture consists of two separate networks. This is the behavior presented in the Huggingface parameter-efficient fine-tuning module (Mangrulkar et al., 2022), but it does not match the implementation from the original prompt tuning paper (Lester et al., 2021). We have decided to use the Huggingface parameter-efficient fine-tuning behavior and just halve the size of the prompt encoder embeddings in configurations.

During the forward or generate of the model the method *get_prompt* is called. This method calls the forward function of the prompt encoder which returns the whole embedding matrix of the prompt encoder. This matrix is returned for each data in the batch and prepended to the input embeddings. After that, the result is inserted into the forward function of the base model. This implementation does not require to override of the original backward method or backpropagation calculation. During the saving and loading of pre-trained *PeftModel*, only the prompt encoder embeddings are saved, and loaded.

We train the source soft prompts individually for the SQuAD (Rajpurkar et al., 2016), SST-2 (Socher et al., 2013), QQP, QNLI (Wang et al., 2018), MNLI (Williams et al., 2018), and ReCoRD (Zhang et al., 2018) datasets. The training is set for 5 epochs and a single run with evaluation after each epoch. Weight decay for the AdamW optimizer is set to 1×10^{-5} with a linear scheduler with 500 warmup steps and a learning rate of 3×10^{-1}. The size of all soft prompts is 100. We use a maximum target length of 128 and a maximum input length of 512 for SQuAD and 256 for others.

4.2 ATTEMPT Implementation

The ATTEMPT method implementation includes an initialization of the prompt encoder with single or multiple pre-trained prompt embeddings, based on whether to train ATTEMPT in a single-task setting or multi-task setting. When initializing the ATTEMPT method, the prompt encoder and the attention module are created. The attention module consists of the sub-network module and the process of creating attention scores similar to the diagram in figure 4. The instance prompt is then created in the forward method of the *PeftModel* module.

The only difference in multi-task ATTEMPT is in the prompt encoder initialization and prompt fetching. While the single-task prompt embedding was just a single embedding, in the multi-task setting there is a *ModuleList* of embeddings for each task. Each embedding is then chosen based on the task IDs of the data. Similar to the prompt tuning, the instance prompt is then forwarded to the base model. During saving and loading of the model together with the prompt encoder embeddings also the attention module is saved and loaded.

We train ATTEMPT on 8 datasets from the GLUE (Wang et al., 2018) benchmark and 5 datasets from the SuperGLUE (Wang et al., 2019) benchmark. We train datasets over 10k samples for 10 epochs and the rest of the datasets for 20 epochs. We conduct 3 runs for each training configuration, initialize the target prompt embeddings with source prompts trained on the MNLI dataset, and use all of our trained soft prompts as source prompts. Weight decay for the AdamW optimizer is set to 1×10^{-2} with a linear scheduler with 500 warmup steps and a learning rate of 3×10^{-1}. The size of all soft prompts is 100. We use a maximum target length of 128 and a maximum input length of 348 for MultiRC and 256 for others. Another different setting from prompt tuning is that we pad the input to the maximum length of the T5 input token sequence.

The same settings are used in multi-task training except that we are using shared attention in every case. We are also not using a different learning rate for the attention sub-network and we are not using pre-trained weights for attention sub-network initialization.

dataset	SQuAD	SST-2	QNLI	MNLI	QQP	ReCoRD	avg.
Authors' soft prompts	31.7	63.7	92	62.9	92.3	82.9	70.9
Our soft prompts	68.8	95.4	95.5	84.6	94.2	82.1	86.8

Table 1: Evaluation of soft prompts provided by authors and our trained soft prompts. We have used accuracy for all of the datasets.

	GLUE									SuperGLUE					
dataset	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	avg.	Multi	BoolQ	WiC	WSC	CB	avg.
ATTEMPT single	**84.3**	**90.3**	93	93.2	89.7	85.7	73.4	57.4	83.4	**74.4**	**78.8**	**66.8**	53.8	78.6	70.5
ATTEMPT multi	83.7	90.1	**93.2**	**94.3**	**90.8**	**87.3**	**82.7**	64.3	**85.8**	**74.4**	78.5	66.5	**69.2**	82.1	**74.1**
Authors' soft prompts single	$72.6_{1.7}$	$\mathbf{90.3_{0}}$	$92.4_{0.3}$	$92.9_{0.2}$	$90_{0.5}$	$84.5_{2.3}$	$63.1_{1.1}$	$\mathbf{76.3_{5.9}}$	$82.8_{1.5}$	$48_{41.5}$	$70.2_{7.3}$	$60_{6.6}$	$64.7_{4.4}$	$67.9_{9.4}$	$62.2_{13.8}$
Our soft prompts single	$83.8_{0.2}$	$\mathbf{90.3_{0}}$	$92.7_{0.3}$	$89.4_{1.3}$	$90_{0.4}$	86.4_{2}	$74.8_{0.7}$	$72.8_{2.3}$	$85_{0.9}$	$71_{0.9}$	$75.1_{0.5}$	$57.8_{7.7}$	$66_{1.1}$	$77.4_{2.1}$	$69.5_{2.5}$
Authors' soft prompts multi	62.1_{7}	$87.7_{0.8}$	$90.4_{0.3}$	$91.1_{1.5}$	$89.6_{1.5}$	72.1_{1}	$47_{4.2}$	$69.4_{0.2}$	$76.2_{2.1}$	$71.7_{0.9}$	$68.9_{6.2}$	$59.6_{4.1}$	$34_{7.7}$	$65.5_{19.7}$	$59.9_{7.7}$
Our soft prompts multi	$83.5_{0.2}$	90_{0}	$92.4_{0.2}$	$90.3_{0.9}$	$90.1_{0.3}$	$81.7_{1.2}$	$73.6_{2.2}$	69.5_{0}	$83.9_{0.6}$	$68.2_{0.6}$	$75_{0.7}$	$51.3_{6.4}$	$56.4_{9.1}$	$\mathbf{84.5_{5.5}}$	$67.1_{4.5}$

Table 2: Test results of our ATTEMPT implementation compared to the results provided by authors. The results are calculated as a mean across 3 runs. We have used Pearson Correlation for STS-B, F1 macro for MultiRC (Multi), and accuracy for other datasets. The first two rows represent results provided by the authors in the ATTEMPT paper.

5 Experiments and Results

Since our replication study focuses mainly on replicating ATTEMPT results, we did not replicate the results provided by the prompt tuning authors; we only compared our results to source prompts provided by the ATTEMPT authors[4]. All of our experiment results and saved weights were documented in Weights & Biases projects, which are available online[5]. We are executing the experiments individually per configuration on a single Nvidia A10, A40, or A100. There is also a config file available for each of the set of experiments, we have created config files for prompt tuning, ATTEMPT single with authors' source prompts, ATTEMPT single with our source prompts, ATTEMPT multi with authors' source prompts, ATTEMPT multi with our source prompts. The ATTEMPT experiments set is multiplied by the number of dataset sets used.

5.1 Prompt Tuning

Better source prompt performance. Based on the results of source prompt training in Table 1, we can say that our source prompts are on average performing better than source prompts provided by authors. These results were not expected, as we followed the authors' hyperparameter settings and only trained the source prompts for 5 epochs. Since we trained the source prompts only for 1 run, we cannot determine stability across multiple runs.

The difference from the authors' results may be caused by the source prompt initialization from T5 vocabulary, which tends to increase instability as reported by ATTEMPT authors Asai et al. (2022). There may be also other randomness factors that we did not take into account, which may have caused the results to differ. Authors are also using their custom implementation of prompt which includes adapting and changing the original T5 code from the transformers library which may behave differently from our adapted CPEFT solution.

5.2 ATTEMPT

Better-performing source prompts over multi-task training. The results from ATTEMPT experiments in Table 2 show that the single-task method with our trained source soft prompt almost matched the authors' multi-task ATTEMPT results in average GLUE datasets score. This leads us to conclude that better-performing source prompts benefit the ATTEMPT performance. However, multi-task training splits the number of trained parameters over all trained tasks, which makes it more efficient compared to single-task training and more suitable for multi-task problems. Another observation is that with the increase of source prompts performance, the overall ATTEMPT performance also increases. This can mean that if the target prompt reaches a point in training in which it outperforms the source prompt attention interpolation the source prompts may start to hold back the target prompt. We can also see that better-performing source prompts tend to increase the stability of multi-task training.

[4] `https://homes.cs.washington.edu/~akari/models/attempt/source_prompts.zip`
[5] `https://github.com/DisAI-Replication-Challenge/ATTEMPT#experiment-results`

	GLUE									SuperGLUE					
dataset	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	avg.	Multi	BoolQ	WiC	WSC	CB	avg.
ATTEMPT single	84.3	90.3	93	93.2	89.7	85.7	73.4	57.4	83.4	**74.4**	**78.8**	66.8	53.8	78.6	70.5
ATTEMPT multi	83.7	90.1	93.2	94.3	**90.8**	**87.3**	**82.7**	64.3	85.8	**74.4**	78.5	66.5	**69.2**	82.1	**74.1**
Authors' soft prompts single	75.4	93.9	94.6	**95.5**	88.9	86.3	75.3	**77.2**	85.9	68.6	77.3	**67.4**	59.6	78.6	70.3
Our soft prompts single	84.6	**94.3**	**95.4**	93.4	89.5	87.2	81.9	75.6	**87.7**	74.2	76.8	65.8	67.3	82.1	73.2
Authors' soft prompts multi	75.3	93.8	93.5	95.4	88.9	80.9	60.1	68.7	82.1	68.9	75	60.2	59.6	78.6	68.5
Our soft prompts multi	**84.8**	94.2	92.5	87.1	88.6	78.9	77.5	68.7	84	69.4	76	64.3	63.5	**85.7**	71.8

Table 3: Cherry picked results of our ATTEMPT implementation – best validation results over all runs. We have used Pearson Correlation for STS-B, F1 macro for MultiRC (Multi), and accuracy for other datasets. The first two rows represent results provided by the authors in the ATTEMPT paper.

Stability problems across multiple runs. We have noticed training instability across multiple runs of ATTEMPT, especially in smaller-size datasets (less than 10k samples). The instability may be caused by the random weight initialization and since we did not use the seed for the weight (only for dataset shuffle) of the attention module, randomness factors may be another reason why our results differ from the authors' results. Since the ATTEMPT authors did not provide any information about stability, we have chosen to select also the best validation results across all of the runs to see how the results shift. These results can be seen in Table 3 and are called cherry-picked results. We can see that cherry-picked results can increase the overall GLUE and SuperGLUE score of our ATTEMPT implementation, but these results do not say anything about the true ATTEMPT performance.

The need for pre-trained attention of multi-task ATTEMPT. We were not able to match the results of multi-task ATTEMPT, but we suspect that one of the reasons why our implementation underperformed the authors' multi-task ATTEMPT implementation is the lack of pre-training of the attention module. We were not able to retrieve more information about the pre-training of the attention module from the ATTEMPT paper, therefore we have decided to not pre-train the attention module. We also did not set a separate learning rate for the attention module sub-network, which may be another cause of why we ended up with different results.

Overall ATTEMPT Results. Our experiments with single-task ATTEMPT achieved a better average GLUE benchmark score than the results reported in the ATTEMPT paper by ATTEMPT authors and almost matched the SuperGLUE benchmark scores. The multi-task ATTEMPT experiments did not achieve better results on both benchmarks and our multi-task ATTEMPT results are lower than the single-task ATTEMPT results. We suspect that the requirement of the attention module may be crucial for yielding better results for the multi-task training, since it may be harder for the attention module to adapt for multiple tasks from scratch. Another reason for not achieving the exact results as provided by the ATTEMPT authors may be the randomness factors. Our prompt initialization, data splits, and even the training environment (i.e. GPU, Python modules versions) were not necessarily the same, which may have caused differences in training.

6 Conclusion

In our replication study, we have successfully replicated the parameter-efficient fine-tuning method presented in the paper ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts (Asai et al., 2022). Based on the results from conducted experiments, we have identified that better-performing source prompts in single-task ATTEMPT training achieve on average better results even when compared to multi-task training. We also discuss the stability problems that we have faced during ATTEMPT training and the possible need for pre-training of the attention module for multi-task ATTEMPT training.

Furthermore, we would like to conduct extended experiments with ATTEMPT and investigate how dataset size and number of trained source prompts affect the performance of ATTEMPT. At the same time, we would like to investigate the transferability of source prompts trained on tasks in multiple languages for multi-lingual tasks. Lastly, we would like to look at the architecture of ATTEMPT and its attention module to investigate, whether there are other ways how to look at attentional task transferability, like replacing the max pooling with another transformation that retains more information.

Acknowledgment

This research was partially supported by DisAI, a project funded by Horizon Europe under GA No.101079164. Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

Akari Asai, Mohammadreza Salehi, Matthew Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6655–6672, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Alexandra Chronopoulou, Matthew Peters, Alexander Fraser, and Jesse Dodge. 2023. AdapterSoup: Weight averaging to improve generalization of pretrained language models. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2054–2063, Dubrovnik, Croatia. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11.

Gamaleldin F. Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. 2019. Adversarial reprogramming of neural networks. In *International Conference on Learning Representations*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8410–8423, Dublin, Ireland. Association for Computational Linguistics.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. 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)*, pages 4921–4933, Online. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

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

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

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Quoc Le, Tamás Sarlós, and Alexander Smola. 2013. Fastfood-computing hilbert space expansions in loglinear time. In *International Conference on Machine Learning*, pages 244–252. PMLR.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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)*, pages 4582–4597, Online. Association for Computational Linguistics.

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

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 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)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. 2023. Multitask prompt tuning enables parameter-efficient transfer learning. *arXiv preprint arXiv:2303.02861*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 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)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Ping Yu, Wei Wang, Chunyuan Li, Ruiyi Zhang, Zhanpeng Jin, and Changyou Chen. 2022. Stt: Soft template tuning for few-shot adaptation. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 941–946. IEEE.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Logically at Factify 2: A Multi-modal Fact Checking System Based on Evidence Retrieval Techniques and Transformer Encoder Architecture: A Replication Study

Ivana Beňová[1,2] and **Stefanos-Iordanis Papadopoulos**[3]
[1] Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic
[2] Kempelen Institute of Intelligent Technologies, Bratislava, Slovakia
[3] Information Technology Institute, Centre for Research & Technology, Hellas
ivana.benova@kinit.sk, stefpapad@iti.gr

Abstract

This paper presents a replication study conducted on two multimodal fact-checking models: a baseline model and the Logically team's model, as part of the Defactify 2 Workshop at AAAI 2023. The replication process involved detailed reimplementation of the model architectures, training procedures, and evaluation methodologies described in the original papers' results. Our results closely align with the reported outcomes, validating the robustness of the models, with only minor discrepancies between the replicated and original results, attributed to factors such as randomness and nuanced variations in model configurations. The Logically team's original result was 78.97% (weighted F1 score on the test set), compared to our replication's 77.96%. Similarly, the baseline model achieved a reported result of 64.99%, while our replication yielded 63.37%. Our study underscores the significance of open science practices for fostering reproducibility and progress in the field of multimodal entailment research. We provide the replicated code for both Baseline and team Logically on GitHub, making it accessible to researchers and practitioners worldwide. The GitHub repository containing the code can be found at https://github.com/ivana-13/DisAI_replication_challenge_2024.

1 Introduction

Within the landscape of artificial intelligence and language technologies, the Kempelen Institute of Intelligent Technologies (KInIT), in collaboration with DFKI, the University of Copenhagen, and CERTH, is deeply engaged in the DisAI project. This collaborative effort aims to elevate the scientific excellence of KInIT in AI and language technologies, with a specific focus on combatting disinformation. Recognizing the societal challenges posed by misinformation and the need for scientific advancement in Slovakia's research and innovation ecosystem, the DisAI project concentrates on three core areas: Multilingual Language Technologies, Multimodal Natural Language Processing, and Trustworthy Artificial Intelligence.

As part of the DisAI project, KInIT has initiated a Replication Challenge, offering an invaluable opportunity for early-stage researchers to collaborate with mentors from leading research institutions. This challenge revolves around replicating existing research in multilingual language technologies, multimodal natural language processing, and trustworthy artificial intelligence, with a primary emphasis on combating disinformation.

In this Replication Challenge, our objective is to reproduce a selected approach presented at the Defactify 2 Workshop (Suryavardan et al., 2023b), (Suryavardan et al., 2023a). The Defactify 2 Workshop, a multimodal fact-checking workshop held at AAAI'23, convened researchers and practitioners to address the escalating issue of fake news. The workshop introduced a multimodal fact verification news dataset called Factify 2, which attracted over 60 participants and yielded nine final test-set submissions.

The approach presented by the Logically team, securing the third position in the Defactify 2 Workshop challenge, delineates a multimodal fact-checking system grounded in evidence retrieval techniques followed by two identical but independent unimodal cross-modal Transformer Encoders. Responding to the urgent need for automated fact-checking systems, this approach highlights the potential of multimodal veracity prediction, leveraging both textual and visual inputs to predict a claim's truthfulness.

Our choice to replicate the approach presented by the Logically team arises from its innovative methodology, the unavailability of its code for public access, and the promising results it achieved in the Defactify 2 Workshop challenge.

This Replication Challenge aligns with the broader objectives of the DisAI project, underscoring the significance of scientific excellence, collaboration, and the practical application of research in addressing societal challenges related to disinformation. In the subsequent sections, we delve into the intricacies of the Defactify 2 Workshop, the Logically team's approach, our replication strategy, and the anticipated contributions to the field of multimodal fact-checking.

2 Related Work

In the pursuit of advancing fact-checking methodologies, researchers have explored various avenues, from unimodal to multimodal approaches, to discern the veracity of claims effectively. Here, we provide an overview of related work in three key dimensions: text-based datasets, multimodal datasets, and modeling approaches, drawing insights from the literature surrounding the Defactify 2 Workshop, the approach presented by the Logically team, and relevant prior research.

Text-based Datasets: Numerous datasets have emerged in recent years to facilitate fact-checking in text-based domains. Notable datasets include LIAR (Wang, 2017), FEVER (Thorne et al., 2018), or COVID-19 dataset (Patwa et al., 2021b), each offering diverse claims and supporting documents for fact-checking purposes.

Multimodal Datasets: Recognizing the limitations of text-only datasets, the research community has shifted towards embracing multimodal datasets that incorporate textual, visual, and sometimes even temporal data. Datasets like fakeddit (Nakamura et al., 2019), FakeNewsNet (Shu et al., 2020), and MOCHEG (Yao et al., 2023) provide a rich source of multimodal instances for fake news detection. Methods to tackle this problem have been proposed in studies by Wu et al. (2021), Jing et al. (2023), Hua et al. (2023), Yadav et al. (2023). For a detailed survey on multimodal fake news detection, please refer to (Alam et al., 2021).

Building on the successes of previous iterations, the Factify 1 dataset (Mishra et al., 2022) at AAAI 2022 served as a multimodal fact-verification dataset, comprising 50k instances categorized into Support, Insufficient, and Refute. Moving forward, Factify 2 at AAAI 2023 extended this dataset with additional instances, introducing data from satirical articles.

Modeling approaches: The text-based datasets employ a range of methods, including CNNs (Saleh et al., 2021), RNNs (Ajao et al., 2018; Sunagar and Kanavalli, 2022), and BERT-based models (Kaliyar et al., 2021; Patwa et al., 2021a; Glazkova et al., 2021), to detect and verify text-based fake news.

In the Defactify 1 Workshop, researchers employed methods like BERT (Dhankar et al., 2022), RoBERTa (Zhuang and Zhang, 2022), and BigBird (Gao et al., 2021) for textual features, while visual features were extracted using ResNet (Gao et al., 2021), DeiT (Wang and Peng, 2022), EfficientNet (Hulke et al., 2021), and VGG (Zhuang and Zhang, 2022). Please refer to (Patwa et al., 2022) for details of all the methods.

In Defactify 2 Workshop, the participants utilized various techniques for text embeddings, including DeBERTa (He et al., 2020), CLIP (Radford et al., 2021), S-BERT (Reimers and Gurevych, 2019), ROUGE (Lin, 2004) and Word2Vec. The image embeddings were extracted through Swinv2 (Liu et al., 2022), ResNet, CLIP (Radford et al., 2021), ViT and DeiT (Wang and Peng, 2022). The first and second best approaches Triple-Check (Du et al., 2023) and INO (Zhang et al., 2023), provided their code publicly. The lack of publicly available code for the third best approach at Defactify 2 Workshop is one of the main reasons why we decided to replicate Logically (Verschuuren et al., 2023).

3 Task Description

In the Factify challenge, the task revolves around detecting multimodal fake news, particularly in verifying the authenticity of claims. This task is framed as multimodal entailment, where both text and image contribute to evaluating a claim's truthfulness. The aim is to determine if a given claim and image align with information from a reliable source, termed the "document". This approach recognizes the complexity of fact-checking, which necessitates a holistic assessment of textual and visual content.

The terms "claim" and "document" denote the entities under scrutiny. A claim usually represents a short public statement or assertion that requires verification. In data collection, claims are gathered from social media, particularly tweets from Twitter, where users express unsupported statements.

On the other hand, a document serves as a credible source of information, typically derived from full articles that cover diverse topics and contexts. Documents, in the form of full articles, are chosen as reliable sources to cross-reference and verify the veracity of the claims.

In this context, veracity prediction refers to the system's capability to predict the truthfulness of a claim, considering both textual and visual content. Multimodal entailment involves evaluating the veracity of a claim by comparing it to a reliable source, utilizing both textual and visual information.

Each data point consists of a claim and its associated document containing textual and visual information. The task involves determining if the document entails the claim. The entailment between the four data sources, namely claim image, claim text, document image, and document text, is used to define the categories that the data are classified into. This is also shown in Figure 1.

The system classifies each data sample into one of the five categories:

- **Support_Text:** Textual data of the claim is entailed by the textual data of document, but their images are not entailed.
- **Support_Multimodal:** Both textual data and image of the claim are entailed by textual data and image of the document.
- **Insufficient_Text:** Textual data is not entailed, but there may be common words, and the images are not entailed.
- **Insufficient_Multimodal:** Textual data is not entailed, but common words may exist, and the images are entailed.
- **Refute:** Refute: Both textual and visual information from the document contradict or refute the claim, indicating that the claim is false.

Category	Text	Image
Support_Multimodal Figure 1a	Text is supported, Similar News	Image is supported
Support_Text Figure 1b	Text is supported, Similar News	Image is neither supported nor refuted
Insufficient_Multimodal Figure 1c	Text is neither supported nor refuted, May have common words	Image is supported
Insufficient_Text Figure 1d	Text is neither supported nor refuted, May have common words	Image is neither supported nor refuted
Refute Figure 1e	Fake Claim	Image is refuted

Table 1: This table shows the categories the dataset has been divided into and the relationship between the multimodal claim and document in each class.

4 Dataset

The Factify 2 dataset maintains the same categories as Factify 1, comprising 50,000 data samples. These samples are evenly distributed among five categories (explained in Section 3), with a split of 70:15:15 into train, validation, and test sets, respectively. The Factify 2 dataset comprises claim-document pairs gathered from diverse sources, including Twitter, fact-checking websites, and satirical news websites. Each pair includes a claim and a document featuring image, text, and OCR text extracted from images. This dataset is a comprehensive resource for advancing multimodal fact-checking methodologies, annotated with labels such as Support Multimodal, Support Text, Refute, Insufficient Multimodal, or Insufficient Text.

4.1 Dataset Collection

The Factify 2 dataset was curated through a dual-pipeline collection process, distinguishing between real and fake news articles. The primary objective was to assemble a comprehensive dataset encompassing textual and visual elements for claims and their corresponding supporting or disproving documents.

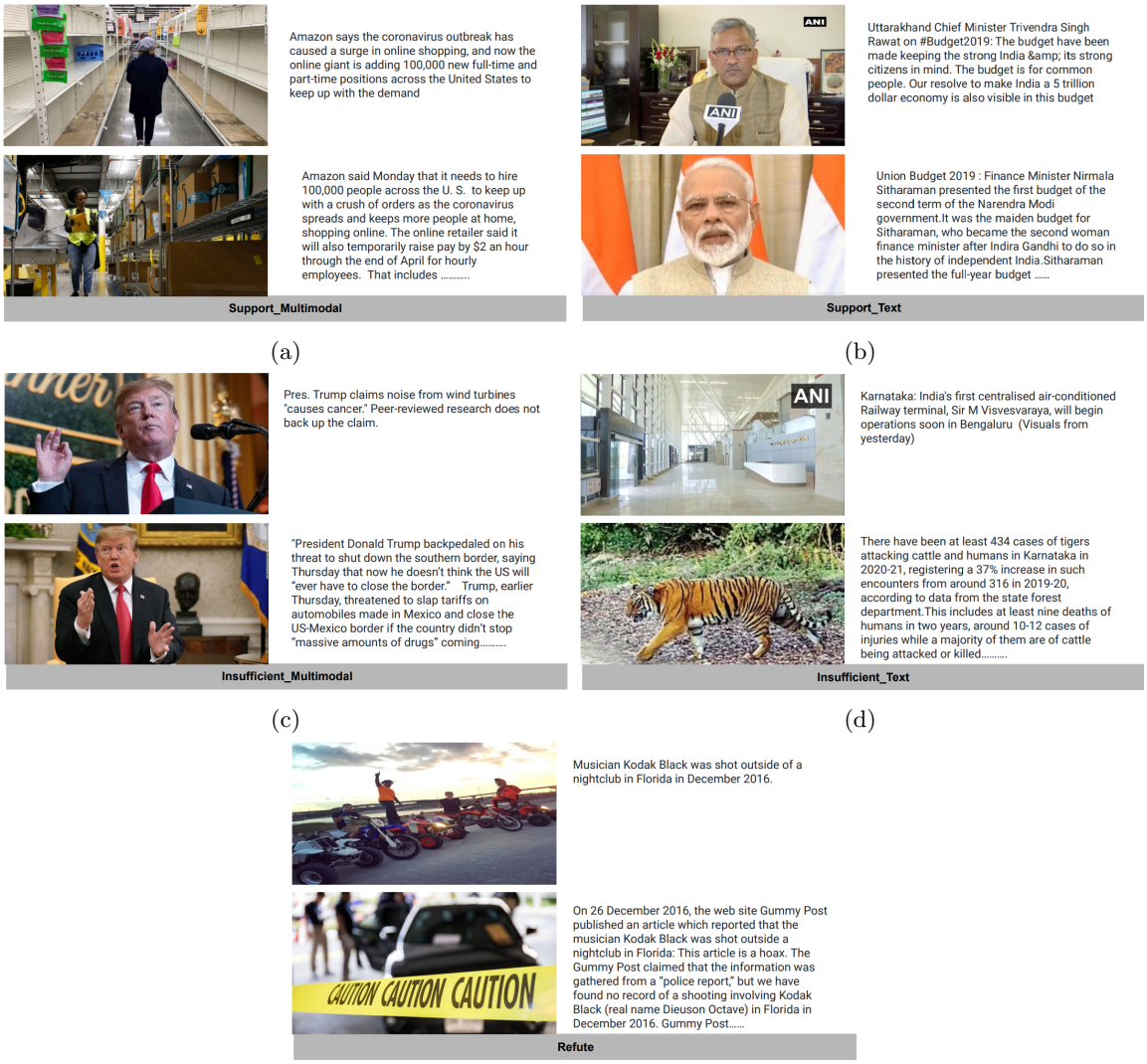

Figure 1: Examples for all the five categories. The upper text and image are the claims and claim image, and the bottom image text and image resemble the document and document image.

Real News Collection (Support and Insufficient Evidence):

- Leveraging the methodology from Factify 1, tweets from reputable news handles such as Hindustan Times[1], ANI[2], ABC[3], and CNN[4] were collected date-wise.
- For each tweet from a news handle (account A), a corresponding tweet from another Twitter account (account B) was selected. These tweets were then compared using Sentence BERT along with a specified threshold to determine if they reported the same news.
- If the tweets are not the same, they are compared for common words using the NLTK library to categorize the tweets as similar or dissimilar.
- Two image similarity metrics, namely cosine similarity of ResNet50 embeddings and histogram similarity, were used to further categorize data based on visual entailment.
- With this collected data, the tweet from one handle (for example, A) is treated as the claim, and the news article associated with the tweet from the other handle (in this example, B) as the supporting document.

[1] https://twitter.com/htTweets
[2] https://twitter.com/ANI
[3] https://twitter.com/ABC
[4] https://twitter.com/CNN

Fake News Collection (Refute):

- Data for the refute category was sourced from fact-checking websites, including Snopes [5], Factly[6], and Boom [7]. These websites provided well-defined claims and documents disproving them.
- In this iteration, satirical articles were introduced, collected from websites like Fauxy[8] and Empire-News[9]. Despite explicitly stating their non-truthfulness, these articles were categorized as support, given their supportive nature to the false claim.
- Images were scraped by searching for article headlines, and manual annotation was conducted to augment data for the refute category.

5 Replicated Methods

5.1 Baseline

To better position the approach proposed by team Logically on Defactify 2 Workshop, we undertook the task of replicating the simple baseline model crafted by Suryavardan et al. (2023b). Recognizing the widespread use of diverse media in online information dissemination, the authors highlighted the crucial consideration of images and text to ensure accurate claim classification, particularly in the context of potential misrepresentation and the propagation of misinformation.

The baseline model follows an entailment-based approach, requiring extracting features from claim and document image-text pairs. Visual features are derived using a pre-trained Vision Transformer model (ViT). To capture textual nuances, the model employs a pre-trained SentenceBERT model. This model generates sentence embeddings for both claim and document attributes. The SentenceBERT embeddings are then concatenated with the pooled output from the ViT model, creating a fused representation of visual and textual features. These combined features are subsequently processed through a classification layer consisting of a Multilayer Perceptron (MLP) layer with 512 nodes, batch normalization, ReLU activation function, 0.5 dropouts, and another NLP layer with five nodes. The overall architecture of the model is illustrated in Figure 2, depicting the sequential flow of operations from feature extraction to classification.

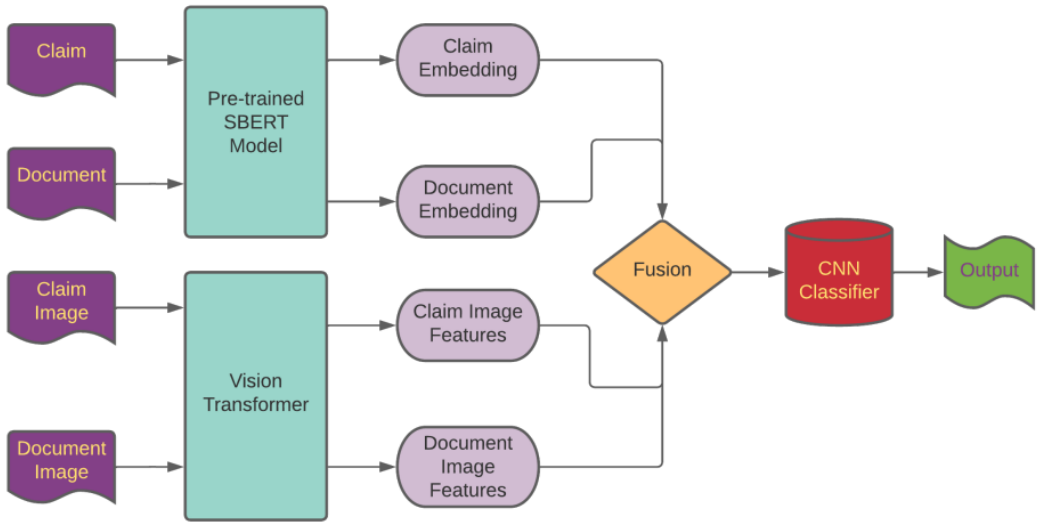

Figure 2: Baseline model architecture (Suryavardan et al., 2023b). Text, image features extracted from the document, and the claim are concatenated and used for final prediction.

[5] https://www.snopes.com/
[6] https://factly.in/category/english/
[7] https://www.boomlive.in/fact-check
[8] https://thefauxy.com/
[9] https://empirenews.net/

This replication of the baseline model serves as a benchmark for evaluating the effectiveness of future enhancements' effectiveness and provides a consistent starting point for researchers and practitioners engaging with the Factify 2 dataset.

	Support Text	Support Multimodal	Insufficient Text	Insufficient Multimodal	Refute	Final
Baseline paper	50%	82.72%	80.24%	75.93%	98.82%	64.99%
Replication	56.73%	79.90%	76.64%	72.29%	96.87%	63.37%

Table 2: Category-wise and overall weighted average F1 score for baseline model on a test set of Factify 2 reported in (Suryavardan et al., 2023a).

5.2 Logically

The system architecture adopts a standard two-stage claim verification approach. Initially, a textual evidence retrieval component identifies relevant evidence passages from the document. Subsequently, two independent transformer-based encoders for cross-modal input (incorporating evidence passages text, claim text, claim image, document image, claim OCR text, and document OCR text) and unimodal input (incorporating evidence passages text and claim text) are used to predict the five multimodal entailment categories.

The architecture utilizes a pre-trained cross-modal model CLIP and a pre-trained text embedding model Word2Vec for cross-modal matching. Employing a list-wise concatenation strategy, the model aims to capture both unified-multimodal and unimodal representations.

5.2.1 Evidence Retrieval

In evidence retrieval, the 'multi-qa-mpnet-base-dot-v1' model computes claim and document text embeddings at the passage level. Based on S-BERT and the MPNet architecture, the model is trained on a QA dataset and encodes text into a 768-dimensional vector. The top K passages obtained from semantic search are re-ranked based on relevancy to the claim text and concatenated to one text.

The reason for this first stage of the model is the length of the document. The Factify 2 dataset was created so that documents are complete, very long news articles. The average length of words per class can be seen in Figure 3. Encoding such a long text with a CLIP encoder (which takes a maximum of 77 tokens) could lead to a loss of information and context and embedding of the tokens, which are unimportant for veracity prediction. Even using a different encoder that can encode more tokens could lead to a loss of information, as such a long text is supposed to be embedded. On the other hand, encoding each token of such a document with the Word2Vec model would lead to enormous representation.

5.2.2 Feature extraction

The embedding layer comprises a cross-modal encoder and an unimodal text encoder. This architecture leverages text-to-text and cross-modal interactions, enhancing multimodal semantic relatedness. The cross-modal encoder utilizes a pre-trained CLIP model (a ViT-B/32 variant). It encodes text inputs (claim text, evidentiary passage, and two images OCR text) and image inputs (claim image and document image), respectively, which are then concatenated into a 6×512 matrix as a single input to the subsequent transformer encoder. The pre-trained Word2vec model (a Google News 300 variant) is adopted as an unimodal text encoder. It encodes the concatenated text sequence of claim and document evidentiary passage text and obtains a 300-dimensional feature vector for each token. Zero-padding is applied to match the longest sentence in the training set. Neither the pre-trained CLIP nor Word2Vec embedding models were fine-tuned.

5.2.3 Cross-modal Veracity Prediction

The veracity prediction component relies on two Transformer Encoders (TE) with self-attention mechanisms. Cross-modal and unimodal inputs are processed through separate transformer encoders, and the outputs are concatenated before passing through an MLP classifier for five-category prediction. The classifier consists of 3 MLP layers, and the number of nodes per layer is set to 3072, 1024, and 5, respectively. A dropout and ReLU activations are applied between the MLP layers.

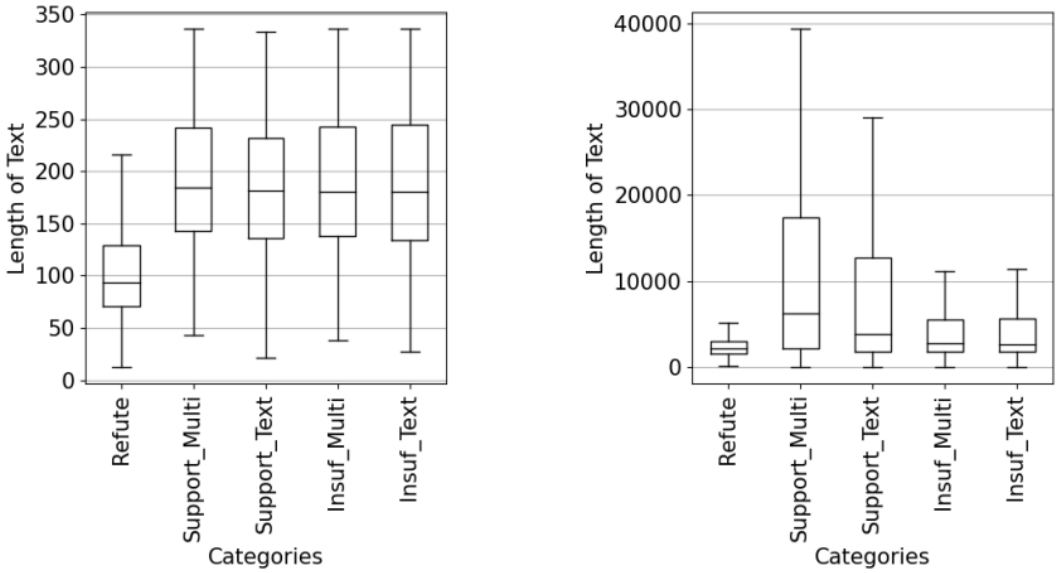

Figure 3: Boxplots of text length distribution of all categories in claim text (left) and document text (right) (Verschuuren et al., 2023).

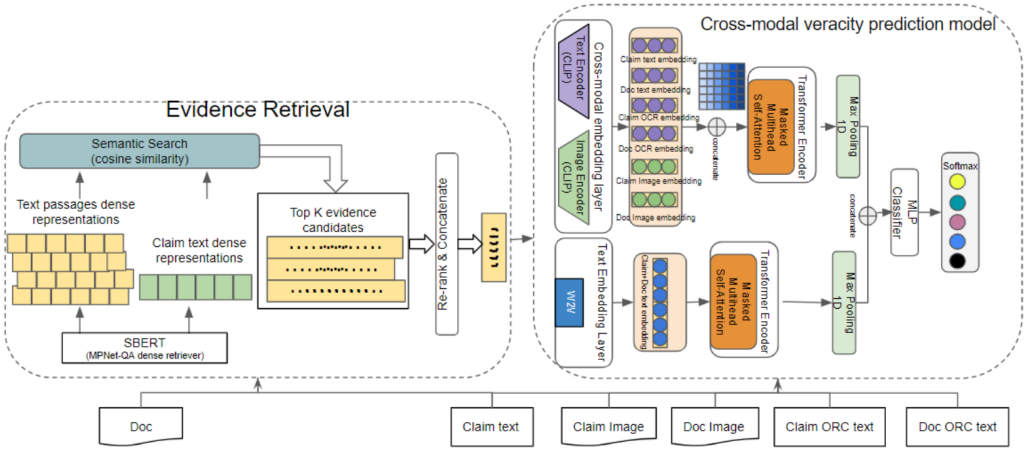

Figure 4: Logically General System Architecture (Verschuuren et al., 2023).

5.2.4 Implementation Details

Experiments in the paper include variations in evidence retrieval, passage length, passage ranking granularity, and alignment strategies with SBERT and CLIP. The best approach, presented on the leaderboard with a test set of Factify 2, consists of the top 5 sentences sorted by the SBERT-QA model and selected as evidentiary passages in this setting. For Word2Vec embeddings, the longest sentence in the training set has 638 tokens. Two transformer encoders are employed with an empirical setting of four heads in two multi-head attention blocks. No information was given in the paper about the dimension of a feed-forward layer or dropout or activation layer for the transformer, so we assumed the default values for the Transformer encoder used in pytorch implementation (2048 feed-forward dimension, 0.1 dropout, ReLU activation). According to the paper, the model was trained up to 80 epochs with early stopping on minimum validation loss by minimizing the cross-entropy loss function. The adaptive AdamW optimizer with an initial learning rate of $\gamma = 1 \times 10^{-4}$ and epsilon $\varepsilon = 1 \times 10^{-8}$ was used for optimization. The batch size was $N_{\text{batch}} = 16$. Early stopping patience was set to 5. A linear decreasing learning rate scheduler was used, including the first $N_{\text{steps}} = 438$ warming-up training steps, during which the learning rate increased linearly to the chosen learning rate.

It was mentioned in the paper that the presented results were obtained after 20 epochs of training. Our replicated results were obtained after eight epochs of training. The evaluations in the table below were done using random seed 42. The experiments were run on a local Linux server (Ubuntu 20.04.3 LTS) with 4 NVIDIA RTX 3090 GPUs, AMD Ryzen Threadripper 3970X 32-Core CPU, and 128GB DDR4. From the available resources, we used 1 GPU. The experiment had been run one more time with a seed 49 and the average F1 score for the dataset was 77.60 after 9 epochs. This suggests that the model is robust.

	Support Text	Support Multimodal	Insufficient Text	Insufficient Multimodal	Refute	Final
Logically paper	80.38%	90.51%	84.39%	85.63%	98.51%	78.97%
Replication	83.00%	90.19%	81.98%	81.89%	98.02%	77.96%

Table 3: Category-wise and overall weighted average F1 score for baseline model on test set of Factify 2 reported in (Suryavardan et al., 2023a).

6 Insights and Discussion

In this study, we successfully replicated both the baseline and Logically team's models, obtaining results that closely align with the reported outcomes in the respective papers. The baseline model yielded a result of 63.37%, which closely resembles the reported 64.99%, while the replication of the Logically team's model produced a result of 77.96%, showing a high degree of similarity to the reported 78.97%. Any minor discrepancies observed can be attributed to randomness, as the papers did not specify the random seed used during training, and nuanced variations in the model configurations left unmentioned in the papers.

Our successful replication underscores the robustness of these models and highlights the importance of transparency in research. The availability of code for the Logically team's model enabled further exploration and comparison, fostering a collaborative research environment.

Looking ahead, with access to the Logically team's code, we aim to contribute to the scientific community by publishing our findings. This will reinforce the reproducibility of results and provide a deeper understanding of the intricacies involved in the multimodal entailment task. Our study emphasizes the significance of open science practices, ensuring that research findings can be scrutinized, replicated, and built upon to advance knowledge in the field.

In conclusion, this endeavor serves as a testament to the collaborative nature of scientific inquiry, emphasizing the importance of code availability and transparent reporting for fostering reproducibility and progress in multimodal entailment research.

References

Oluwaseun Ajao, Deepayan Bhowmik, and Shahrzad Zargari. 2018. Fake news identification on twitter with hybrid cnn and rnn models. In *Proceedings of the 9th international conference on social media and society*, pages 226–230.

Firoj Alam, Stefano Cresci, Tanmoy Chakraborty, Fabrizio Silvestri, Dimiter Dimitrov, Giovanni Da San Martino, Shaden Shaar, Hamed Firooz, and Preslav Nakov. 2021. A survey on multimodal disinformation detection. *arXiv preprint arXiv:2103.12541*.

Abhishek Dhankar, O Zaiane, and Francois Bolduc. 2022. Uofa-truth at factify 2022: A simple approach to multi-modal fact-checking. In *Proceedings of De-Factify: Workshop on Multimodal Fact Checking and Hate Speech Detection*.

Wei-Wei Du, Hong-Wei Wu, Wei-Yao Wang, and Wen-Chih Peng. 2023. Team triple-check at factify 2: Parameter-efficient large foundation models with feature representations for multi-modal fact verification. *arXiv preprint arXiv:2302.07740*.

Jie Gao, Hella-Franziska Hoffmann, Stylianos Oikonomou, David Kiskovski, and Anil Bandhakavi. 2021. Logically at factify 2022: Multimodal fact verification. *arXiv preprint arXiv:2112.09253*.

Anna Glazkova, Maksim Glazkov, and Timofey Trifonov. 2021. g2tmn at constraint@ aaai2021: exploiting ct-bert and ensembling learning for covid-19 fake news detection. In *International Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 116–127. Springer.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Jiaheng Hua, Xiaodong Cui, Xianghua Li, Keke Tang, and Peican Zhu. 2023. Multimodal fake news detection through data augmentation-based contrastive learning. *Applied Soft Computing*, 136:110125.

Nainesh Hulke, Bharath Raj Siva, Ankesh Raj, and Ali Asgar Saifee. 2021. Tyche at factify 2022: Fusion networks for multi-modal fact-checking.

Jing Jing, Hongchen Wu, Jie Sun, Xiaochang Fang, and Huaxiang Zhang. 2023. Multimodal fake news detection via progressive fusion networks. *Information processing & management*, 60(1):103120.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. Fakebert: Fake news detection in social media with a bert-based deep learning approach. *Multimedia tools and applications*, 80(8):11765–11788.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. 2022. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019.

Shreyash Mishra, S Suryavardan, Amrit Bhaskar, Parul Chopra, Aishwarya Reganti, Parth Patwa, Amitava Das, Tanmoy Chakraborty, Amit Sheth, Asif Ekbal, et al. 2022. Factify: A multi-modal fact verification dataset. In *Proceedings of the First Workshop on Multimodal Fact-Checking and Hate Speech Detection (DE-FACTIFY)*.

Kai Nakamura, Sharon Levy, and William Yang Wang. 2019. r/fakeddit: A new multimodal benchmark dataset for fine-grained fake news detection. *arXiv preprint arXiv:1911.03854*.

Parth Patwa, Mohit Bhardwaj, Vineeth Guptha, Gitanjali Kumari, Shivam Sharma, Srinivas Pykl, Amitava Das, Asif Ekbal, Md Shad Akhtar, and Tanmoy Chakraborty. 2021a. Overview of constraint 2021 shared tasks: Detecting english covid-19 fake news and hindi hostile posts. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1*, pages 42–53. Springer.

Parth Patwa, Shreyash Mishra, S Suryavardan, Amrit Bhaskar, Parul Chopra, Aishwarya Reganti, Amitava Das, Tanmoy Chakraborty, Amit Sheth, Asif Ekbal, et al. 2022. Benchmarking multi-modal entailment for fact verification. In *Proceedings of De-Factify: Workshop on Multimodal Fact Checking and Hate Speech Detection, CEUR*.

Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. 2021b. Fighting an infodemic: Covid-19 fake news dataset. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1*, pages 21–29. Springer.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Hager Saleh, Abdullah Alharbi, and Saeed Hamood Alsamhi. 2021. Opcnn-fake: Optimized convolutional neural network for fake news detection. *IEEE Access*, 9:129471–129489.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big data*, 8(3):171–188.

Pramod Sunagar and Anita Kanavalli. 2022. A hybrid rnn based deep learning approach for text classification. *International Journal of Advanced Computer Science and Applications*, 13(6).

S Suryavardan, Shreyash Mishra, Megha Chakraborty, Parth Patwa, Anku Rani, Aman Chadha, Aishwarya Reganti, Amitava Das, Amit Sheth, Manoj Chinnakotla, et al. 2023a. Findings of factify 2: multimodal fake news detection. *arXiv preprint arXiv:2307.10475*.

S Suryavardan, Shreyash Mishra, Parth Patwa, Megha Chakraborty, Anku Rani, Aishwarya Reganti, Aman Chadha, Amitava Das, Amit Sheth, Manoj Chinnakotla, et al. 2023b. Factify 2: A multimodal fake news and satire news dataset. *arXiv preprint arXiv:2304.03897*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

P Jordi Verschuuren, Jie Gao, Adelize Van Eeden, Stylianos Oikonomou, and Anil Bandhakavi. 2023. Logically at factify 2: A multi-modal fact checking system based on evidence retrieval techniques and transformer encoder architecture.

Wei-Yao Wang and Wen-Chih Peng. 2022. Team yao at factify 2022: Utilizing pre-trained models and co-attention networks for multi-modal fact verification. *arXiv preprint arXiv:2201.11664*.

William Yang Wang. 2017. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.

Yang Wu, Pengwei Zhan, Yunjian Zhang, Liming Wang, and Zhen Xu. 2021. Multimodal fusion with co-attention networks for fake news detection. In *Findings of the association for computational linguistics: ACL-IJCNLP 2021*, pages 2560–2569.

Ashima Yadav, Shivani Gaba, Haneef Khan, Ishan Budhiraja, Akansha Singh, and Krishna Kant Singh. 2023. Etma: Efficient transformer-based multilevel attention framework for multimodal fake news detection. *IEEE Transactions on Computational Social Systems*.

Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. 2023. End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2733–2743.

Yinuo Zhang, Zhulin Tao, Xi Wang, and Tongyue Wang. 2023. Ino at factify 2: Structure coherence based multi-modal fact verification. *arXiv preprint arXiv:2303.01510*.

Yan Zhuang and Yanru Zhang. 2022. Yet at factify 2022: Unimodal and bimodal roberta-based models for fact checking. In *Proceedings of De-Factify: Workshop on Multimodal Fact Checking and Hate Speech Detection, CEUR*.

SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer: A Replication Study

Ivan Vykopal[1,2], **Simon Ostermann**[3] and **Marián Šimko**[2]
[1] Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic
[2] Kempelen Institute of Intelligent Technologies, Bratislava, Slovakia
[3] German Research Institute for Artificial Intelligence (DFKI),
Saarland Informatics Campus, Germany
ivan.vykopal@kinit.sk, simon.ostermann@dfki.de, marian.simko@kinit.sk

Abstract

Efficient model training approaches that enable task-specific learning without training the entire model have been the focus of multiple works. One such effort is the development of PromptTuning, a technique designed to train soft prompts for individual tasks. By extending the PromptTuning paper, a Soft Prompt Transfer approach was developed that exploits the ability to transfer knowledge between tasks. This study replicates the methods outlined in the SPoT paper (Vu et al., 2022). Our investigation reveals the positive impact of inter-task prompt transfer on downstream tasks, demonstrating improved performance compared to the original PromptTuning. To assess the broader applicability of the approach, an analysis of impacts is conducted using four source tasks and three target tasks. This analysis not only measures the efficiency of knowledge transfer between various tasks but also evaluates the adaptability of the method across high and low-resource settings.

1 Introduction

In recent years, there has been a rapid development of large language models (LLMs), characterized by an increase in size, requiring significant computational resources for training. Their model capacity has accompanied this growth in model size and enhances their ability to solve many natural language processing (NLP) tasks. Most models have achieved state-of-the-art results on various NLP benchmarks, prompting the emergence of diverse methodologies, such as efficient fine-tuning aimed at continual improvement.

Emerging models are not exclusively developed for addressing a single task but demonstrate the capability to tackle a broader spectrum of tasks (Chung et al., 2022). These tasks include question answering, natural language inference (NLI), sentiment analysis, etc. Furthermore, these models can benefit from additional fine-tuning on diverse NLP benchmarks, highlighting the importance of parameter-efficient fine-tuning methods, which facilitate fine-tuning with fewer trainable parameters and, as a result, reduce computational demands.

A specific category of the parameter efficient fine-tuning (PEFT) methods involves fine-tuning through soft prompts, in which additional trainable parameters are incorporated into the model, and only these parameters are further trained, leaving the rest of the model frozen. Soft prompts are represented by additional embeddings added to the model, used for improving the model performance. These approaches achieve comparable results with minimal increase in trainable parameters. An example is PromptTuning (Lester et al., 2021), in which a task-oriented prompt is added for each downstream task, and only these prompts are fine-tuned.

Our study focuses on applying PromptTuning for fine-tuning the T5 language model and analyzing its impact on the GLUE and SuperGLUE benchmarks. We aim to replicate the results reported by Vu et al. (2022), who focused on applying prompt-tuning within the context of task transfer in the SPoT paper. PromptTuning, as investigated in our study, is a promising avenue for performance improvement on the target task, especially in the context of SPoT (**S**oft **Pro**mpt **T**ransfer).

In our replication study, we improved PromptTuning by employing intermediate steps using the SPoT approach and evaluated its performance on the GLUE and SuperGLUE benchmarks. SPoT first learns a prompt on a single or mixture of source tasks and then uses the trained prompts to initialize the prompt for target tasks, on which the performance is evaluated. For GLUE and SuperGLUE benchmarks, we employed two single tasks (MNLi and SQuAD) and a mixture of 7 NLI tasks. Additionally, we utilized a task transferability approach that includes four source tasks (MNLI, QQP, SQuAD, SST-2) and three target tasks (BoolQ, CoLA, MRPC). This analysis aims to contribute to understanding the effectiveness

of prompt-tuning within the broader context of task transferability and its implications for downstream NLP tasks.

2 Related Work

Language Models. The language models (LM) landscape has witnessed rapid and substantial development, enhancing their proficiency in addressing diverse tasks. This progress has been instrumental in shaping the attention method and transformer architecture, as evidenced by prior research (Vaswani et al., 2017).

Benchmark models like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have emerged, representing cutting-edge solutions for many NLP tasks. These models have demonstrated the effectiveness of pre-trained language representations in capturing contextual information, thereby achieving remarkable performance in downstream applications.

In 2019, the introduction of the T5 model (Raffel et al., 2020) marked a notable milestone. Unlike its predecessors, T5 is a sequence-to-sequence model where both the input and output are in textual form. This departure from traditional BERT-style models, which typically produce a class or span as output, underscores the versatility of T5.

Building upon the foundation laid by T5, Flan-T5 (Chung et al., 2022) represents an advancement. This model inherits the sequence-to-sequence paradigm and performs better than its predecessor. The development of Flan-T5 involved training on an extensive dataset comprising 473 datasets and tackling 1837 diverse tasks, illustrating its adaptability and effectiveness across a wide spectrum of NLP challenges.

Parameter Efficient Fine-Tuning. The increasing number of parameters in contemporary language models poses a significant challenge when fine-tuning them for downstream tasks. To address this challenge, researchers have explored various techniques falling under the **P**arameter **E**fficient **F**ine-**T**uning (PEFT) methods, aiming to reduce the computational demand of fine-tuning without compromising performance. Currently, there are a large number of PEFT methods, and they can be divided into five categories: additive, unified, reparametrized, hybrid, and partial fine-tuning (Xu et al., 2023).

Reparametrized methods, a prominent subset of PEFT, leverage low-rank transformations to diminish the number of trainable parameters while preserving the ability to work with high-dimensional matrices. Noteworthy examples in this category include LoRA (Hu et al., 2021), AdaLoRA (Zhang et al., 2023) or QLoRA (Dettmers et al., 2023). These methods have gained prominence for their effectiveness in achieving parameter efficiency during fine-tuning.

In addition to reparametrized methods, additive fine-tuning techniques have emerged as another influential category within PEFT. In these approaches, trainable parameters are introduced and added to the model while the rest remains frozen. Examples of additive fine-tuning methods include Adapters (Houlsby et al., 2019), PromptTuning (Lester et al., 2021), Prefix-Tuning (Li and Liang, 2021), as well as the derived SPoT (Vu et al., 2022) and Attempt (Asai et al., 2022).

In the context of additive fine-tuning, PromptTuning (Lester et al., 2021) represents a method that incorporates additional information to condition the model during the generation process. This additional information is in the form of an added embedding layer to the base model, where only this embedding undergoes training during the fine-tuning process, and the rest of the model keeps frozen. PromptTuning builds upon the concept of prompting techniques, commonly employed only during inference to introduce extra information to accomplish the desired task.

Task Transferability in NLP. Numerous studies have been conducted to analyze and predict the transferability of tasks within the realm of NLP (Bingel and Søgaard, 2017; Vu et al., 2020; Poth et al., 2021). These investigations delve into the intricate dynamic of transferring knowledge between various NLP tasks. One key finding highlighted in existing research is the efficacy of transferring knowledge from tasks with resource-rich data in the source domain (Phang et al., 2018).

The prevailing methodologies in these studies often revolve around using embeddings extracted from the input text. Additionally, researchers explore alternative approaches, such as those based on adapters, to enhance the understanding of task transferability. While these efforts have significantly contributed to our understanding, there is still space for exploring novel techniques and methodologies to improve task transferability predictions.

Furthermore, recent research has extended the exploration of task transferability into the domain of prompt-tuning. Investigations have been conducted to analyze the effectiveness of transferring knowledge through soft prompts across different downstream tasks (Su et al., 2022). This line of inquiry opens up

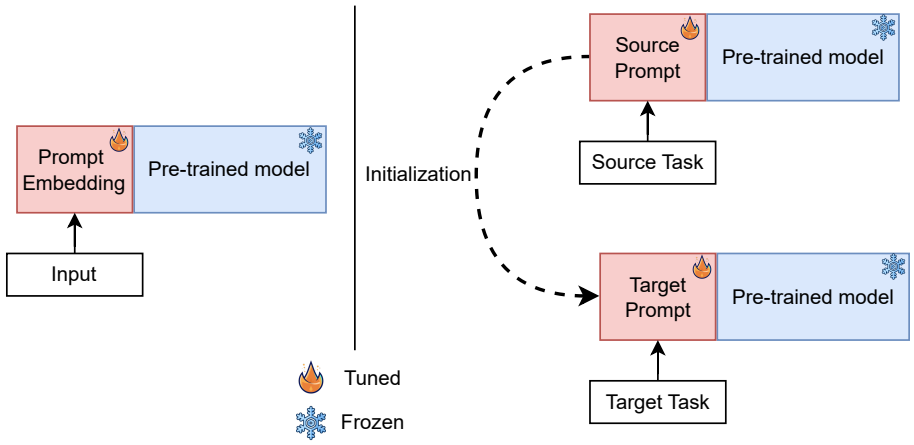

Figure 1: An illustration of the PROMPTTUNING (***left***) and SPOT method (***right***). **Left**: Standard prompt-tuning without the prompt transfer. The Prompt embedding is trained only on a single task. **Right**: Prompt transfer between the source and target tasks by initializing the target prompt with the trained embeddings on the source task.

new avenues for understanding how prompt-based strategies can influence the transferability of knowledge in NLP.

3 Replicated Method

The replicated method consists of two prompt-tuning steps, where the authors utilize the knowledge transfer between various NLP tasks. This section describes the standard prompt-tuning approach and extended version using soft prompt transfer.

3.1 Parameter-Efficient Prompt Tuning

Previous work done by (Brown et al., 2020) demonstrated that the PROMPTDESIGN can adapt the behaviour of frozen GPT-3 through text prompts and improve the overall performance of LLMs. The PROMPTDESIGN formulates each task as a language modeling task, where the pre-trained model remains frozen during inference. This approach exhibited remarkable performance on several NLP benchmarks utilizing GPT-3 models. However, it still under-performs traditional model fine-tuning approaches (Lester et al., 2021).

Following the findings in the PROMPTDESIGN, Lester et al. (2021) introduced the parameter efficient fine-tuning method, PROMPTTUNING, which adds additional information for the model in the form of token embeddings that are prepended before the input embedding. In the training process, this added information aims at conditioning the model output to improve the overall performance. Figure 1 ***left*** shows the original implementation of prompt-tuning on the target task.

The authors employed the T5 model with the text-to-text approach for their experiments and fine-tuned the prompts for several NLP tasks. During the fine-tuning, the pre-trained model is frozen, and only the added embedding layer is trained, significantly reducing the number of trainable parameters. This brings the advantage of not fine-tuning the whole model but instead fine-tuning the desired embedding layer that can be replaced by another prompt embedding, making the model easier to adapt to the downstream tasks.

3.2 Soft Prompt Transfer

The SPOT paper introduces the prompt transfer between source and target tasks to improve the original implementation of PROMPTTUNING . Source prompt-tuning is an intermediate step between the pre-trained model and the standard prompt-tuning on the target task. In both cases, only the prompt is trained, and the rest of the model is frozen. The SPOT approach consists of two steps.

The source prompt is trained using the target tasks in various settings in the first step. These settings include a single supervised learning, employing only a single source task, and a multi-task mixture, where they utilized several datasets in the mixture and trained the model on the created mixture of tasks.

The second step involves fine-tuning the target prompt on the target task using the previously trained source prompt. The source prompt constitutes the intermediate representation used to initialize the target prompt before the final prompt-tuning. The approach benefits from all the advantages of PROMPTTUNING and adds the benefit of inter-task information transfer, which helps better adapt the target prompt to the desired task. The two-step approach is in Figure 1 ***right***.

4 Experiments & Results

We conducted experiments to improve the results of PROMPTTUNING using the soft prompt transfer between source and target tasks. We aimed to show that a prompt pre-trained on a data-rich dataset can improve results on low-resource datasets. This transfer can then be exploited not only on our selected datasets but on a wider range of data, even those not originally used to train the model.

4.1 Improving PromptTuning with SPoT

The first experiment focused on improving the original prompt-tuning with the knowledge transfer between source and target tasks.

4.1.1 Experimental setup

We chose the T5 model for our experiments, focusing on only one size, namely the BASE version, which has 220M parameters. In the original SPOT paper approach, they utilized the LM-adapted version of the T5-BASE model that was found to be much better to optimize based on PROMPTTUNING (Lester et al., 2021). However, in our experiments, we were unable to transform the T5-BASE LM-adapted properly to the HuggingFace format, thus we used the unofficial version from HuggingFace[1]. Employing the unofficial version of the T5-BASE LM-adapted model, we observed instability during the training process, resulting in inferior outcomes compared to the original results reported in the SPoT paper.

4.1.2 Datasets

In the selection process of datasets from the original SPOT paper, we focused primarily on the natural language inference (NLI) task and question answering (QA) datasets. Table 1 shows a list of the selected datasets for all our experiments.

We selected two task variants. The first single supervised learning task, in which we chose the MNLI and SQUAD datasets as the single source task. The second variant is a multi-task mixture, in which we utilized an NLI task consisting of 7 datasets. We selected these datasets as the data on which we trained source prompts that were used to initialize the target prompt-tuning. In our experiments, we considered the same data preprocessing and transformation to sequence-to-sequence format according to (Raffel et al., 2020).

As evaluation benchmarks, we selected GLUE (Wang et al., 2018) and SUPERGLUE (Wang et al., 2019), as in SPoT, to compare our results against their outcomes. The GLUE contains eight datasets targeting various tasks such as NLI, sentiment analysis, grammatical acceptability, and paraphrase detection. However, the original version of GLUE also contains the problematic WNLI (Levesque et al., 2012) dataset, which we excluded from the evaluation. The second benchmark used to evaluate the trained source prompts is SUPERGLUE, consisting of 8 datasets, including tasks such as QA, NLI, coreference resolution, or word sense disambiguation.

4.1.3 Training details

We use the steps and training parameters similar to those in the SPOT paper. These steps of the training process are divided into 2 phases. Only the prompt is trained during both phases without the rest of the model. The trainable prompt is an embedded input sequence, and in all cases, we use a length of 100 tokens for both source and target prompt-tuning. In all the training runs, we set a fixed number of steps during which the source and target prompts were trained.

The first phase concerns training the prompt on the source task. For this purpose, we use a number of steps equal to 2^{18}, which is 262,144. There are several different methods of initializing the source prompts, such as using names of classes in the case of the classification task, using specific text encoded before training, or using the most commonly used tokens in the dictionary. All source prompts were initialized

[1] `https://huggingface.co/liangtaiwan/t5-v1_1-lm100k-base`

Name	Task type	Type	Size	Citation
MNLI	NLI	source	393K	Williams et al., 2018
SQuAD	QA	source	88K	Rajpurkar et al., 2016
NLI				
DocNLI	NLI	source	942K	Yin et al., 2021
SNLI	NLI	source	550K	Bowman et al., 2015
MNLI	NLI	source	393K	Williams et al., 2018
QNLI	NLI	source	105K	Wang et al., 2018
ANLI	NLI	source	17K	Nie et al., 2020
RTE	NLI	source	2K	Dagan et al., 2005
CB	NLI	source	250	De Marneffe et al., 2019
SuperGLUE				Wang et al., 2019
ReCoRD	QA	target	101K	Zhang et al., 2018
MultiRC	QA	target	27K	Khashabi et al., 2018
BoolQ	QA	target	9K	Clark et al., 2019
WiC	word sense disambiguation	target	5K	Pilehvar and Camacho-Collados, 2019
RTE	NLI	target	2K	Dagan et al., 2005
WSC	coreference resolution	target	554	Levesque et al., 2012
COPA	QA	target	400	Roemmele et al., 2011
CB	NLI	target	250	De Marneffe et al., 2019
GLUE				Wang et al., 2018
MNLI	NLI	target	393K	Williams et al., 2018
QQP	paraphrase detection	target	364K	Iyer et al., 2017
QNLI	NLI	target	105K	Wang et al., 2018
SST-2	sentiment analysis	target	67K	Socher et al., 2013
CoLA	grammatical acceptability	target	9K	Warstadt et al., 2019
STS-B	semantic similarity	target	6K	Cer et al., 2017
MRPC	paraphrase detection	target	4K	Dolan and Brockett, 2005
RTE	NLI	target	2K	Dagan et al., 2005

Table 1: The list of datasets used in our experiments of improving PromptTuning with SPoT.

with the most common tokens in the vocabulary, selecting the first 100 tokens to encode and initialize the incorporated embedding layer with their values. We utilized a learning rate of 0.3, the Adafactor optimizer, a batch size 32, and a weight decay of $1e^{-5}$ as additional parameters. For the mixture of source tasks with more than 2^{19} records, we utilized the examples-proportional mixing strategy introduced by Raffel et al. (2020). In contrast to the original implementation, we employed the PyTorch framework and our implementation of the prompt-tuning method.

Phase two consists of training the target prompt, whereas in this phase, the prompt is initialized using the prompt trained on the selected source task. The fine-tuning parameters remain the same as for source prompt-tuning. In contrast, during target tuning, we stored a checkpoint every 500 steps, and simultaneously, we analyzed the best model every 500 steps based on the validation score of the loss function. As a result, we considered the best-trained model as the one with the best results on the loss function.

4.1.4 Effect of SPoT

To evaluate the effect of Soft prompt-tuning and the intermediate step of prompt-tuning the model on the source task, we utilized several tasks and evaluated them on GLUE and SuperGLUE datasets. In Table 2, we present our outcomes of the T5-Base and T5-Base LM-adapted in comparison with the results from the SPoT paper. Additionally, we have also included the results of the original PromptTuning method as a baseline to compare the effect of soft prompt transfer.

Similarly to the SPoT paper, we reported the results on the validation split as the mean of all metrics calculated for each dataset from GLUE and SuperGLUE.

Results using T5-Base. Based on the results presented in Table 2, soft prompt transfer in most cases yields improved results compared to the Baseline, which represents standard prompt-tuning without

	SPoT		Ours – T5		Ours – T5-LM	
	GLUE	**SuperGLUE**	**GLUE**	**SuperGLUE**	**GLUE**	**SuperGLUE**
PromptTuning	81.2	66.6	82.16	65.63	73.81	59.66
MNLI	82.5	**72.6**	**82.97**	65.62	58.57	59.82
SQuAD	82.2	72.0	82.89	66.48	74.76	**60.92**
NLI (7 tasks)	**82.6**	71.4	82.16	**66.70**	**76.92**	59.90

Table 2: Results of the T5-Base and T5-Base LM adapted model prompt-tuned on three source tasks and evaluated on GLUE and SuperGLUE. The best results for each GLUE and SuperGLUE experiment are boldfaced.

transfer from other tasks. Notably, the distinctions between standard prompt-tuning and prompt-tuning with soft prompt transfer remain within the 1% margin. We performed experiments with the same seeds only once, owing to computational constraints and the required time for each experiment.

The T5-Base model, initially pre-trained on the MNLI task, exhibits the highest level of inter-task transfer capabilities compared to other pre-trained prompts. Conversely, the mixture of datasets for the NLI task demonstrated negligible impact, possibly attributed to the dataset diversity within the NLI task. In this regard, multi-task mixing did not prove to be the optimal soft prompt transfer method for the GLUE benchmark.

In contrast, SuperGLUE comprises predominantly low-resource datasets in our experiments, characterized by fewer than 10K samples, potentially contributing to its modest mean scores. Unlike the GLUE benchmark, the mixture of tasks exhibits superior performance to the single-task scenario.

Overall, our experiments with the T5-Base model yielded higher scores for both standard prompt-tuning and transferring from individual tasks to the GLUE benchmark than the reported results in the SPoT paper. However, we did not observe an identical impact as in the original implementation, potentially attributed to factors such as randomness and the influence of the example-proportional mixing strategy, where we do not necessarily have the same training and validation splits. Additional consideration for results discrepancies may stem from our implementation in the PyTorch framework and the absence of the original T5-Base LM-adapted for comparison. We rely instead on the standard T5-Base, recalled by the SPoT paper authors for its reduced capabilities in terms of knowledge and prompt transfer between tasks, and the unofficial version of T5-Base LM-adapted.

Results using T5-Base LM-adapted. In our investigation of the T5-Base LM-adapted, we observed a more notable impact of task transfer compared to the original T5-Base. Notably, the adapted language model exhibited lower performance than the original T5 model and the results reported in the SPoT paper. Evident differences emerged in cross-source tasks for the GLUE benchmark, particularly in the MNLI dataset, where the LM-adapted version achieved approximately 58.6%, contrasting with the Baseline's 73.8%. These findings underscore the instability in the training process for the adapted language model, reflecting observations made by Asai et al. (2022). Conversely, the validation results on the SuperGLUE benchmark are the comparatively low performance from the SPoT Baseline, scoring a 7% reduction.

4.2 Task transferability

With our first experiments, we evaluated the impact of individual datasets and multi-task mixture on GLUE and SuperGLUE datasets. We identified during this experiment that the added intermediate step of source prompt-tuning improved the results over performing the target prompt-tuning from scratch. In this regard, we further explored and analyzed the impact of each dataset utilized in source prompt-tuning on the selected datasets to leverage rich-data datasets as source and low-resource datasets as a target. In this manner, we can also investigate what impact datasets with more data can have in transferring information to datasets with fewer records.

4.2.1 Datasets

To investigate the knowledge transfer between high-resource datasets and those with a low number of data (less than 10K), we selected four datasets that represent source tasks and three that represent target tasks with fewer records. We selected MNLI, QQP, SQuAD, and SST-2 as the high-resource datasets on which we trained the source prompts, with each dataset focused on a different task. This also allows us to

	SPoT			Ours		
	BoolQ	**CoLA**	**MRPC**	**BoolQ**	**CoLA**	**MRPC**
Baseline	73.0	52.9	86.1	77.76	54.89	89.14
MNLI	77.6	54.2	88.4	78.34	54.56	92.29
QQP	75.9	55.6	88.1	77.76	56.40	89.62
SQuAD	76.0	54.9	88.7	79.61	57.80	86.39
SST-2	73.3	52.3	85.6	78.49	53.65	88.35

Table 3: Most of the tasks benefit from the prompt transfer. BASELINE shows the results of prompt-tuning of T5-BASE model only on the target task without prompt transfer. Each cell presents the mean of the metrics for individual target tasks on the validation split. The positive effects of soft prompt transfer are shown in green.

verify information transfer not only between datasets but also between tasks. As target datasets with less than 10K records, we selected BOOLQ, COLA, and MPRC. Like the source datasets, each target data is designed for a different NLP task. More details on the used source and target datasets are shown in Table 1.

4.2.2 Training details

The training of the source prompt on the selected datasets was performed similarly as in Section 4.1.3. The number of prompt-tuning steps was 262,144 on each source task. We saved checkpoints of the prompt at regular intervals and identified the best prompt based on the validation performance, which is then used to initialize the target prompt. The only difference was in the data we used, where we only focused on the single supervised learning task and hence data without a multi-task mixture.

Since target datasets contain less data and are considered low-resource, we utilized less training steps, which we set to 100K on each target task, with a constant saving of the checkpoint every 500 steps, where the best prompt trained is saved based on the validation performance.

4.2.3 Measuring transferability

The results of the task transferability experiments are shown in Table 3. The table presents mean scores on the validation split of individual target datasets and the BASELINE, which is prompt-tuning of the selected model on the target task from scratch without using the intermediate step. Our results are based only on the T5-BASE model, as we identified unstable results for the T5 LM-adapted model in previous experiments.

Results using T5-Base. Table 3, and specifically the OURS section, presents the results we achieved with the T5-BASE model, either on a task fine-tuned from scratch (BASELINE) or using soft prompt transfer between tasks. Based on the data, in most cases, this transfer positively impacted fine-tuning of the target task. However, this transfer did not improve the results in some cases, especially when using SST-2 as the source task. In this case, we identified that except for the target BOOLQ dataset, the soft prompt transfer did not improve the results, similar to the findings in the SPoT paper.

Prompt transfer from the SQUAD datasets achieved the best improvement over the BOOLQ and COLA tasks, specifically by 2 to 3% over the BASELINE, but we did not observe this improvement on MRPC. On the other hand, the prompt pre-trained on the MNLI task reached 92.3% on the MRPC, improving our results over BASELINE by more than 3%.

The transfer outcomes on the BOOLQ task demonstrated that almost all source tasks have a positive effect, except for the QQP task, where we obtained similar results as BASELINE.

Comparison with the SPoT paper. To compare the results obtained by our implementation and those presented in the SPoT paper, we focused on identifying the effect between the individual tasks and the results we achieved on the target tasks compared to those in SPoT.

Assessing the impact of individual source tasks, our study generally corroborates the positive outcomes observed in the original paper regarding prompt transfer between tasks. Nevertheless, we note marginal discrepancies in our findings, particularly in three cases where the transfer failed to improve the outcomes, aligning with the observations reported in the original paper.

We consistently achieved superior scores on the validation datasets on nearly all target tasks. However, an exception arose in the case of the prompt fine-tuned on the source SQuAD dataset and subsequently transferred to the MRPC task, resulting in lower scores by more than 2%. On the other hand, the best enhancement occurred in the SST-2 source task and the BoolQ target task, where the T5-Base model exhibited a score increase of over 5%, contributing to a mean improvement of 1.9%.

5 Conclusion

In this paper, we investigated the effects of transfer learning using soft prompts and attempted to replicate outcomes achieved in SPoT paper. Through our experiments, we demonstrated that soft prompt transfer positively affects the target task, leading to enhanced performance compared to conventional prompt-tuning. Our investigations utilize the T5-Base model, and we note substantial disparities between our results and those of the T5-Base LM-adapted model. These differences are evident compared to the original implementation and manifest as instability in the different training cycles. Concurrently, we observed a notable improvement in transferring knowledge between individual tasks, exceeding 3% in some cases.

While replicating the SPoT , we delved into the PEFT methods and learned that PEFT methods could be more efficient than full model fine-tuning from various aspects, such as the number of trainable parameters. Soft prompt transfer can outperform the full model fine-tuning at the XXL model size. We also found a positive transfer between tasks, in which prompt transfer provides a gain on most target tasks.

The prompt-based transfer learning is not limited to single-language datasets but can be extended to multilingual contexts. An in-depth exploration of the effectiveness of soft prompt transfer in language transferability tasks is essential to future research, particularly concerning advanced multilingual models. A key focus should be investigating the impact of soft prompt training in one language and assessing their transferability to other languages. Moreover, analyzing the potential synergies between language and task transferability is warranted, examining whether prompt-tuning in one language and on one task could influence performance across different languages and tasks. The nuanced investigation promises insights into the intricate dynamics of prompt-based transfer learning across diverse linguistic and task-oriented scenarios.

Acknowledgment

This research was partially supported by DisAI, a project funded by Horizon Europe under GA No.101079164. Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.

References

Akari Asai, Mohammadreza Salehi, Matthew Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6655–6672, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169, Valencia, Spain. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark,

Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Ido Dagan, 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, page 177–190, Berlin, Heidelberg. Springer-Verlag.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Neil Houlsby, 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*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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

Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. 2017. First quora dataset release: Question pairs. *data. quora. com.*

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning 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)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? Efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10585–10605, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. 2022. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel Bowman. 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)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

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

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A large-scale dataset for document-level natural language inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4913–4922, Online. Association for Computational Linguistics.

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

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension.

INO at Factify 2: Structure Coherence based Multi-Modal Fact Verification: A Replication Study

Samuel Revúcky[1] and **Stefanos-Iordanis Papadopoulos**[2]
[1] Comenius University Bratislava,
Faculty of Mathematics, Physics and Informatics, Slovakia
[2] Information Technology Institute, Centre for Research & Technology, Hellas
revucky.samuel@gmail.com, stefpapad@iti.gr

Abstract

In this paper, we present a replication study performed on a multimodal fact-checking model, originally created by team INO to participate in Factify2 challenge at AAAI 2023. In the replication process we reimplemented the model and training procedures to our best abilities. We encountered minor obstacles due to incomplete information in the paper and provided code. Our results slightly vary from the reported numbers. We observe a moderate difference between the replicated and original results, with INO achieving 80.80% and us achieving 76.28% (weighted F1 score on the test set). Our study highlights the importance of implementing open science methodologies to enhance the reproducibility and advancement of multimodal entailment studies. We provide the replicated code with results on GitHub, at https://github.com/samuelrevucky/replication_challenge.

1 Introduction

The DisAI initiative brings together the expertise of the Kempelen Institute of Intelligent Technologies (KInIT), DFKI, University of Copenhagen, and CERTH with the aim of enhancing the scientific excellence of KInIT in the fields of AI and language technologies to combat disinformation.

Within the framework of the DisAI initiative, KInIT is hosting a Replication Challenge to provide a distinctive chance for novice researchers to partner with experts from renowned academic organizations. This challenge is designed to involve the replication of current studies in various fields such as multilingual language technologies, multimodal natural language processing, and ethical artificial intelligence with a primary emphasis on countering disinformation.

Our task in this Replication Challenge is to recreate the approach of team INO on the Factify2 challenge, presented at AAAI 2023 (Suryavardan et al., 2023a). The approach proposed by team INO (Zhang et al., 2023) uses a structure coherence-based approach with components such as textual feature similarity, textual semantic similarity, text length and image similarity. The model is trained and tested on a dataset provided in the competition. The dataset consists of textual and visual data. The architecture extracts textual features using CLIP, S-BERT and the ROUGE. Extraction of visual features is done via ResNet50 pre-trained CNN. These components are used for the final classification through a Random forest classifier. Team INO provided their code publicly.

We were able to replicate the architecture to the scope of the paper. We achieved a noticeably lower performance than team INO, particularly 76.28% as opposed to 80.80% weighted F1 score. We attribute this discrepancy to incomplete information about the architecture and implementation details in both the paper and code, such as the SBERT version or the random seed used in experiments. During the replication we communicated with the authors to clarify the version of SBERT encoder they have used in their work.

2 Related Work

Many different models and methods have been utilized in approach to fact-verification and fake news detection, such as CNNs (Saleh et al., 2021), BERTs (Kaliyar et al., 2021), (Patwa et al., 2021), (Dhankar et al., 2022), RoBERTA (Zhuang and Zhang, 2022), ResNet (Gao et al., 2021), (Zhang et al., 2023), CLIP (Radford et al., 2021), ROUGE (Lin, 2004). Factify 1 (Mishra et al., 2022) provided us one of the largest multimodal fact-verification datasets, with 50k data points and covers news from India and the US, categorized into Support, Insufficient, and Refute. Following it's success at AAAI 2022, Factify 2 at AAAI

2023 released another 50k instances, including data from satirical articles, categorized into 5 categories. Other than Factify, rather large amount of text-based or multi-modal datasets were created in recent years in effort to alleviate fact-checking process. FEVER (Thorne et al., 2018) provides manually updated 185k instances of Wikipedia claims and associated supporting documents, categorised as Support, Refute, or NotEnoughInfo. The fakeddit (Nakamura et al., 2019) dataset contains one million text+image instances taken from reddit and labeled into 6 classes. FakeNewsNet (Shu et al., 2020) provides spatiotemporal and visual data along with news and social context. MOCHEG (Yao et al., 2023) consists of 21,184 assertions, each of which is given a veracity label (support, refute, and not enough information) and an explanation statement.

3 Task Description

The dataset provided in Factify 2 comprises of 50000 samples evenly distributed over 5 categories, which will be described shortly. It uses a 70:15:15 split into training, validation and test sets respectively. The dataset consists of claim-document pairs and is a combination of data from Twitter, fact checking websites and satirical news websites.

The extraction on claims was done from tweets of Hindustan Times, ANI, ABC and CNN, with their corresponding documents extracted from the news articles linked to the tweets. Based on metrics like textual and image similarity, the collected samples were classified into the Support and Neutral categories. For the collection of refute samples, fact checking websites such as Snopes, Factly, and Boom were used, selecting the fake-news as the claim and the article contents as the corresponding document.

Compared to Factify 1 dataset, this iteration had data scraped from satirical websites i.e. Fauxy and EmpireNews as well. These articles were fake but formulated such that it seems real to the reader. Therefore they were added to the support category. By scraping images via searching for the headlines of the articles multimodality of the claims was obtained.

The Factify challenge focuses on detecting fake news through multimodal means. It consists of verifying claims' authenticity by assessing if they align with reliable information sources – documents. This approach recognizes that fact-checking requires thoroughly evaluating textual and visual content.

Every sample includes a claim requiring verification, alongside a supporting document utilized for assessing its accuracy through a comparison or entailment-based method. Both the claim and document incorporate textual and visual data, facilitating a multi-modal approach for verifying facts.

The following five categories are defined to describe the entailment of the claim and document: Support_Text, Support_Multimodal, Insufficient_Text, Insufficient_Multimodal, and Refute. The specific description of these categories is as follows:

- **Support_Text:** the textual data for the claim and document are entailed but their images are not entailed.
- **Support_Multimodal:** the textual data is entailed and the images are also similar for the claim and document.
- **Insufficient_Text:** the textual data is not entailed but the claim and document may have several common words, and the images are not entailed.
- **Insufficient_Multimodal:** the claim and document text are not entailed but they may have common words and the images are also entailed in this case.
- **Refute:** The document text and image both contradict or refute the claim text and image, thus, indicating that the given claim is false.

Examples are shown in Figure 1.

4 Replicated Method

4.1 Architecture Description

Team INO designed a structure coherence-based fact-checking method in which structure-coherence between claims and documents is computed. They extract the following four aspects to reflect structure coherence: literal text similarity, semantic text similarity, text length, and image similarity. In particular, they use ROUGE to extract the literal text similarity, two pre-trained models CLIP and SBERT

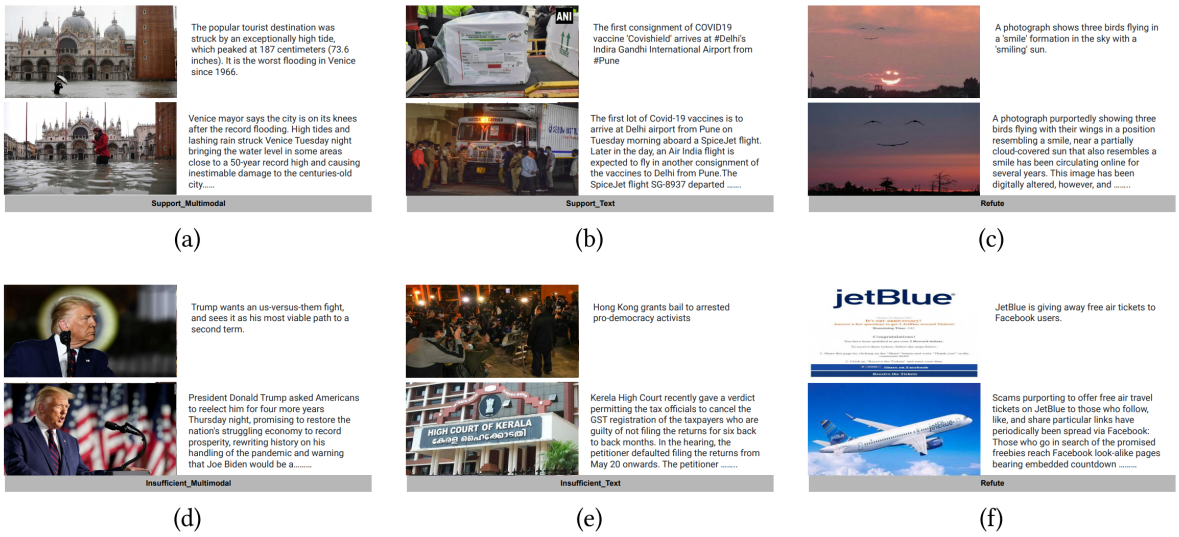

Figure 1: Examples of all the 5 categories from Suryavardan et al. (2023b). The document text supports the claim text in images (a) and (b), it is insufficient in images (d) and (e), while it refutes the claim in images (c) and f). The claim and document images are entailed in images (a) and (d) and not entailed in images (b) and (e).

to extract the semantic text similarity, text length is computed, and finally ResNet50 is utilized to extract image similarity. All the obtained features are normalized and spliced before being passed into the random forest classifier for the final classification result. The architecture is shown in Figure 2.

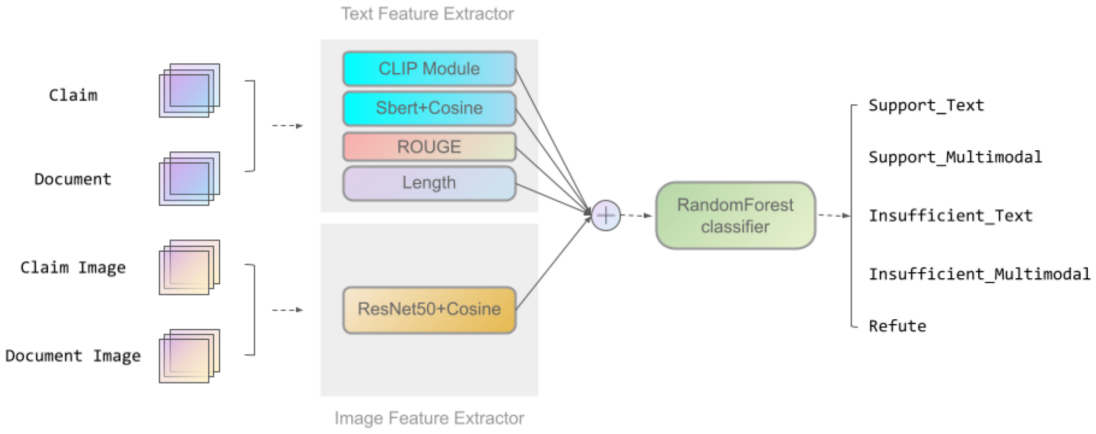

Figure 2: The overall architecture

4.2 Text Feature Extraction

ROUGE is used in the literal text similarity extraction process. Particularly, out of 3 values that ROUGE returns (precision, recall, F1 score), recall is used. This is observable in the provided code [1] but omitted in the paper. Next, claim and document lengths are computed.

To extract semantic similarity, INO uses two approaches. One utilizes an SBERT pre-trained transformer to extract embeddings from claim and document and further applying cosine similarity on them. Important to note is the fact that there is no mention of the particular variant of the SBERT model neither in the paper nor the code. After contacting the authors in this regard they clarified the use of 'paraphrase-MiniLM-L6-v2' variant.

Furthermore CLIP model is used, particularly the 'clip-ViT-B-32-multilingual-v1' according to the code, to extract claim and document features. The obtained 512-dimensional vectors are concatenated and put into an MLP classifier. The classification is done into three categories: support (0), insufficient (1), and refute (2). The MLP is first trained on the train dataset and then is utilized to predict the 3-category labels for the train set, validation set, and test set as well.

4.3 Image Feature Extraction

Extraction of image similarity uses pre-trained ResNet50 convolutional network. Obtained features are then compared using cosine similarity resulting in the final feature.

4.4 Implementation Details

Multiple variations of the architecture were experimented with by INO in the development process, as well as various ablation experiments discussed in 4.5. They divide the exploration into two parts: 1) selection of text pre-training models; 2) ways of using the CLIP. SimCSE, RoBERTa, and the text encoder of CLIP were used to replace Sentence BERT. In addition they also tried replacing the ResNet50 in the image side of the task by the CLIP image encoder. None of these options surpassed the results of SBERT and ResNet50. In the latter, three feature combination methods to use the CLIP module were tried: 1) concatenate the text feature vector into the MLP layer for the three-category classification, which was eventually chosen for the final design; 2) concatenate the image feature vector into the MLP layer for the three-category classification; 3) concatenate all the image and text feature vectors, and input them into the MLP layer for the five-category classification.

For the MLP layer in the CLIP Module, a network with one hidden layer of size 100 and Adam estimator is used. The network was trained in 20 epochs. For the final random forest classifier, following settings were used: number of estimators = 500; max depth = 40; random state = 16. As the particular variant of SBERT used in the text feature extraction is never mentioned by INO, we compared multiple variants, out of which 'all-MiniLM-L6-v2' gave closest results to the samples presented in INO's code. Experiments were conducted using the 'all-MiniLM-L6-v2' and 'paraphrase-MiniLM-L6-v2'.

4.5 Results

During the replication we weren't able to retrieve images for all samples in the datasets. Specifically, we missed claim or document images for 162 samples in the train set, 68 samples in the validation set, and for 81 samples in the test set. These samples were filtered out for further experiments. Our replicated architecture achieved 76.28% weighted F1 score, while our alternative approach with the 'all-MiniLM-L6-v2' SBERT model achieved 77.28%.

The conducted ablation experiments with original results and our results using two different SBERT models are shown in Table 1.

5 Conclusion

In this study, we replicated the INO team's model, achieving similar but slightly lower results to the original paper. Our model scored a result of 76.28%, contrasting with the 80.8% reported by INO. We were able to achieve a one percent improvement by using 'all-MiniLM-L6-v2' SBERT model, resulting in a score of 77.28%.

[1] https://github.com/Catrin-baze/INO-of-factify/blob/main/code.ipynb?short_path=35f3de2#L3111

Model Alternation	INO	Us (all-MiniLM-L6-v2)	Us (paraphrase-MiniLM-L6-v2)
Main Result	0.8080	0.7728	0.7628
Without SBERT	0.7926	0.7602*	0.7602*
Without CLIP	0.7911	0.7058	0.7166
Without ROUGE + length	0.7709	0.7446	0.7425
Without ResNet50	0.6007	0.5962	0.5832
Baseline (SBERT + ResNet50)	0.6664	0.4861	0.4742

Table 1: Ablation experiments showing F1 scores from INO, and our two variants with two SBERT models. * denotes that values for both variants are the same since SBERT model was omitted in this experiment.

Possible causes for score differences may lie in the image processing part. This is supported by three observations. Conducted ablation experiments suggest that the ResNet50 feature accounts for the negative score difference, since when omitted, our results almost align with INO's. This is visible in Table 1. Next, we were not able to retrieve images for all samples, as mentioned in 4.5. Finally, team INO doesn't provide parts of code where they extracted image features using ResNet50. Although following the exact settings mentioned in the paper, we observed minor discrepancy between our features obtained from ResNet50 and values from samples shown in team INO's code.

Our findings indicate that certain configurations might be incomplete in both the paper and code. Nonetheless, it is noteworthy that team INO made their code publicly available. Ultimately, this initiative underscores the collaborative essence of scientific exploration, highlighting the significance of transparent reporting and code accessibility in advancing reproducibility and progress in multimodal entailment research.

References

Abhishek Dhankar, O Zaiane, and Francois Bolduc. 2022. Uofa-truth at factify 2022: A simple approach to multi-modal fact-checking. In *Proceedings of De-Factify: Workshop on Multimodal Fact Checking and Hate Speech Detection*.

Jie Gao, Hella-Franziska Hoffmann, Stylianos Oikonomou, David Kiskovski, and Anil Bandhakavi. 2021. Logically at factify 2022: Multimodal fact verification. *arXiv preprint arXiv:2112.09253*.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. Fakebert: Fake news detection in social media with a bert-based deep learning approach. *Multimedia tools and applications*, 80(8):11765–11788.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shreyash Mishra, S Suryavardan, Amrit Bhaskar, Parul Chopra, Aishwarya Reganti, Parth Patwa, Amitava Das, Tanmoy Chakraborty, Amit Sheth, Asif Ekbal, et al. 2022. Factify: A multi-modal fact verification dataset. In *Proceedings of the First Workshop on Multimodal Fact-Checking and Hate Speech Detection (DE-FACTIFY)*.

Kai Nakamura, Sharon Levy, and William Yang Wang. 2019. r/fakeddit: A new multimodal benchmark dataset for fine-grained fake news detection. *arXiv preprint arXiv:1911.03854*.

Parth Patwa, Mohit Bhardwaj, Vineeth Guptha, Gitanjali Kumari, Shivam Sharma, Srinivas Pykl, Amitava Das, Asif Ekbal, Md Shad Akhtar, and Tanmoy Chakraborty. 2021. Overview of constraint 2021 shared tasks: Detecting english covid-19 fake news and hindi hostile posts. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1*, pages 42–53. Springer.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Hager Saleh, Abdullah Alharbi, and Saeed Hamood Alsamhi. 2021. Opcnn-fake: Optimized convolutional neural network for fake news detection. *IEEE Access*, 9:129471–129489.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big data*, 8(3):171–188.

S Suryavardan, Shreyash Mishra, Megha Chakraborty, Parth Patwa, Anku Rani, Aman Chadha, Aishwarya Reganti, Amitava Das, Amit Sheth, Manoj Chinnakotla, et al. 2023a. Findings of factify 2: multimodal fake news detection. *arXiv preprint arXiv:2307.10475*.

S Suryavardan, Shreyash Mishra, Parth Patwa, Megha Chakraborty, Anku Rani, Aishwarya Reganti, Aman Chadha, Amitava Das, Amit Sheth, Manoj Chinnakotla, et al. 2023b. Factify 2: A multimodal fake news and satire news dataset. *arXiv preprint arXiv:2304.03897*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. 2023. End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2733–2743.

Yinuo Zhang, Zhulin Tao, Xi Wang, and Tongyue Wang. 2023. Ino at factify 2: Structure coherence based multi-modal fact verification. *arXiv preprint arXiv:2303.01510*.

Yan Zhuang and Yanru Zhang. 2022. Yet at factify 2022: Unimodal and bimodal roberta-based models for fact checking. In *Proceedings of De-Factify: Workshop on Multimodal Fact Checking and Hate Speech Detection, CEUR*.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding: A Replication Study

Patrícia Vnenčáková[1], **Matej Jurčák**[1] and **Simon Ostermann**[2]
[1] Comenius University Bratislava,
Faculty of Mathematics, Physics and Informatics, Slovakia
[2] German Research Institute for Artificial Intelligence (DFKI),
Saarland Informatics Campus, Germany
patriciaa.vnencakova@gmail.com, jurcak.matej@gmail.com, simon.ostermann@dfki.de

Abstract

We were trying to replicate the results of the original BERT paper (Devlin et al., 2018) on the General Language Understanding Evaluation (GLUE) benchmark tasks. The BERT model was fine-tuned on GLUE datasets using the same methodology, and its performance was evaluated. The results closely match those reported in the original paper, affirming BERT's effectiveness across diverse NLP tasks.

1 Introduction

BERT is a pre-trained language model that has demonstrated remarkable performance gains on a variety of natural language processing (NLP) tasks. The GLUE benchmark provides a comprehensive evaluation of the ability of models to understand different aspects of language. We discuss this in more detail in subsection 2.1.

The replication process involved fine-tuning the BERT model on the various GLUE datasets and evaluating its performance. We followed the procedures outlined in the original paper to ensure that we obtained results as close as possible to those in the paper. Despite various challenges we encountered, our replication results closely match the performance metrics reported in the original paper for the GLUE tasks we selected.

2 Related Work

As previously stated, the primary source of information for this work was the BERT paper. This section provides a brief summary of it.

2.1 Bidirectional Encoder Representations from Transformers (BERT)

In the original BERT paper (Devlin et al., 2018), the authors – Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova - identified a critical limitation of standard language models: their unidirectional architecture, which prevents effective pre-training. To overcome this constraint, they introduced **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT), an innovative model designed to pretrain deep bidirectional representations from unlabelled text. Unlike previous language representation models, BERT considers both left and right context across all layers, making it a unique advance in the field of language modelling.

BERT addresses the unidirectionality constraint through a unique pre-training mechanism known as the 'masked language model' (MLM). This technique involves randomly masking tokens from the input and asking the model to predict the original vocabulary of these masked words based solely on the contextual cues provided by the surrounding words. By allowing the integration of both left and right contexts, MLM facilitates the pre-training of deep bidirectional transformers, thus broadening the scope of linguistic understanding.

The bidirectional nature of the pre-trained BERT model simplifies the fine-tuning process, requiring only the addition of a single output layer to achieve exceptional performance across a wide range of tasks. This adaptability allows researchers and practitioners to seamlessly tailor BERT to specific applications, including but not limited to sentence-level semantics and sentiment analysis. In particular, the authors present extensive fine-tuning results on 11 NLP tasks, including the GLUE benchmark tasks, SQuAD v1.1, SQuAD v2.0 and SWAG. Process of pre-training and fine-tuning is illustrated on the Figure 1.

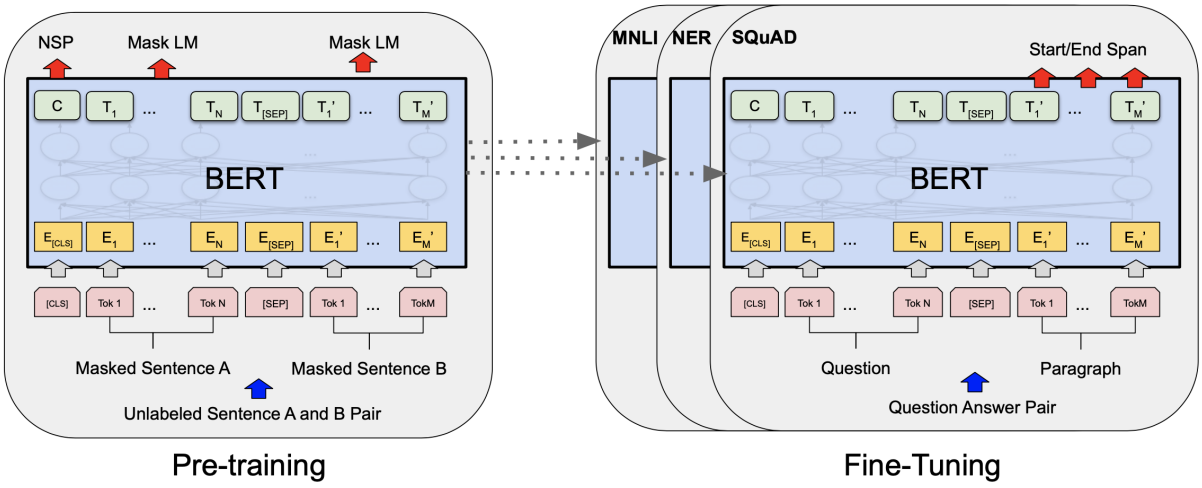

Figure 1: Illustration of the pre-training / fine-tuning approach. Three different downstream NLP tasks, MNLI, NER, and SQuAD, are all solved with the same pre-trained language model, by fine-tuning on the specific task. Image credit: Devlin et al. (2018)

Given our focus on replicating the GLUE tasks, we discuss this collection of diverse tasks in more detail in the 2.2 subsection. The comprehensive evaluation provided by the GLUE benchmark underlines BERT's versatility and effectiveness across different linguistic challenges. By simplifying the model development pipeline, BERT accelerates progress in Natural Language Processing research and application development. Its importance as a foundational tool in the NLP landscape cannot be overstated, providing a flexible and efficient solution for modelling linguistic phenomena and addressing real-world challenges across multiple domains.

2.2 GLUE

The **G**eneral **L**anguage **U**nderstanding **E**valuation benchmark (Wang et al., 2018) provides a collection of different natural language understanding tasks. Designed to capture the breadth of linguistic phenomena, GLUE consists of a series of nine different tasks, each representing a unique dimension of language comprehension and reasoning. These tasks cover various domains such as sentence-level semantics, textual entailment, paraphrase detection and sentiment analysis, providing a comprehensive evaluation of a model's linguistic capabilities.

Each GLUE task is carefully designed to capture different aspects of language understanding, starting from binary classification tasks such as sentiment analysis (SST-2, CoLA) and paraphrase detection (MRPC), to more complex classification tasks such as Multi-Genre Natural Language Inference (MNLI) and Recognizing Textual Entailment (RTE). In addition, GLUE includes tasks that evaluate semantic similarity between sentence pairs (STS-B) and natural language inference between question-answer pairs (QNLI), further enriching the evaluation of the language model.

By taking advantage of the wide range of tasks embedded in GLUE, researchers and practitioners can accurately evaluate the ability of language models to generalise and maintain robustness across a wide range of linguistic phenomena. GLUE will become a fundamental benchmark, facilitating comparative analysis, supporting model selection and driving progress in the field of natural language processing. This in turn will encourage the development of more sophisticated and advanced models of language understanding.

The authors undertook the task of fine-tuning the pre-trained BERT model across all GLUE tasks, with the exception of WNLI due to its particular difficulties[1]. Results from the paper are illustrated on Figure 2. For a fuller explanation of the individual tasks included in GLUE, a detailed description is provided below:

- **MNLI** (Multi-Genre Natural Language Inference): A large-scale, crowdsourced entailment classification task. Given a pair of sentences, the goal is to predict whether the second sentence is an entailment, contradiction, or neutral with respect to the first one (Williams et al., 2018).

[1] See (12) in https://gluebenchmark.com/faq

- **QQP** (Quora Question Pairs): A binary classification task where the goal is to determine if two questions asked on Quora are semantically equivalent (Chen et al., 2018).
- **QNLI** (Question Natural Language Inference): A version of the Stanford Question Answering Dataset (Rajpurkar et al., 2016), which has been converted to a binary classification task (Wang et al., 2018a). The positive examples are (question, sentence) pairs which do contain the correct answer, and the negative examples are (question, sentence) from the same paragraph which do not contain the answer.
- **SST-2** (Stanford Sentiment Treebank): A binary single-sentence classification task consisting of sentences extracted from movie reviews with human annotations of their sentiment (Socher et al., 2013).
- **CoLA** (Corpus of Linguistic Acceptability): A binary single-sentence classification task where the goal is to predict whether an English sentence is linguistically "acceptable" or not (Warstadt et al., 2019)
- **STS-B** (Semantic Textual Similarity Benchmark): A collection of sentence pairs drawn from news headlines and other sources (Cer et al., 2017). They were annotated with a score from 1 to 5 denoting how similar the two sentences are in terms of semantic meaning.
- **MRPC** (Microsoft Research Paraphrase Corpus): Consists of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent (Dolan and Brockett, 2005).
- **RTE** (Recognizing Textual Entailment): A binary entailment task similar to MNLI, but with much less training data (Bentivogli et al., 2009).

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Figure 2: Results from the paper. Image credit: Devlin et al. (2018)

3 Experiments

In this section, we detail the experimental setup used to replicate the results presented in the original BERT paper for a subset of the GLUE benchmark tasks.

3.1 Resources

The original paper presented eight GLUE tasks, which we aimed to replicate. There was optimism from the beginning that we could pull that off, because the authors themselves claimed "all of the results in the paper can be replicated in at most 1 hour on a single Cloud TPU, or a few hours on a GPU, starting from the exact same pre-trained model". However, we soon realized that obtaining a sufficiently powerful GPU or Cloud TPU for the assignment was not feasible. Google Colab was initially considered as an option, but due to limited computing units, we had to settle for a single GeForce RTX 2060 graphics card with 1920 CUDA cores and 6GB of CUDA memory on the local machine.

3.2 Selection of GLUE Tasks

Due to resource limitations, we replicated only a subset of the tasks which have a "reasonable" number of training examples – RTE, MRPC, STS-B, and CoLA. The remaining tasks (MNLI, QQP, and SST-2) have roughly an order of magnitude more training examples, as shown in Table 1. The table also displays the computational time required for fine-tuning using a single learning rate on our machine.

Table 1: Number of Training Examples for GLUE Tasks

Task	**MNLI**	**QQP**	**QNLI**	**SST-2**	**CoLA**	**STS-B**	**MRPC**	**RTE**
# Examples	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k
Time	N/A	N/A	N/A	N/A	$\sim$ 7h	$\sim$ 5h	$\sim$ 2h	$\sim$ 1.5h

3.3 Implementation Details

The fine-tuning procedure was implemented using native PyTorch libraries. We utilized the Hugging Face Transformers library for pre-trained model access and efficient data processing. The pre-trained model chosen was `google-bert/bert-base-uncased`. Although a larger model is available, we chose the base model for convenience and as a starting point. For training, we used PyTorch `TrainingArguments` and `Trainer` classes. Both interfaces provide a user-friendly way to train and evaluate models, while abstracting away the complexities of optimization, batching, and monitoring. As newcomers to the field, we were therefore able to seamlessly integrate the fine-tuning workflow into our replication process. The authors of the paper state they used "batch size of 32 and fine-tune for 3 epochs over the data for all GLUE tasks. For each task, we selected the best fine-tuning learning rate (among 5e-5, 4e-5, 3e-5, and 2e-5)". We used the same parameters except for the batch size, which we had to reduce to 16 due to CUDA memory constraints. To evaluate the results, we utilized the `evaluate` library and a custom function to calculate accuracy. The `compute_metrics` 1 function takes in an `eval_pred` parameter that contains two arrays: `logits` and `labels`. The `logits` array contains the raw output scores or probabilities predicted by the model for each potential class, while the `labels` array represents the true classes to which each data point belongs. The `argmax` function deduces the predicted class for each data point by selecting the class with the highest score. The function should be specified with `axis=-1`, so it selects the class exhibiting the highest score among all possible classes for each data point. For the non-binary STS-B task, we had to adjust the method slightly to use Mean Squared Error (MSE) since the original method was only suitable for binary tasks. As with many well-known datasets, the test split of the dataset does not include labels. Therefore, we used the validation split for evaluation.

Listing 1: Evaluation method

```
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    predictions = np.argmax(logits, axis=-1)
    return metric.compute(predictions=predictions, references=labels)
```

3.4 Results

As the authors did not specify the best learning rates for specific tasks, we trained the model at different learning rates and tried to see how accuracy changes with different learning rates. Table 2 presents our replicated results for each learning rate alongside the corresponding results reported in the original BERT paper. The replicated results demonstrate a close correspondence to the original BERT paper for all chosen GLUE tasks. Although there are minor differences, these can be attributed to the hardware limitations mentioned earlier, differences in batch size, and evaluation on the validation split instead of the test split. In the case of the RTE dataset, we obtained better results than those reported in the paper. However, it is possible that the model learned specific patterns or biases from the validation data that were not as prominent in the original paper's test set. This could result in a slight improvement on the validation set, but it may not necessarily lead to better generalization on unseen data.

Table 2: Replicated GLUE Benchmark Accuracy

Task	**Replicated Numbers**				**BERT**$_{\text{BASE}}$
	5e-5	4e-5	3e-5	2e-5	
CoLA	44.3%	46%	46.7%	45.6%	52.1%
STS-B	83.2%	83.4%	82.3%	81.2%	85.8%
MRPC	87.5%	86.7%	86.5%	86.8%	88.9%
RTE	68.9%	62%	67.5%	62.4%	66.4%

4 Conclusion & Discussion

The important note to take here is the reasoning behind choosing the paper instead of the more specific DisAI one with much more complexity. Reason number one was our limited knowledge of NLP, so after careful consideration and discussion with our mentor, we took the road with choosing the best of two worlds — replicating the numbers from the wildly known paper and diving into the world of fine-tuning and learning a lot. Regarding the replication process itself, the biggest bottleneck was the unavailability of necessary hardware. If we were able to obtain it, we could not only replicate other GLUE tasks or potentially replicate the large model instead of the base one, but we could also iterate much more efficiently. In larger datasets, we only had one opportunity to run the code, which was not ideal due to potential errors the program may encounter. We considered manually fine-tuning at a lower level in native PyTorch, but due to limited iterations, we unfortunately did not pursue this idea.

The results show that the results of the BERT paper are correct, defensible and easy to replicate. The replication process can be carried out by newcomers to the field, as all the necessary hyperparameters are mentioned and the process is described in great detail.

5 Acknowledgement

Despite some challenges, our mentor Simon was helpful throughout the process and gave us valuable feedback. We are very thankful for his help. We would also like to thank KInIT[2] for the opportunity to participate in the Replication Challenge. It is always a great experience to try new things that we have never done before. We have learnt a lot even though we have only covered a small part of NLP.

References

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel Bowman. 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)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

[2] https://kinit.sk/

SpotFake: A Multi-modal Framework for Fake News Detection: A Replication Study

Marek Kajan[1] and **George Karantaidis**[2]
[1] Comenius University Bratislava,
Faculty of Mathematics, Physics and Informatics, Slovakia
[2] Information Technology Institute, Centre for Research & Technology, Hellas
kajan20@uniba.sk, karantai@iti.gr

Abstract

In this paper, we conducted a replication study on a multimodal model for fake news detection called SpotFake, which was published in IEEE BigMM 2019. The original model architecture was implemented in TensorFlow framework. Our replication was implemented in PyTorch framework. Additionaly, we implemented dataset processing for two publicaly available datasets and training procedures according to the methodologies described in the original paper. However, the derived results are below compared to those stated in the original paper. For the Twitter dataset, our replication achieved an accuracy of 62.02%, while for the Weibo dataset it reached an accuracy of 72.59%. In contrast, the original paper reports accuracies of 77.77% for the Twitter dataset and 89.23% for the Weibo dataset. The replicated code can be found at https://github.com/Marek-Kajan/SpotFake-Replication-Challenge.

1 Introduction

DisAI project is the collaborative effort involving the Kempelen Institute of Intelligent Technoologies (KInIT), Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), University of Copenhagen (UCPH) and The Centre for Research & Technology (CERTH). KInIT recognizes that combating disinformation is one of the most important societal challenges today. There's also a lack of investment in R&I sector from the private sector in Slovakia with only 0.5% of Slovak's Gross Domestic Product in 2020 being allocated, placing it at the lower end within the EU. The goal of this project is to improve scientific excellence of KInIT in AI and language technologies to fight disinformation and enhance research management and administrative skills at KInIT. The focus areas of this project are Multilingual Language Technologies, Multimodal Natural Language Processing and Trustworthy Artificial Intelligence.

SpotFake falls into the Multimodal Natural Language Processing focus area. SpotFake was first presented at IEEE International Conference on Multimedia Big Data (Singhal et al., 2019). The term fake news is defined by the paper as "to be news articles that are intentionally and verifiably false, and could mislead readers." (Allcott and Gentzkow, 2017). Reasons to intentionally mislead readers of such news are numerous. Readers' lack of knowledge and neglecting to check the credibility of the sources are the most common reasons for spread of such news. Such news can quite negatively impact the public perception. Another reason behind spread of fake news is lack of automated processes for fact checking. Even now, five years after the publication of the original SpotFake paper, this issue remains highly relevant.

SpotFake addresses this problem by using a multimodal approach to fake news detection. It takes into account both text and images to classify social media posts as either fake or real news. In the following sections, we will examine more closely the methodologies used to create this multimodal framework.

2 Related Work

In 2019, when SpotFake was released, majority of models for fake news detection relied on text and user metadata to detect accounts that create fake news. Some attempts tried to analyze the writing style, because this can have a great impact on overall perception of the news. TransR model tried improve fact checking analysis of the news using Knowledge Graph Embeddings (Lin et al., 2015). Using fake images is also a common way of spreading fake news. There were models trying to analyze if images were altered using the metadata information of the image or the image itself. However all of these approaches are unimodal.

The researchers behind SpotFake while researching various models for fake news detection were mainly inspired by two multimodal models, which they later compare SpotFake to. One of them was Event Adversarial Neural Networks for Multi-Modal Fake News Detection (EANN) (EAN, 2018). This model uses a Convolutional Neural Network (CNN) (Kim, 2014a) for text embedding into representation vector and pre-trained VGG-19 on ImageNet (Simonyan and Zisserman, 2015) to extract image representations. These two vectors were then concatenated and used in two fully connected neural network classifiers for even discrimination and fake news classification.

The second model they were inspired by was Multimodal Variational Autoencoder for Fake News Detection (MVAE) (Khattar et al., 2019). This model used bi-directional LSTMs to extract text representation. Image representations were extracted by VGG-19. These two vectors were then concatenated and used in a decoder that tried to reconstruct the original samples. Secondary task of this model was the fake news detection.

Both of these models share the same problem. That problem is that the fake news detection is a secondary task. This increases the training complexity, model size overhead and can have negative impact on generalization of the fake news detection task, as there is a lack of data for the secondary task. SpotFake solves this by making fake news detection the only task the model solves.

Newer approaches to fake news detection still use similar approaches as SpotFake, trying to combine multiple modalities to get more accurate results. For example, Fake News Detection model based on BLIP (FNDB) (Liang, 2023) extracts textual features using XLNet and visual features using VGG-19. However, these models are pre-trained on single modalities. So in addition it also extracts multimodal features using BLIP, which is a model pre-trained on text image pairs. This can increase the performance of the model, because the unimodal feature extractors can miss some details that the multimodal feature extractor can pick up on.

Another example is the Similarity-Aware Multi-Modal Fake News Detection (SAFE) by Zhou et al. (2020). The aim of this model is to exploit the relationship between the text and image of the news article to determine whether the article is fake or real news. Often, fake news articles will use irrelevant images to attract readers' attention. Textual features are extracted using the Text-CNN model by Kim (2014b). Images are processed by a pre-trained image2sentence model (Vinyals et al., 2017), which creates a text description of the image. These descriptions are then processed using the Text-CNN model by Kim (2014b).

An ambiguity-aware multimodal fake news detection method CAFE Chen et al. (2022) is another multimodal model for fake news detection. It first extracts textual features using pre-trained BERT model (Devlin et al., 2019) and image features using pre-trained ResNet-34 model (He et al., 2015). It then creates a cross-modality alignment by transformint the text and image embeddings into a shared space. For this they have a sub-task, which classifies if the two modalities share common semantics, called semantic correlation (Chen et al., 2022). The CAFE model then fuses the modalities using interaction matrix and along with the semantic correlation it classifies the pair of text and images as fake or real news.

3 Task Description

The task at hand involves determining whether a pair of text and image can be classified into one of two categories: fake or real news. The datasets used in this replication consist of social media posts from microblogging platforms, where both the image and the text of each post contribute to assessing the validity of the news they present. Each entry in the dataset has been manually labeled by its creators, which guides the model in distinguishing between fake and real news. During training, the model learns the types of images and texts that are commonly used to spread misinformation on these platforms.

While this may seem like a straightforward task, distinguishing between real and fake news is not easy and poses a significant challenge, even for humans. Those who intentionally spread misinformation often carefully choose their words and manipulate images to appear as credible as possible. The complexity is further compounded by the fact that the model relies solely on text and images, without any data regarding the source of the information.

4 Methodology

4.1 BERT module

SpotFake is composed of three sub-modules. The first sub-module is the Textual Feature Extractor. This sub-module extracts contextual text features using pre-trained Biderectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019). BERT aims to represent words in a way that best captures their semantic meaning and context effectively. The version of BERT used here includes 12 encoding layers. Each layer applies a self-attention mechanism and then feeds the output through a feed-forward network. The textual features are extracted from the last output layer and then it is passed through fully connected layer to reduce the vector to final size of 32.

BERT begins by taking a tokenized sentence and converting it into token embeddings. These token embeddings are learned during pre-training. They also take into account position of each token in the sequence, because BERT doesn't use recurrence or convolution, it needs a way to capture the sequential order of words in a sequence. These token embeddings are then passed through the 12 encoding layers using a feedforward network. In each encoding layer, Multi-Head Self-Attention Mechanism is applied to the input of the layer. This mechanism allows each word in a sequence to attend to all other words simultaneously, calculating attention scores between all pairs of words in a sequence, enabling the model to capture dependencies between words regardless of their positions in the sequence. The output from these encoding layers is the encoded representation of the sentence, which is further utilized in the SpotFake framework.

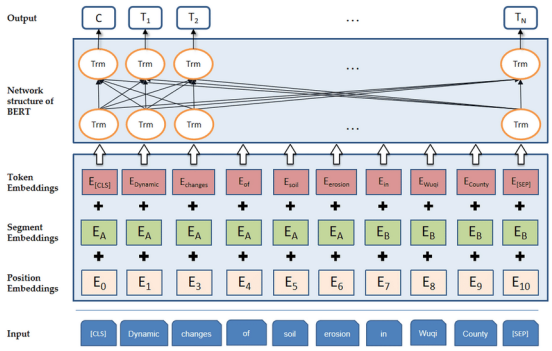

Figure 1: Diagram of BERT architecture (Sun et al., 2022).

4.2 VGG-19 module

The second sub-module is the Visual Feature Extractor, which utilizes the VGG-19 model pre-trained on ImageNet dataset. Similarly to the Textual Feature Extractor, the image representations are extracted from the last output layer, which is then passed through a fully connected layer to reduce the size of the vector to 32.

VGG-19 takes RGB images of size 244x244 as input. The network consists of 16 convolution layers, that are activated by ReLU function. These convolution layers are grouped into blocks. After each block, max pooling is applied to reduce the dimension of the feature maps. The ouput of this block is then passed through 3 fully connected layers activated by ReLu functions. After each fully connected layer, droupout with probability of 0.5 is applied to prevent overfitting during training.

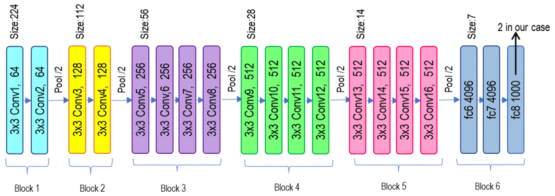

Figure 2: Diagram of VGG-19 architecture (Khattar and Quadri, 2022)

4.3 Multimodal module

Third sub-module is Multimodal Fusion. Here the text and image representations are concatenated, which then passed through two fully connected layers. The first of the layers is activated by ReLU function. The second layer is the final output layer activated by a sigmoid function. Output from this layer is then used to classify the original input into fake and real news.

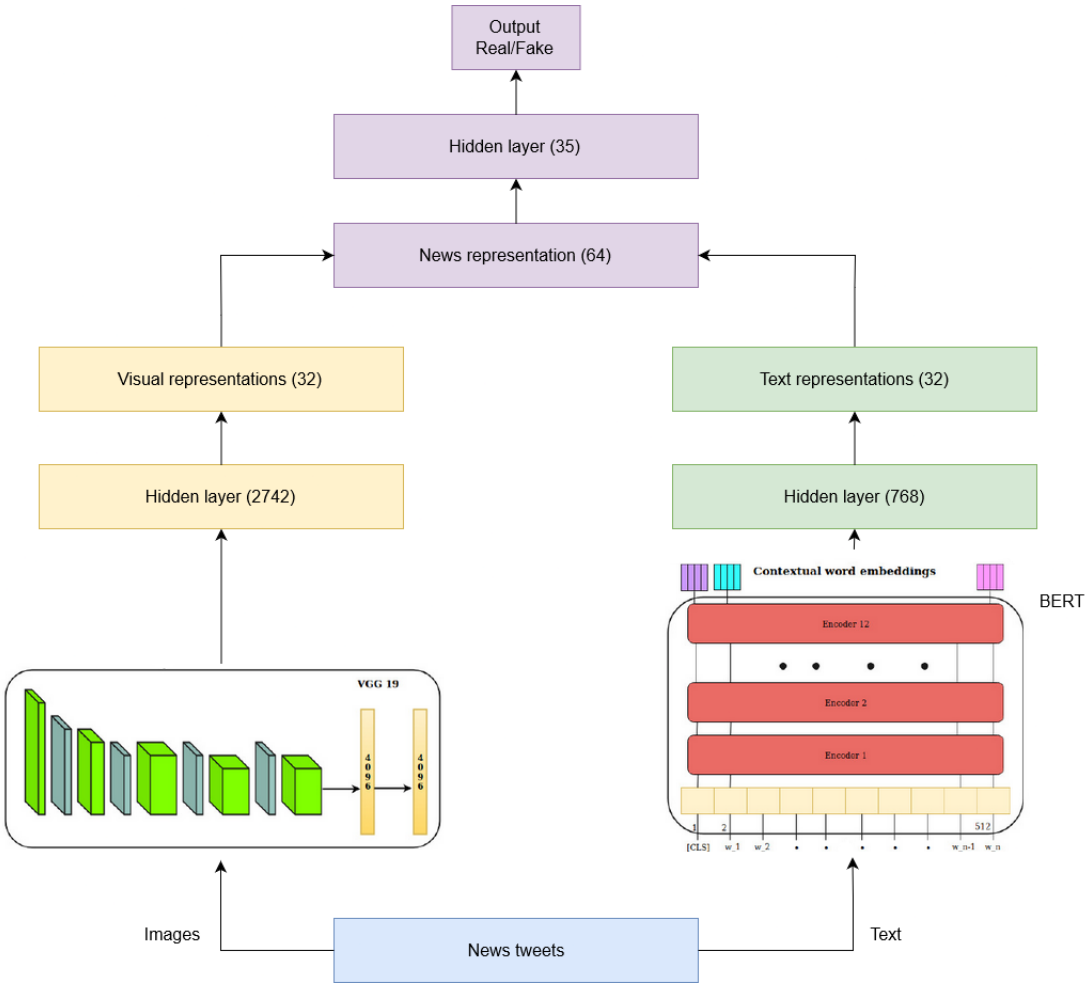

Figure 3: A schematic diagram of the proposed SpotFake model (Singhal et al., 2019). Value in () indicates number of neurons in a layer.

5 Datasets

For SpotFake training, two different publicly available datasets were used. The model was also slightly tweaked for both of the datasets

5.1 Twitter MediaEval Dataset

This dataset originates from The Verifying Multimedia Use challenge, part of the MediaEval Workshop on Detection and Visualization of Misleading Content on Twitter (Detection and visualization of misleading content on Twitter, 2018). It is available in the following GitHub repository `https://github.com/MKLab-ITI/image-verification-corpus`. The version used was the one from 2016. The training set comprises approximately 17,000 unique tweets from the social media platform Twitter of which 9000 tweets are labeled as fake news and 6000 as real news. The test set contains 2000 news tweets. Included in the dataset are also tweets, that are not written in English language.

5.2 Weibo Dataset

This dataset consists of real news from Chinese news sources, such as Xinhua News Agency and posts from a Chinese microblogging platform, called Weibo (Jin et al., 2017). It contains 9527 samples, which are split into 7531 training samples and 1996 testing samples. It is available at the following GitHub repository: `https://github.com/wangzhuang1911/Weibo-dataset`.

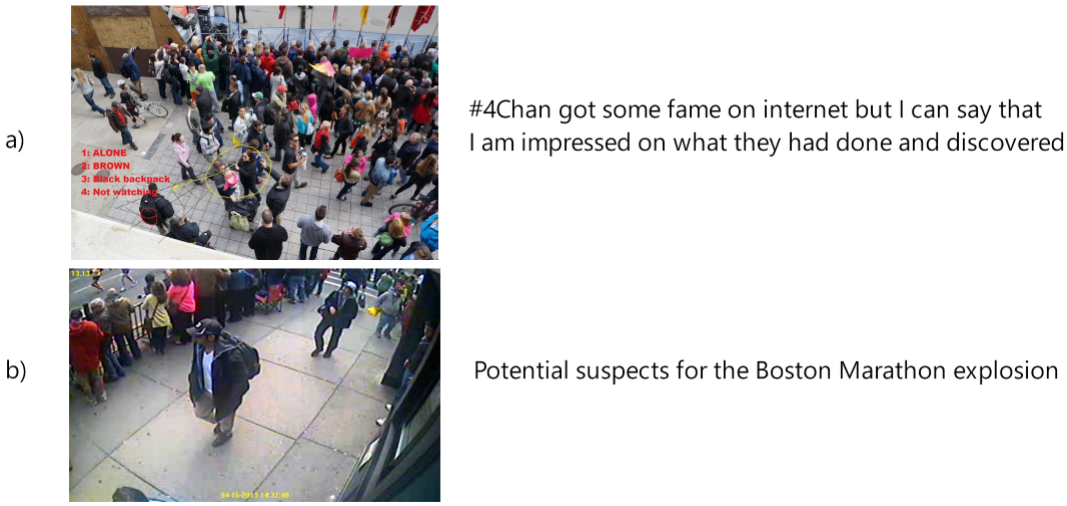

Figure 4: Example of tweets from Twitter dataset regarding Boston Marathon bombing (Staff, 2013). The top image depicts a tweet labeled as fake news while the bottom image depicts a tweet labeled as real news.

6 Replication Methods

For replicating the SpotFake approach, we chose to employ the PyTorch framework, whereas the original paper's model was implemented in TensorFlow. While the original code is available, we chose not to use it for our implementation. Instead, we endeavored to replicate the model using information solely from the paper, resorting to consulting the code only when certain details were unclear.

The text underwent minimal pre-processing, which involved fixing the length of the sequence. Using the BERT tokenizer, the text was tokenized and then trimmed or padded to achieve a desired length of 23 word tokens for the Twitter dataset and 200 character tokens for the Weibo dataset.

As for the images extracted from the dataset, the only pre-processing applied was resizing them to dimensions of 244x244x3.

The BERT model used in the Textual Feature Extractor was the 'bert-base-uncase' model, accessible on Hugging Face. Upon encoding the tokenized text, it yielded a vector of length 768. This vector was then passed through two fully connected layers. The first layer had a size of 768 and the second layer had a size of 32. The output of this final layer represented the text.

For the Image Feature Extractor, we employed the VGG-19 model pre-trained on ImageNet from the PyTorch library. This model encodes the images into a vector of length 4096. Subsequently, this vector is passed through two fully connected layers with sizes of 2742 and 32 for the Twitter dataset, and through one fully connected layer with a size of 32 for the Weibo dataset. The resulting vector of length 32 serves as the final image representation.

The final text and image representation vectors are concatenated into a single vector of size 64, which is then passed through two fully connected layers with sizes of 35 and 1.

Each fully connected layer is activated by the ReLU activation function, except for the last output layer used for classification, which is activated by a sigmoid function. Additionally, after each fully connected layer, a dropout with a probability of 0.4 is applied.

Training was conducted with a batch size of 256 using the Adam optimizer, with early stopping based on validation accuracy. However, we did not have a separate validation dataset. Similar to the original code, we utilized the test datasets for validating accuracy.

No hyperparameter tuning was performed. Instead, we adopted the hyperparameters from the paper, which were determined to be the most effective

parameters	Twitter	Weibo
BERT trainable	False	False
VGG trainable	False	False
# of hidden layers (text)	2	2
# of neurons in each layer (text)	768, 32	768, 32
# of hidden layers (image)	2	1
# of neurons in each layer (image)	2742, 32	32
# of dense layers (concatenation)	1	32
# of neurons in dense layer (concatenation)	64	64
text length	23 words	200 characters
batch size	256	256
optimizer	Adam	Adam
learning rate	0.0005	0.001

Table 1: An overview of hyper parameter setting used in SpotFake.

7 Results

The resulting accuracy of our replication is presented in Table 2, showing that our accuracy is notably lower than that of the original model. Several factors may contribute to this difference. Firstly, the models used in our replication and the original SpotFake implementation may not be identical. Our replication was conducted in the PyTorch framework, whereas the original SpotFake model was implemented in TensorFlow. Additionally, different sources may have been utilized for pre-trained models, and changes in libraries over time since the original implementation in 2019 could also affect performance.

Another possible factor is the lack of a validation dataset for early stopping during training. We used the test dataset for validation, which is not considered best practice. However, it appears that the original source code also followed a similar approach.

Overall, these differences in framework, model sources, and training practices may contribute to the observed variance in accuracy between our replication and the original model.

Model	Twitter accuracy %	Weibo accuracy %
SpotFake (Singhal et al., 2019)	77.77	89.23
Replication	62.02	72.59

Table 2: Accuracy results of the original SpotFake model and the replicated one for the Twitter and Weibo datasets.

8 Conclusion & Discussion

In this study, our attempt to replicate the results of the SpotFake model from the original paper did not yield the same outcomes. The accuracy we achieved for the two different datasets was 62.02% for the Twitter dataset and 72.59% for the Weibo dataset, noticeably lower than the original results of 77.77% for the Twitter dataset and 89.23% for the Weibo dataset. This significant disparity in accuracy between the original and replicated models may stem from various factors. The original code dates back to 2019, and since then, there have been substantial changes in libraries and models. Additionally, differences between TensorFlow and PyTorch frameworks could have played a role. It's also possible that a small oversight on our part contributed to this notable discrepancy.

However, it's important to note that this does not invalidate the original results. Replicating studies is crucial for advancing cumulative knowledge in a specific field. It allows for the identification of potential errors, biases, and limitations in methodologies. In cases where replications fail, they can serve as catalysts for further investigation into the reasons behind the discrepancies. This process deepens our understanding in the field and fosters collaboration among researchers. Moreover, it underscores the significance of transparency in research. When research is transparent and replicable, it paves the way for further examination and advancement in the field.

In conclusion, this highlights the importance of transparency and collaboration in research across all fields. These principles are essential for ensuring the continuous development and evolution of knowledge. In the case of our failed replication, further investigation into the reasons behind the discrepancies will be valuable for advancing our understanding of the topic.

References

2018. Eann: Event adversarial neural networks for multi-modal fake news detection,. *KDD '18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Hunt Allcott and Matthew Gentzkow. 2017. Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives*, 31(2):211–236.

Yixuan Chen, Dongsheng Li, Peng Zhang, Jie Sui, Qin Lv, Tun Lu, and li Shang. 2022. Cross-modal ambiguity learning for multimodal fake news detection. pages 2897–2905.

Detection and visualization of misleading content on Twitter. 2018. Boididou, christina and papadopoulos, symeon and zampoglou, markos and apostolidis, lazaros and papadopoulou, olga and kompatsiaris, yiannis. *International Journal of Multimedia Information Retrieval*, 7(1):71–86.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition.

Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, and Jiebo Luo. 2017. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 795–816, New York, NY, USA. Association for Computing Machinery.

Anuradha Khattar and Syed Quadri. 2022. "generalization of convolutional network to domain adaptation network for classification of disaster images on twitter". *Multimedia Tools and Applications*, 81.

Dhruv Khattar, Jaipal Singh Goud, Manish Gupta, and Vasudeva Varma. 2019. Mvae: Multimodal variational autoencoder for fake news detection. In *The World Wide Web Conference*, WWW '19, page 2915–2921, New York, NY, USA. Association for Computing Machinery.

Yoon Kim. 2014a. Convolutional neural networks for sentence classification.

Yoon Kim. 2014b. Convolutional neural networks for sentence classification.

Zhiping Liang. 2023. Fake news detection based on multimodal inputs. *Computers, Materials & Continua*, 75(2):4519–4534.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2181–2187. AAAI Press.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition.

Shivangi Singhal, Rajiv Ratn Shah, Tanmoy Chakraborty, Ponnurangam Kumaraguru, and Shin'ichi Satoh. 2019. Spotfake: A multi-modal framework for fake news detection. In *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, pages 39–47.

CNN Staff. 2013. What we know about the boston bombing and its aftermath. *CNN*.

Junlin Sun, Yanrong Liu, Jing Cui, and Handong He. 2022. Deep learning-based methods for natural hazard named entity recognition. *Scientific Reports*, 12:4598.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2017. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):652–663.

Xinyi Zhou, Jindi Wu, and Reza Zafarani. 2020. Safe: Similarity-aware multi-modal fake news detection.

Scaling sentence embeddings with large language models: A Replication Study

Kristína Sásiková[1] and **Michal Gregor**[2]
[1] Comenius University Bratislava,
Faculty of Mathematics, Physics and Informatics, Slovakia
[2] Kempelen Institute of Intelligent Technologies, Bratislava, Slovakia
sasikova10@uniba.sk, michal.gregor@kinit.sk

Abstract

In our paper, we present a replication study focusing on sentence embeddings, especially scaling them with large language models (LLM) and trying new methods, that could improve their performance. The replication process involved a thorough study of the suitable materials, reimplementation of the code, training procedures and evaluation methodologies described in the original paper. Results of our process closely match the reported ones, affirming the robustness of used methods. Minor disparities between replicated and original results can be attributed to factors like subtle variations in model configurations and randomness. Our research highlights the importance of embracing open science practices to enhance reproducibility and advance progress in LLMs, especially their application on sentence embeddings. Replicated code can be accessed at the following GitHub repository: https://github.com/KristinSas/DISAI_replication_challenge_2024.

1 Introduction

The DisAI project brings together the Kempelen Institute of Intelligent Technologies (KInIT), DFKI, University of Copenhagen, and CERTH to enhance KInIT's scientific excellence in artificial intelligence and language technologies to combat disinformation. As the Slovak research and innovation ecosystem faces challenges, which include a lack of scientific excellence and cooperation between industry and academia, the main goals of the DisAI project are to strengthen research management, administrative skills and support for excellent research and to improve the scientific excellence of KInIT in three main areas of artificial intelligence and language technologies: multilingual language technologies, multimodal natural language processing, and trustworthy artificial intelligence.

The replication challenge is a part of the DisAI project. The main task is to replicate existing research in multilingual language technologies, multimodal natural language processing, or trustworthy artificial intelligence with specialization on disinformation combating.

Our aim in the replication challenge was to reproduce research described in the article called "Scaling Sentence Embeddings with Large Language Models" (Jiang et al., 2023). It presents a way in which a model trained for the express purpose of creating sentence embeddings can, in principle, be replaced with a large language model pre-trained in a much more general and task-agnostic way and which typically has support for much longer context lengths. The approach is fairly general and could be used in a number of applications, including disinformation combating. In the following sections, we are going to explore the details of this article and the different methods introduced by the authors. We are going to present the results of our replication and the expected contributions to the field of LLM-based sentence embeddings.

2 Related Work

To provide the reader with context, in this section we are going to consider some related works concerning both sentence embeddings and large language models. This section is closely modelled on Jiang et al. (2023).

Sentence embeddings: Sentence embeddings involve converting a sentence into a fixed-size vector that encapsulates its semantic meaning and context. This enables efficient retrieval of similar sentences based on vector similarity. Recent advancements, such as SimCSE (Gao et al., 2022), have demonstrated the effectiveness of contrastive learning in generating sentence embeddings using models like BERT, both in unsupervised and supervised scenarios.

In the unsupervised setup, SimCSE predicts the input sentence itself from within-batch negatives, employing various dropout masks (Srivastava et al., 2014). In supervised learning, natural language inference datasets (see (Conneau et al., 2017; Reimers and Gurevych, 2019)) are utilized to create positive and negative pairs. Building upon SimCSE's success, there has been a surge in exploring contrastive learning based methods. For instance, DiffCSE (Chuang et al., 2022) introduces a replaced token detection loss into the contrastive learning framework. PromptBERT (Jiang et al., 2022) reveals the potential of prompts in enhancing BERT's sentence representation capability.

Furthermore, studies have investigated data augmentation techniques for sentence embeddings using Large Language Models (LLMs). Approaches like SentenceT5 (ST5) (Ni et al., 2021) leverage the encoder-decoder structure of models like T5 (Raffel et al., 2020) for generating sentence embeddings, showcasing improvements through scaling the model's parameters. However, utilizing LLMs directly for sentence embedding generation remains an area of active research.

Large Language Models: Recently, Large Language Models (LLMs) (Zhang et al., 2022; Touvron et al., 2023; Chowdhery et al., 2022) have demonstrated remarkable performance across various natural language processing tasks, benefiting from their substantially larger parameter sizes compared to previous pretrained language models. LLMs are adept at efficiently acquiring new task capabilities through in-context learning, leveraging training data as demonstrations (Brown et al., 2020). Remarkably, LLMs employing in-context learning can tackle intricate tasks such as multitask language understanding (Hendrycks et al., 2021), commonsense reasoning (Lin et al., 2022), and mathematical problem-solving (Cobbe et al., 2021) without requiring gradient updates. Scaling up language models further enhances their performance in these tasks (Hoffmann et al., 2022; Kaplan et al., 2020).

3 Task description and methodology

In this section, we describe different methods introduced by the researchers in their papers. First, we specify their prompt-based sentence embeddings method called Explicit One word Limitation (PromptEOL) in section 3.1. Afterwards, we describe two settings: with and without fine-tuning. For the setting without fine-tuning, in section 3.2 we describe the method based on in-context learning and for the setting with fine-tuning the one based on contrastive learning (section 3.3).

3.1 Explicit One word Limitation (PromptEOL)

The basis for the method they propose was PromptBERT (Jiang et al., 2022), which leverages a prompt-based method to represent a sentence by feeding the model manually created templates like *This sentence: " [text] " means [MASK]*, where [text] is the placeholder for a sentence and the output vector of [MASK] token is used as the sentence embedding. This method demonstrates superior results compared to previous sentence representation methods like averaging the final hidden representations of all tokens or using the representation of the [CLS] token.

The authors of the replicated paper created a similar method, which uses two modifications, so the template then looks like this: *This sentence: " [text] " means in one word: "*

First, they added *in one word* to the prompt to constrain LLMs to embed semantic information into the next token. Secondly, they incorporated *: "* at the end of the template to prevent the model form generating punctuation as the next token, as *This sentence: "* is used to indicate the input of a sentence. This approach improved results for all OPT models (Zhang et al., 2022) and matched or outperformed BERT in some tasks.

3.2 In-context Learning

In-context learning is widely utilized as an effective method to help LLMs understand problems. This method improves comprehension of inputs and outputs by directly adding a few examples to the prompts. But when addressing the challenge of sentence embeddings, it is essential to independently project sentences into vectors based on their semantic information. To employ in-context learning for sentence embeddings, the researchers introduce the framework illustrated in Figure 1 to automatically construct demonstration sets and search for demonstrations, aiming to enhance LLMs' sentence embeddings.

The goal of this framework is to create sentence-word pairs. To create a dataset for training, the authors used two different methods, which are described in section 4.1.

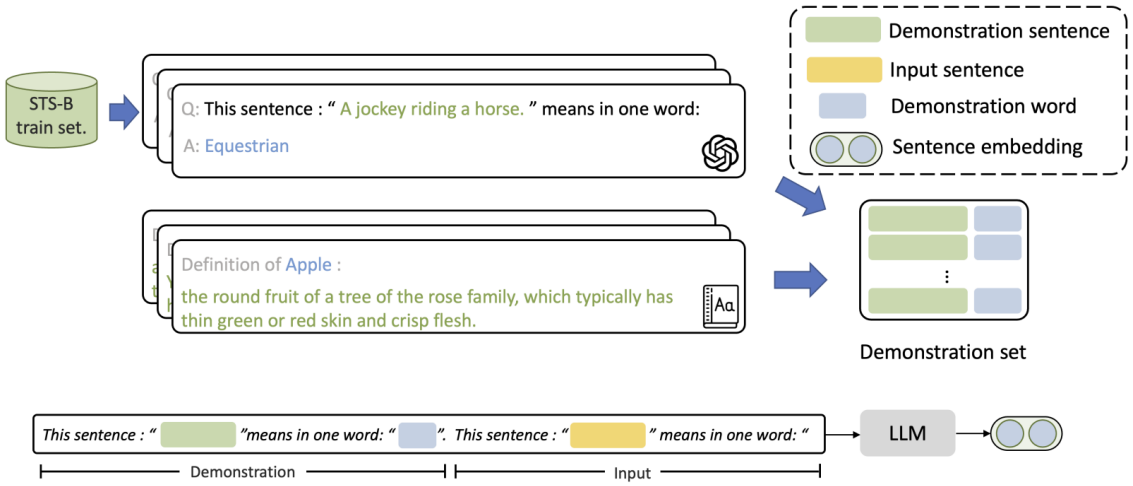

Figure 1: An illustration of in-context learning based sentence embeddings. Green denotes a demonstration sentence, while blue denotes the demonstration words. The corresponding colored rectangles indicate their respective slots in the template.

3.3 Contrastive Learning

Contrastive learning on LLMs has been demonstrated as an efficient way to learn sentence embeddings (Gao et al., 2022). According to the datasets, it can be divided into unsupervised and supervised setting. The authors of research focused only on supervised setting, which they trained as follows.

Following Gao et al. (2022), they used SNLI and MNLI datasets. Each sentence x_i in these datasets has a corresponding positive x_i^+ and hard negative x_i^- sentence. First, they added x_i into the template and got hidden states

$$h_{i1}, ..., h_{il} = LLM(\text{This sentence: "}x_i\text{" means in one word: "})$$

where l is the number of hidden states. Than they used the last token's hidden state as its sentence embedding $x_i = h_{il}$. So the embedding representation for triplet (x_i, x_i^+, x_i^-) is (h_i, h_i^+, h_i^-). The training objective is then as follows:

$$l_i = -log \frac{e^{cos(h_i, h_i^+)/\tau}}{\sum_{j=1}^{N}(e^{cos(h_i, h_i^+)/\tau} + e^{cos(h_i, h_i^-)/\tau})}$$

where N is the batch size and τ is the temperature hyperparameter in contrastive learning.

4 Dataset

4.1 Dataset for in-context learning

For the demonstration set, the authors' goal was to create sentence and word pairs, where the word can represent the semantic information of the sentence. They used two methods to generate these pairs.

For the first method, the researchers used ChatGPT and a dataset from the SentEval Toolkit (Conneau and Kiela, 2018). By querying ChatGPT using the template shown in Figure 1, ChatGPT outputs a one-word summary for the given sentence. The second method involves using the Oxford dictionary, from which the authors directly extract words and their corresponding definitions, using them as word-sentence pairs.

Altogether, they generated 300 pairs, 100 using the first and 200 using the second method. All these pairs are provided by the authors in Jiang et al. (2023).

4.2 Dataset for evaluation

Following some previous works (Gao et al., 2022; Jiang et al., 2022), the authors used the SentEval toolkit (Conneau and Kiela, 2018) to conduct experiments on the STS datasets (these include STS tasks from 2012 – 2016). Sentence pairs in each STS dataset are scored from 0 to 5 to indicate semantic similarity. Spearman correlation is used as a metric to evaluate the correlation between the cosine similarity of sentence embeddings and the golden similarity scores.

5 Replicated experiment

In this section, we describe implementation details and look at the original vs. the replicated results.

5.1 Implementation Details

Since we were trying for the best possible replication, we tried to leave as many implementation details as possible unchanged. For the setting without fine-tuning, we used OPT (Zhang et al., 2022). In the original article, the authors went up to the 66B parameter version. Due to our computational constraints, we limited our experiments to models with 125M to 6.7B parameters. We used the template described in section 3.1 for all models.

For each model, we chose only one demonstration from the collected dataset as their demonstration in template for evaluation. We also used quantization to 4 bits using `bitsandbytes` (`https://huggingface.co/docs/bitsandbytes/main/en/index`) with 4-bit normalfloat and double quantization. In the original article, the authors mention results with and without quantization (16bit weights).

For the setting with fine-tuning, the researchers used QLoRA (Dettmers et al., 2023) to fine-tune OPT and LLaMA with contrastive learning. All details can be found in the original article (Jiang et al., 2023). The authors also make their fine-tuned trained models available at `https://huggingface.co/royokong`. Due to our computational constraints, we decided not to replicate the fine-tuning, but rather to use these already fine-tuned models.

5.2 Results

Comparing the original vs. the replicated results is one of the most important parts of this report and replication challenge. Detailed results are presented in the tables below.

STS tasks with fine-tuning Table 1 shows the results achieved by fine-tuning with PromptEOL on the supervised dataset. As we can see, replicated and original results are only a little different (note that, as stated before, in this part of the replication study, we are actually using models fine-tuned by the authors of the original paper, so this result is unsurprising). We also added some other models to compare our results against.

Research	Params	STS12	STS13	STS14	STS15	STS16	Avg.
Fine-tuning on supervised datasets with contrastive learning							
Original	1.3 B	79.01	89.26	84.10	88.30	84.62	85.06
Original	2.7 B	79.49	89.64	84.80	89.51	85.91	85.87
Original	6.7 B	80.14	90.02	84.94	89.78	85.84	86.14
Replicated	1.3 B	78.90	88.93	84.14	88.23	84.97	85.03
Replicated	2.7 B	79.16	89.43	84.78	89.37	85.85	85.72
Replicated	6.7 B	79.98	89.97	84.99	89.60	85.69	86.05
Comparison to other BERT-based methods							
SBERT-NLI	220M	72.27	78.46	74.90	80.99	76.25	76.57
SimCSE-RoBERTa	123M	76.53	85.21	80.95	86.03	82.57	82.26
PromptRoBERTa	123M	76.75	85.93	82.28	86.69	82.80	82.9
SGPT	5.8B	74.28	85.35	79.21	85.52	82.54	81.38
ST5-Enc	4.8B	80.10	88.75	84.70	88.86	85.17	85.52

Table 1: Performance of original authors' method on STS tasks with fine-tuning. SBERT-NLI and SimCSE-RoBERTa (Gao et al., 2022), PromptRoBERTa (Jiang et al., 2022), SGPT (Muennighoff, 2022), ST5-Enc (Ni et al., 2021)

We can also see these results on the Figure 2. Since the lines are very similar and you couldn't see them both, we made the replicated one dotted. The Bert prompt and the ST5-Enc had only one measurement so we decided to plot them as a line.

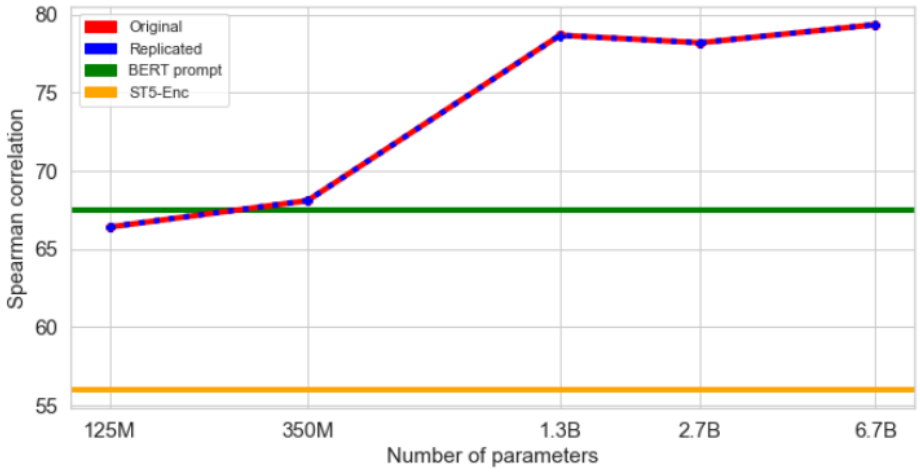

Figure 2: Performances of OPT PromptEOL (original and replicated) in all STS tasks averaged. Green and orange lines represent the result of Bert prompt and ST5-Enc.

STS tasks without fine-tuning Table 2 shows the results without fine-tuning. As we can see, the replicated and the original results are almost the same. Also in this case we added some other models to compare our results against.

Research	Params	STS12	STS13	STS14	STS15	STS16	Avg.
Without fine-tuning with in-context learning							
Original	125 M	61.02	71.00	59.75	69.67	70.52	66.39
Original	350 M	64.14	72.45	62.58	71.05	70.18	68.08
Original	1.3 B	73.45	82.55	73.11	83.63	80.60	78.67
Original	2.7 B	68.50	84.73	74.62	82.23	80.87	78.19
Original	6.7 B	70.23	84.64	76.08	83.73	82.06	79.35
Replicated	125 M	61.05	71.03	59.75	69.68	70.51	66.4
Replicated	350 M	64.21	72.43	62.57	71.04	70.17	68.08
Replicated	1.3 B	73.46	82.55	73.10	83.62	80.59	78.66
Replicated	2.7 B	68.50	84.72	74.62	82.24	80.87	78.19
Replicated	6.7 B	70.23	84.64	76.08	83.72	82.06	79.35
Comparison to other BERT-based methods							
BERT avg.	110M	30.87	59.89	47.73	60.29	63.73	52.5
BERT prompt	110M	60.96	73.83	62.18	71.54	68.68	67.44
ST5-Enc	4.8B	34.97	60.19	47.59	66.40	70.62	55.95

Table 2: Performance of original authors' method on STS tasks without fine-tuning. BERT avg. (Gao et al., 2022), BERT prompt (Jiang et al., 2022), ST5-Enc (Ni et al., 2021)

We have also shown the comparison of different models on the figure 3. For both original and replicated OPT PromprEOL models, we averaged all measurements, i.e. OPT models with 1.3B, 2.7B and 6.7B parameters.

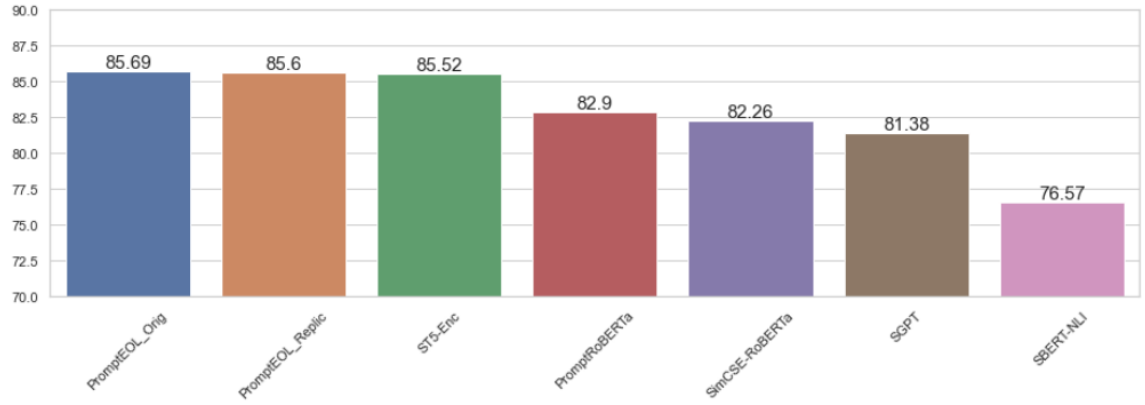

Figure 3: Average Spearman correlation on all STS tasks with fine-tuning with different models

6 Conclusion & Discussion

In this study, we successfully replicated the original research, which was focused on exploiting Large Language Models (LLMs) to improve sentence embeddings. We replicated the original authors' new sentence embedding method PromptEOL and shown that the improvement described in the original paper can be replicated. Furthermore, we also replicated in-context learning, which also improved performance of the LLMs and led to better results. We obtained results that are very similar to the ones published in the original paper. Only small differences were spotted in the results and these can be attributed to randomness. Our successful replication confirms the effectiveness of the methods used to improve sentence embeddings using LLMs.

In conclusion, our attempt to replicate the article turned out very positive and great. Through our replication efforts, we have confirmed the result, that were presented in the original article. Our research emphasizes the importance of open science practices, guaranteeing that research results can be reproduced and expanded upon to further advance knowledge in the field.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen tau Yih, Yoon Kim, and James Glass. 2022. Diffcse: Difference-based contrastive learning for sentence embeddings.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2022. Simcse: Simple contrastive learning of sentence embeddings.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2023. Scaling sentence embeddings with large language models.

Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. 2022. Promptbert: Improving bert sentence embeddings with prompts.

Jared Kaplan, 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.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.