Interpreting Prompt-Tuning Across SuperGLUE Tasks

Finn Dayton
Stanford University
finniusd@

Oscar O'Rahilly
Stanford University
oscarfco@

Eric Werner
Stanford University
ewern@

Abstract

Prompt-tuning is an efficient, low-cost way of adapting a large foundation models to new downstream tasks without retraining the model and updating its weights. Recent work has demonstrated this technique can lead to strong performance on tasks not originally pre-trained on and for a fraction of the resources. In the paper, we explore the interpretability of tuned prompts both at an embedding level and a semantic one. We then look at prompt stochasticity under different conditions. Lastly, we investigate the zero-shot task transferability of learned prompts. We find that understanding the inner working of prompts leads to better design choices for prompt-tuning in general.

1 Introduction

In recent years, large language models (LLMs) have grown exponentially in size, with current state of the art models, like GPT-4 (OpenAI, 2023), possibly exceeding over 100 trillion parameters . While these extremely large LLMs possess human-level capabilities at a number of natural language tasks, due to their size it is extremely computationally expensive to train them on new, unseen tasks. Parameter efficient finetuning methods, therefore, have become a very effective way to train large models for unseen tasks in a way that is fast and inexpensive.

Prompt-tuning (Lester et al., 2021) is a simple yet effective parameter efficient finetuning method that only trains and updates a prepended series of tokens to a prompt for a pretrained model. This way, one can use a pretrained model whose weights are frozen, and train and update a smaller set of prompt parameters for each downstream task instead of fully finetuning a separate model. As a result, prompt-tuning can be seen as a computationally inexpensive method for training large language models on downstream tasks.

To further illustrate how prompt-tuning works, consider the following example. Suppose we have a large language model and would like to train it on a text summarization dataset. Using prompt-tuning, we add a small set of tunable tokens that are prepended to the input (the text to summarize) of the model. Then, during training, we use the same objective as we would in the setting where we are finetuning the entire model, however, this time the model parameters are completely frozen and we only update the prepended tokens. Through this optimization procedure, we are left with a prompt that has been optimized for the specific task. (Lester et al., 2021) and many others have shown that prompt tuning can achieve or even beat a model finetuned on that task.

Extensive work has been conducted on prompt-tuning's efficacy when it comes to improving model performance on unseen tasks (Shin et al., 2020), (Li and Liang, 2021) (Lester et al., 2021) (Vu et al., 2022). We noticed, however, that limited work has been conducted on interpreting the prompts on an embedding and semantic level once they are tuned. Extracting semantic meaning from tuned prompts could be extremely useful. For instance, it might inform humans on how better to design prompts themselves, showing us what is truly useful or not useful to include in a prompt. Moreover, analysing and comparing the embedding of soft-prompts could be extremely useful also. For instance comparing the similarity between soft-prompts trained on different tasks could inform us on the overall similarity between tasks.

In this paper we hold the hypothesis that having a better understanding of the soft-prompts generated through prompt-tuning at the embedding and semantic levels will help us design more effective prompts. Moreover, we hypothesize that better understanding the stochasticity of generated prompts across different tasks can help build intuition and lead to more stable training practices. Lastly, a deeper analysis on the embedding level will help in predicting the zero shot transferability of prompts.

In order to test these hypotheses, our paper will look at interpreting these soft-prompts on a subset of datasets from SuperGLUE: CB, RTE, COPA & WSC. First we will explore the semantic meaning of the tuned prompts. Then we will look at the stochasticity to help how the learned prompts vary across separate trains for each task. Finally, we will turn our attention to the specific information encoded in these prompts by looking at the zero shot prompt transferability of the generated soft-prompts.

We show that indeed, these three investigations–semantic meaning of learned prompts, stochasticity and zero-shot transferability–do shed light on the innerworkings of prompt tuning and are helpful both for building intuition and building good prompt tuning algorithms.

2 Prior Literature

(Shin et al., 2020) introduces the concept of prompt-tuning. The authors show that large, pretrained language models have the inherent capability to perform sentiment analysis and NLI (natural language inference)–tasks not specifically trained on–*without* finetuning on them. They demonstrate that this is possible with carefully worded prompts that provide the model with enough information to successfully complete the new task. Writing good prompts, however, is hard. The authors propose a scheme for *generating* prompts for a specific task and given model. The prompt takes the format sentence[T][T][T][T][T][P], where sentence is an example query for the new task, and the T's are "trigger words", words learned through gradient search to maximize the correct output. They essentially inject more rich information into the prompt to assist in retrieving the correct answer. P is the end token and is set to "[MASK]". After feeding the prompt into the MLM, it produces a prediction for [MASK], which is the answer. This paper is the main inspiration for our analysis of learned prompts. It was the first to show that prompt-tuning could match finetuning performance.

(Li and Liang, 2021) builds on (Shin et al., 2020). (Shin et al., 2020) uses discrete promoting which is limited to real words. (Li and Liang, 2021) uses learns continuous embeddings in the prefix for more expressively. The models used are GPT-2 and BART for table-to-text and summarization, respectively. The authors also show that in full-data settings, prefix tuning and finetuning get comparable results but that in low-data settings, prefix tuning outperforms finetuning. The authors emphasize that the main benefit of prefix-tuning over finetuning is savings in computation and memory: a model does not need to be duplicated, trained and stored to specialize to certain tasks. Lastly, the authors go one step further by optimizing the activations of *all* layers by altering embeddings deeper in the network. The authors also ablate placing the learned embeddings at the front of the prompt (prefix) and between two chunks of the prompt (infix) and find the former works better. Finally, the authors demonstrate that a longer prefix means more trainable parameters, and therefore more expressive power. Not being limited to domain of real words and instead tuning continuous embeddings was the main contribution of this paper to our work.

(Lester et al., 2021) can be seen as a simplification of (Li and Liang, 2021). They emphasize that better-than-finetuning performance can be achieved by just tuning the prompt embedding, i.e. with no intermediate layers in the model as (Li and Liang, 2021) also trains. They freeze the entire pre-trained model and only allow an additional k tunable tokens per downstream task to be prepended to the input text. The authors provide a an end-to-end learned approach that outperforms GPT-3's few-shot learning by a large margin. This paper helped us simplify our approach: instead of adding trainable layers inside the model, we can focus solely on training the prompt itself.

(Vu et al., 2021) presents soft-prompt Transfer (SPoT), which builds on the PromptTuning architecture presented by (Lester et al., 2021). SPoT first trains a prompt on one or more "source" tasks, and then uses the resulting prompt to initialize the prompt for a "target" (downstream) task. In other words, the prompts are first trained on the source tasks and then further adapted for the target task. This is analogous to transfer learning where a model (e.g. Resnet) is trained to classify CIFAR images and then finetuned to classify cancer images. This allows the prompts to carry over some of the knowledge learned from the source tasks, which can potentially improve the model's performance on the target task. Further, the authors design an efficient retrieval algorithm to identify source tasks that will likely yield positive transfer for new target tasks. To better understand this, authors conducted a systematic study of the T5 model using 26 NLP tasks in 160 combinations of source and

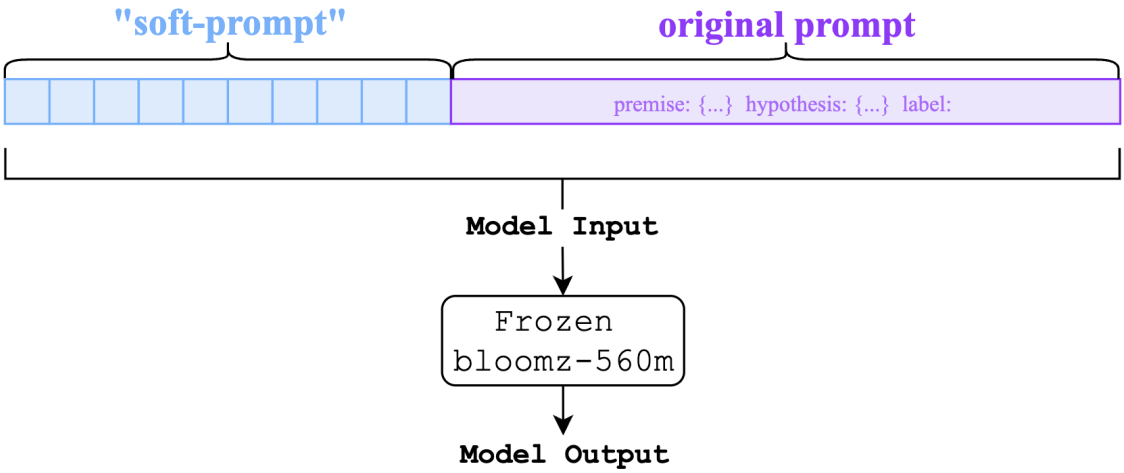

Figure 1: Prompt-Tuning Diagram. Example for RTE Dataset.

target tasks. The results are impressive: SPOT matches or outperforms standard model finetuning on the Super GLUE benchmark, while using up to 27,000× fewer task-specific parameters. This paper was our inspiration for examining the transferabilty of prompts. Unlike this paper, however, we do not train the prompt on the new task and are purely interested in the zero-shot performance between similar and dissimilar tasks.

3 Data

We trained soft-prompts for the following four datasets from SuperGLUE (Wang et al. (2020)): CB, RTE, COPA and WSC. We used datasets from SuperGLUE because it is a popular benchmark that contains challenging NLU tasks. We picked this particular subset because the model we chose, *bloomz-560m*, was finetuned on the other SuperGLUE datasets, meaning it had seen all their validation sets before.

3.1 CB: CommitmentBank

The CommitmentBank dataset annotates whether the hypothesis logically follows ("entails") from the premise. The three output labels are "entailment", "contradiction", or "neutral". "entailment" means the claim logically follows from the premise, "contradiction" means the claim goes against what is stated in the premise, and "neutral" means the claim could be possible based on the premise, but it doesn't definitively follow from it.

Example data
premise:`"B: and, you know, they just love kittens. A: Yeah. B: They just are fascinated. A: Oh, yeah. B: So she doesn't know that this is a cat yet."`
hypothesis: `this is a cat`
label: `0 (entailment)`

3.2 RTE: Recognizing Textual Entailment

The RTE dataset annotates "entailment" or "not entailment" on a premise + hypothesis sentence. This is like CB, but without an option for contradiction.

Example data
premise:`"There has been a lot of concern over the rise of drug-resistant bacteria."`
hypothesis:`"Bacteria is winning the war against antibiotics."`
label: `1 (not_entailment)`

3.3 COPA: Choice of Plausible Alternatives

COPA is a good measure of a causal reasoning of our model, as the *question* conditions the model to predict whether choice 1 or 2 was a *cause* or *effect* of the premise. Each premise is given this question and two choices, and the model must predict which of the two choices caused/effected the premise.

Example data
premise: `"My eyes became red and puffy."`
choice 1: `"I was sobbing."`
choice 2: `"I was laughing."`
question: `"cause"`
label: `0 (choice 1)`

3.4 WSC: Winograd Schema Challenge

The WSC dataset contains examples of which *noun* a *pronoun* refers to. Each example gives a sentence, a noun (and index of noun within sentence), pronoun (and index), and a T/F label of whether the pronoun refers to noun.

Example data
text:"`Jane gave Joan candy because she was hungry.`"
span1: `(Jane, 0)`
span2: `(she, 5)`
label: `0 (False)`

4 Model

We decided to use the `bloomz-560m` model from Hugging Face as our model (Muennighoff et al., 2023). `bloomz-560m` is model capable of following human instructions in dozens of languages zero-shot. It is the `bloom-560m` base model fine-tuned on HuggingFace's crosslingual task mixture (xP3). It is capable of crosslingual generalization to unseen tasks & languages.

We settled on `bloomz-560m` for a few reasons. The first is due to its size. As Lester et al, mentioned, prompt-tuning performance improves drastically with larger parameter models. Thus we wanted to use a model that took full advantage of our available compute space. This is why we did not pick a smaller model such as Bert or RoBerta. Secondly, we thought `bloomz-560m's` ability to represent many languages would provide some interesting insights in our interpretability work. Specifically, we wondered if for certain tasks, different languages might appear in our tuned prompts.

5 Methods

5.1 Semantic Interpretability

For our first experiment, we wanted to explore the semantic interpretability of our generated soft-prompts for each of our tasks, to see if we could learn anything about what constitutes an optimal prompt.

Each soft-prompt we trained can be viewed as an embedding matrix, P, of size $n \times 1024$, where n refers to the number of tokens in the soft-prompt and 1024 the embedding size used by the `bloomz-560m model`. To extract the semantic meaning of a soft-prompt for a particular task we first optimize the prompt on a task's train split. We employed an early stopping strategy in our training procedure as we did not want the soft-prompts to overfit to our training set. Thus, we ended training once the evaluation loss stopped decreasing, corresponding to a patience value of 5.

Once training finishes, we look at each row (which corresponds to a particular token embedding) in the resulting soft-prompt matrix, P, one at a time. Then, we compare the token embedding with every token embedding in the `bloomz-560m` dictionary, using cosine similarity, and return the most similar tokens for each row in P. In this way, we are able to reconstruct the semantic meaning of any soft-prompt.

We use the cosine similarity metric because the tuned token embedding will most likely not directly correspond to an actual token in `bloomz-560m's` dictionary due to the continuous nature of token embeddings.

An important point to note about our method is that we initialized our soft-prompts such that they were "good" prompts for their respective tasks rather than simply randomly intializing them. We found that randomly initialized prompts would both take significantly longer to converge and would yield much lower accuracy.

We came up with the 'good' prompts ourselves and have listed the them below for each task as well as the number of tokens they are made up of:

- CB: *"Does the hypothesis entail, contradict or is neutral about the premise?"* (15 tokens)
- RTE: *"Does the hypothesis entail, or not entail the premise?"* (14 tokens)
- COPA: *"Pick the correct choice given the premise and the question."* (12 tokens)
- WSC: *"Given the following text, decide whether the span1_text is referenced by the pronoun in span2_text"* (20 tokens)

5.2 Stochasticity

In this experiment, we explore the variability in final soft-prompts across independent training runs and between the four tasks, CB, RTE, COPA and WSC.

Each training run has inherent stochasticity from the Adam optimizer, which uses randomly initialized momentum vectors. This random initialization can lead to varying final model states despite the

same initial conditions and identical hyperparameters across runs. Additionally, the inherent differences in the complexity and nature of each task (CB, RTE, COPA, and WSC) contribute to variability in the training process and final soft-prompts. Understanding this variability can provide insights into the robustness of the training process, and inform potential strategies for model optimization and performance improvement.

We ran training 4 times, each for 50 epochs, for each of the tasks. This was 16 full training runs in total.

5.3 Zero Shot Task Transferrability

Our final experiment involved analyzing and comparing the embeddings of the resulting soft-prompts for each task with one another. We hypothesize that a soft-prompt generated from one task that is similar to a soft-prompt from another task would perform well as the prompt for that other task.

To go about testing this, we defined a similarity metric between two soft-prompts as the Frobenius Norm of the difference between the two soft-prompt embedding matrices. For instance, consider two soft-prompt embedding matrices, A and B, we calculate the similarity score as follows, where a lower score indicated greater similarity

$$Similarity = \|A - B\|_F$$

The Frobenius Norm of a matrix is a measure of its magnitude, similar to how the Euclidean norm measures the magnitude of a vector. If X is a $m \times n$ matrix, the Frobenius Norm of X can be defined mathematically as as follow, where x_{ij} denotes the elements of matrix X:

$$\|X\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|x_{ij}|^2}$$

After we find the similarities between all the soft-prompts, we take each learned soft-prompt and use it as the prompt for another task and run inference on that task's validation set. For instance, we take the soft-prompt learned from RTE and use it as the prompt for WSC and perform inference on the WSC validation set. We then report the change in accuracy that using the RTE prompt has over simply running the WSC validation set with no prompt. For all the datasets in this paper, we use the exact match metric for reporting accuracy.

Finally we look to see if there is any correlation between the similarity of embeddings and the cross-task prompt performance.

6 Results

Note: All accuracy metrics presented are the exact match accuracy for each task on its respective datatset.

6.1 Semantic Interpretability Results

Figures 2, 3, 4 and 5 show the results of the semantic extraction of the soft-prompts for each of our four datasets.

Each row in the figures contain the 3 most similar extracted tokens for the corresponding soft-prompt token, with the most similar to the left and least similar to the right. You should read from top to bottom. We decided to provide the 3 most similar tokens to highlight that no one token completely conveys the meaning of a particular soft-prompt embedding.

The results are extremely interesting. At first glance the soft-prompts seem to have diverged drastically from their original starting points. We see that different languages have appeared and even non-language tokens like the new line character that appears in the COPA extraction. However, upon closer insepction we see that in CB, RTE and COPA, certain tokens have remained constant from the original prompts (highlighted in red). This is particularly interesting since other tokens diverge and become very different from their original, indicating that the tokens that stay the same must really help the model in making a prediction.

Moreover we observe that the tokens switch to other languages, such as French, Arabic and Farsi, which is not surpising considering `bloomz-560m` was trained on a wide variety of languages. However, what is particulary interesting is we see certain key words present in the original prompt actually translated into different languages as shown in the CB prompt extraction in figure 2. The sixth from the bottom row contains a Persian word followed by a Punjabi word. Both of these words mean *contradiction* in English which was one of the words in the original CB prompt.

The final interesting thing we see are the words highlighted in orange. Across three of the four tasks (RTE, COPA, WSC) we all see tokens that have meanings that convey accuracy. In RTE we have the English word "accurate", in COPA we

have the chinese characters for "accurate" and in WSC we see the Kannada word for "correct". This shows that the model likes to be explicitly told to be accurate when completing one of these tasks.

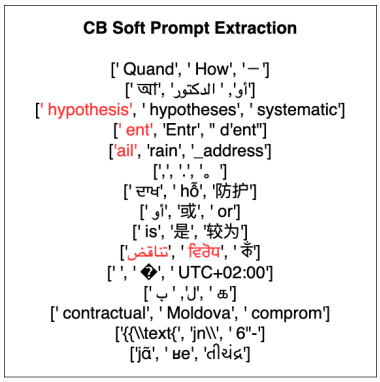

Figure 2: CB Prompt Extraction
"Does the hypothesis entail, contradict or is neutral about the premise?"

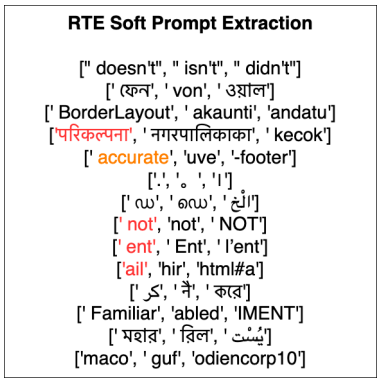

Figure 3: RTE Prompt Extraction
"Does the hypothesis entail, or not entail the premise?"

6.2 Stochasticity Results

For each of the four tasks, we ran four duplicate trainings. Each learned prompt is a large 2d matrix. Figure **??** shows the result of projecting each high-dimensional learned prompt into a location on a two dimensional map using t-distributed stochastic neighbor embedding (T-SNE). Each point in Figure **??** is learned prompt, and the prompts are color code for each task. Prompts of the same task are connected via color-coded lines for visual effect.

Figure 6 highlights the first interesting result of this section: learned prompts for the same tasks are all nearly equidistant. This shows training to be stable.

We then looked at the variance within each class for the learned prompts. Table 1 presents these results.

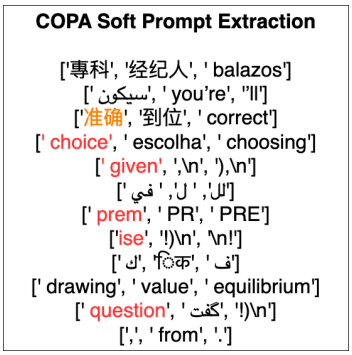

Figure 4: COPA Prompt Extraction
"Pick the correct choice given the premise and the question."

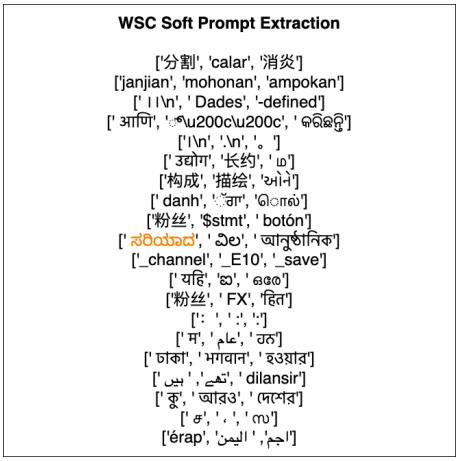

Figure 5: WSC Prompt Extraction
"Given the following text, decide whether the span1_text is referenced by the pronoun in span2_text"

These findings highlight another interesting result. Harder tasks have higher variance among the learned prompts than easier ones. We quantify the relative difficulty of our four tasks based on the median validation accuracy reported on each in the original `bloomz-560m` paper (Muennighoff et al., 2023). Table 2 shows these.

The table reinforces an interesting intuition: CoPA, the task with the most variance in soft-prompts in Table 1 also has biggest delta between its median accuracy and maximum accuracy in Table 2. Figure 6 gives a visual intuition of this, with the largest rectangle being formed by the CoPA soft-prompts, indicating the widest swings in optimal prompts.

6.3 Zero Shot Task Transferability Results

The results from this experiment show that zero shot task transferability using learned soft-prompts on our datasets hold weight. In Figure 7 we show

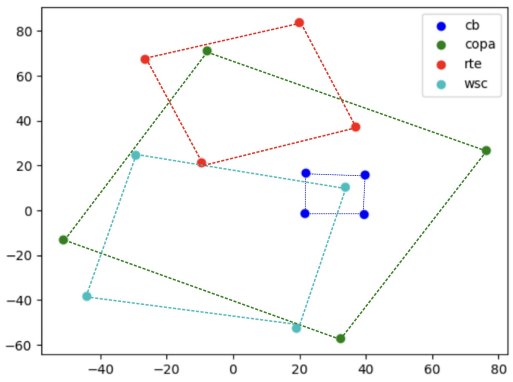

Figure 6: Duplicate Trains Produce (Nearly) Equidistant Prompts

	CB	RTE	COPA	WSC
Avg.	119.2	107.7	111.2	135.3
Var.	1.31	3.5	**38.5**	3.8

Table 1: Average Norm and Variance of Learned Prompts Per Class

the similarities of the final prompt embedding matrices using the frobenius norm metric defined previously. Figure 8 shows the change in accuracy over the test set when we use a soft-prompt trained on one task on another task. For instance, the top right cell of Figure 8 shows that there is a +0.4 accuracy increase when we evaluate copa on our model using the soft-prompt generated in CB versus when we simply run our model on copa with no prompt.

We can clearly see that for low values in Figure 7 (i.e more similar tasks) there are larger improvement gains when we run the soft-prompts trained on those tasks on the other dataset. For instance take the first row of Figure 8, namely the row which demonstrates the performance gains of using the CB soft-prompt on the other four datasets. As is to be expected, the largest performance gain is when the CB soft-prompt is used on the CB validation split. Although we also see improvement gains on rte, wsc and copa and moreover there is a rough trend corresponding to their forbenius norm similarity.

7 Analysis

We believe our results were very helpful in gaining a deeper understanding of the final soft-prompts generated through prompt-tuning.

From a semantic standpoint, our experiments demonstrate that the model clearly values certain words in prompts. This could be seen by the fact

Task	Median Accuracy	Maximum Accuracy	Delta
CoPA	55%	67%	**12%**
CB	43%	61%	8%
WSC	51%	52%	1%
RTE	53%	54%	1%

Table 2: `bloomz-560m` Original Paper (Muennighoff et al., 2023) Results on Our Four Tasks

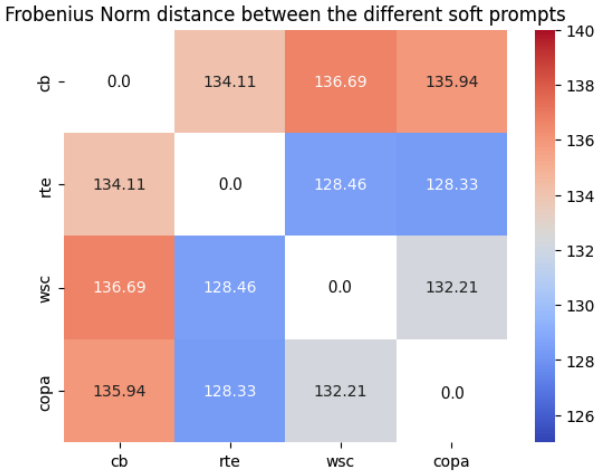

Figure 7: Soft Prompt Similarity Via Frobenius Norm

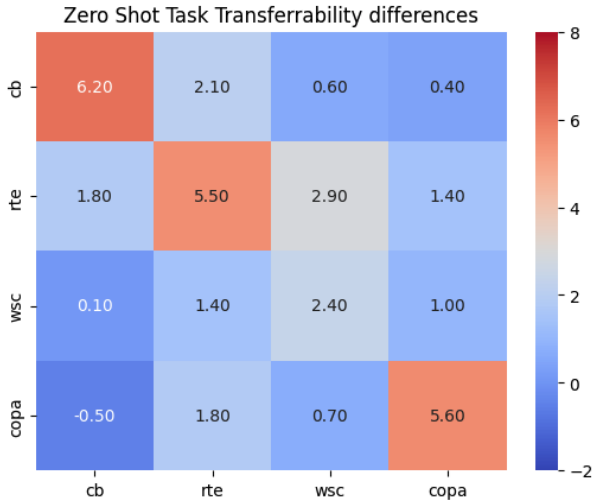

Figure 8: Zero Shot Task Transferrability Exact Match accuracy differences

that across training, they never disappeared from the prompt, for instance the word *'entail'* in both the CB and RTE prompts. For both CB and RTE, the notion of entailment is the central task of the dataset and so it makes sense that this word would be highly valuable to the model. Moreover, there appear to be other specifications within prompts that the prompts trained on our tasks converged

on, such as the notion of *'accuracy'*. It must also be noted that these prompts do display less clear semantic meaning than we had originally thought, which is a nice reminder that these models operate in fundamentally different ways to humans and that interpretaing what these models are doing under the hood is still an extremely difficult thing to do.

Our results regarding the stochasticity of soft-prompts are also enlightening. Before conducting these experiments we were very curious as to whether certain tasks might have many different 'optimal' prompts. Our experiments with stochasticity demonstrated while the soft-prompts for the same task across separate trains are not exactly the same, they tend to be equidistant from each other. This was true for each of the four tasks. Further, the variance, or distnace between prompts was noticeably different for each task. We drew a correlation with these findings and the performance variance–defined as the difference between median and max accuracy–original observed in the `bloomz-560m` paper (Muennighoff et al., 2023). These findings are important because they show soft-prompt tuning is relatively stable, but the variance in outcomes is correlated with the unique challenges and characteristics of each task.

Finally, our comparison of the soft-prompts' zero-shot transferability clearly demonstrates that similar trained soft-prompts can be transferred across tasks and yield perforamnce in accuracy over using no prompt at all. This indicates that as well as learning a task specific description, these soft-prompts also encode other meanings relating to general NLU tasks.

8 Conclusion

In this paper, we investigated the interpretability of the soft-prompts generated through prompt tuning. We showed that interpretaing the semantic meaning of soft-prompt is very difficult. We showed, however, that in some circumstances it can help us understand aspects of prompts that models find beneficial such as notions of accuracy or key definitions needed for the task at hand. Furthermore, we found for our tasks that prompt tuning yields fairly consistent results, but more difficult tasks lead to more variance in the final prompts. Finally, through comparison at the embedding level, we showed that soft-prompts can carry a lot of generalized NLU information as well as task specific information. This was evident by the success of the various soft-prompts on the zero-shot prompt transferability challenges. Overall, we believe that our results shed light on the inner workings of prompt-tuning and are helpful both for building intuition and building good prompt tuning algorithms.

Known Project Limitations

Our findings are based on experiments with a specific set of datasets from SuperGLUE and on `bloomz-560m`. While these datasets cover a wide range of tasks, they may not be representative of all possible natural language processing tasks. As such, the learned prompts might not display the same level of transferability in tasks that are significantly different from those in our study. We encourage further studies that extend our analysis to diverse and more complex tasks and on other pretained language models.

This same limitation extends to our analysis of learned-prompt stochasticity. These results are also dependent on the random initialization choice of the learned prompt and of the optimizer (Adam in our case). We urge ablations are performed on both of these in future research.

For our analysis of zero-shot transferability, it should be noted that while we hypothesize that a deeper analysis at the embedding level will help in predicting the zero-shot transferability of prompts, the performance of these prompts in a zero-shot scenario might vary greatly depending on the complexity and novelty of the unseen tasks. Therefore, our findings should be considered as exploratory in this context and used as a basis for future investigations.

Authorship Statement

Oscar: wrote the initial prompt interpretability and task transferability code and imported the models from HuggingFace as well as writing the introduction and analysis.

Finn: wrote the code to test learned prompt stochasticity and contributed to the methods section well as project limitations and the abstract.

Eric: generated figures and visualizations, proofread the paper and helped write the stochasticity code. Wrote the dataset and models section.

Note: For a majority of the project, the team worked together in a group, doing pair programming and writing the paper.

References

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

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

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning.

OpenAI. 2023. Gpt-4 technical report.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV au2, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2021. Spot: Better frozen model adaptation through soft prompt transfer. *CoRR*, abs/2110.07904.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2020. Superglue: A stickier benchmark for general-purpose language understanding systems.