Are soft prompts actually not interpretable?

Aleksandra Bakalova
Saarland University
alba00013@stud.uni-saarland.de

Abstract

In this work we explored the geometry of soft prompts trained with prompt tuning for different tasks and with different initializations. We located the trained tokens as well as some words from the vocabulary of the model in the embedding space. We found that while the initialization is likely to play a more important role in the location of the trained token, than the task it is trained for, the location of the prompt token is still highly dependent on the target task.

All the code is available on github [1]

1 Introduction and related work

The trend to train Large Language Models (LLMs) with increasing number of parameters poses new challenges to the community. While these models are proven to be very useful and outperform older models on various benchmarks, scaling models further requires more resources for training, inference and adapting the models to particular tasks.

There exist several methods that efficiently deal with the costs of adapting the model for a downstream task [4, 5, 2]. In this work we will focus on one of the most popular methods named *prompt-tuning* [4]. To adapt the model for a specific task using prompt tuning, one needs to prepend every input to the model with a selected number of tokens. These tokens are called *soft prompts*, and they are the only parameters that are trained during the adaptation process.

This method is very similar to prompting in a sense that during inference the only modification we make to simply running the model on the input is prepending the input with several more tokens. The difference is that in vanilla prompting these tokens are carefully chosen tokens taken from the model vocabulary, and in prompt tuning they are not present in the model vocabulary, but instead trained for the specific task. Such similarity between these methods opens up potential opportunity to interpret the learned soft prompts as new tokens from the model's vocabulary. For example, in theory the tokens in the vocabulary that are closest to the learned soft prompt might provide an interpretation of the learned tokens. However, there is evidence that interpreting soft prompts in this way can be misleading [3, 1]. In fact, it is proven that in a neighborhood of any discrete token there exists a soft prompt that can solve the task reasonably well [3]. It is also known that the soft prompts are generally located very far from the words in the model vocabulary both in terms of cosine similarity and l2 norm [1], and thus can not be with confidence interpreted simply as new tokens in the model's vocabulary.

However, while it is true that one must take the interpretation of soft prompts with caution, it is still unclear whether the location of soft prompts in the embedding space can be interpreted and from what perspective. There still might be some specifics in the architecture, the training

[1] https://github.com/CUPalex/soft_prompts

process, or the loss surface, that could make the prompts that are actually learned by the model more interpretable than we think even in terms of their closeness to the words from the vocabulary. Moreover, the learned prompts can be interpretable in terms other than similarity to the vocabulary tokens, for example in terms of similarity to each other.

In this work we explore the learned soft prompts for several different tasks and several different initializaion in terms of their similarity to each other and to the words from the vocabulary. We attempt the question whether the soft prompts are interpetable and in what sense.

2 Experimental setup

We train the soft prompts for one model, on several datasets with several initializations of the prompts. We train 4 soft prompts each time.

2.1 Model

For all our experiments we chose the GPT-XL, 1.5 billion parameter model, [6]. This model is big enough for the prompt tuning to show good results and also basic in terms of architecture. We leave the exploration of the prompts learned by different pretrained models outside of the scope of this work.

2.2 Initialization

An important part of training the soft prompts is choosing the initialization [4]. We choose to initialize the soft prompts with some words from the vocabulary of the model. We choose 10 simple words that are tokenized by the model as one token. Then each initialization of the prompt is a sequence of one of these words repeated as many times as there are prompt tokens.

The words we used are the following: *cat, dog, bird, book, house, chair, table, pen, paper, water*.

2.3 Datasets

For our experiments we take 6 relatively simple datasets from [7] and split them into train and test. The datasets are easy enough for the model to learn them well and are also grouped into three pairs that are similar between each other. This could help us facilitate further analysis. The tasks are the following:

- **Capitalize first letter**: The input here is a single word in lowercase, and the output is the first letter of this word capitalized. The dataset contains 650 training examples and 163 test examples.
- **Lowercase first letter**: The input here is a single word written in capital letters, and the output is the first letter of this word in lowercase. The dataset contains 651 training examples and 163 test examples.
- **Country-capital**: The input here is a country, and the output is the capital of this country. The dataset contains 157 training examples and 40 test examples.
- **Landmark-country**: The input here is a landmark, and the output is the country it is located in. The dataset contains 668 training examples and 168 test examples.
- **Present-past**: The input here is a verb in present tense, and the output is the same verb in past tense. The dataset contains 234 training examples and 59 test examples.
- **Singular-plural**: The input here is a noun in singular form, and the output is the same noun in plural. The dataset contains 164 training examples and 41 test examples.

All the tasks were formatted according to the following template:

```
Input: {input}\nLabel: {label}
```

2.4 Training and evaluation setup

We train for 50 epochs with batch size 8, Adam optimizer and a linear scheduler. We optimize for cross-entropy loss on label tokens.

For evaluation we generate at most 5 tokens and count the continuation given by the model as correct if it starts with the correct label.

2.5 Achieved performance

The test accuracy achieved by prompt-tuned models is depicted in figure 2.5 and the test loss is in 2.5. In all the runs the prompt-tuned model was able to learn the task to a significant extent, and for 4 out of 6 tasks the model learned an almost optimal solution. Apart from that, there is no sign that any of the models were overoptimized for the training dataset at the end of the training, thus from now on we will look at the embeddings of soft prompt learned by the end of the training, after 50th epoch.

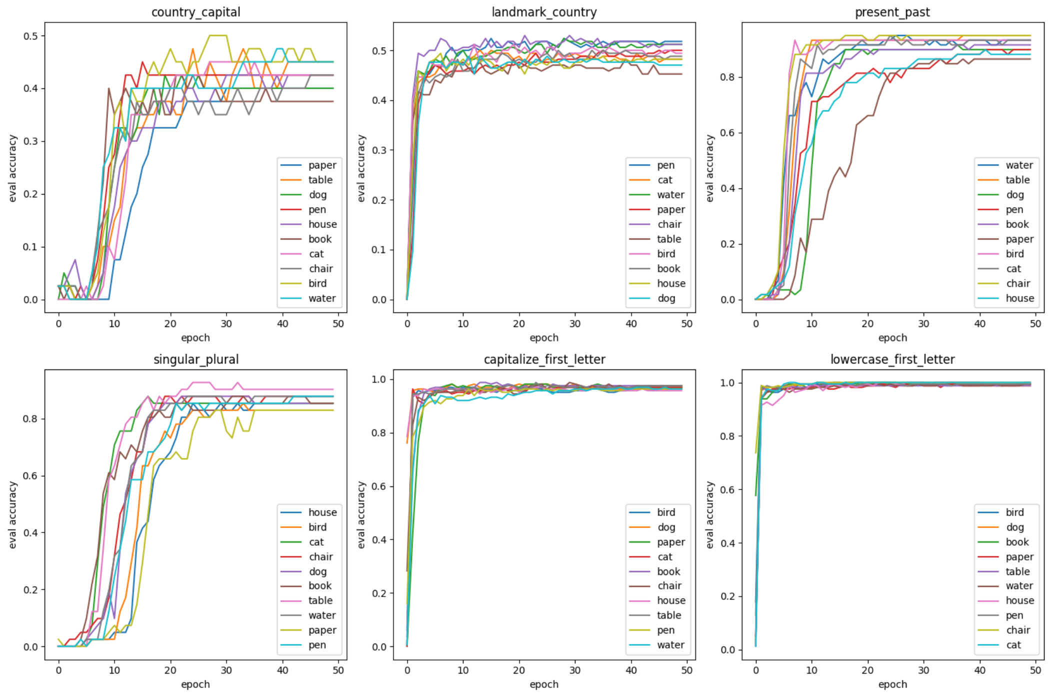

Figure 1: Test accuracy achieved by of the prompt-tuned models.

3 Analysis

3.1 What is more important: initialization or a task?

Since [1] showed that the performance of soft prompts is more sensitive to the changes in angle of the soft prompt vectors than their magnitude, we will use cosine distance to measure the similarity of two embedding vectors. Cosine distance is defined as follows:

$$cosine\_distance(v1, v2) = 1 - cosine\_similarity(v1, v2)$$

It can take values from 0 to 2, where 0 means that the two vectors are identical and 2 means that they are opposite.

In the figure 3.1 we can see the average distance between trained soft prompts of different groups. From this picture it is evident that prompt embeddings are in general located closer to each other than regular words in vocabulary. It can also be seen that embeddings of prompt

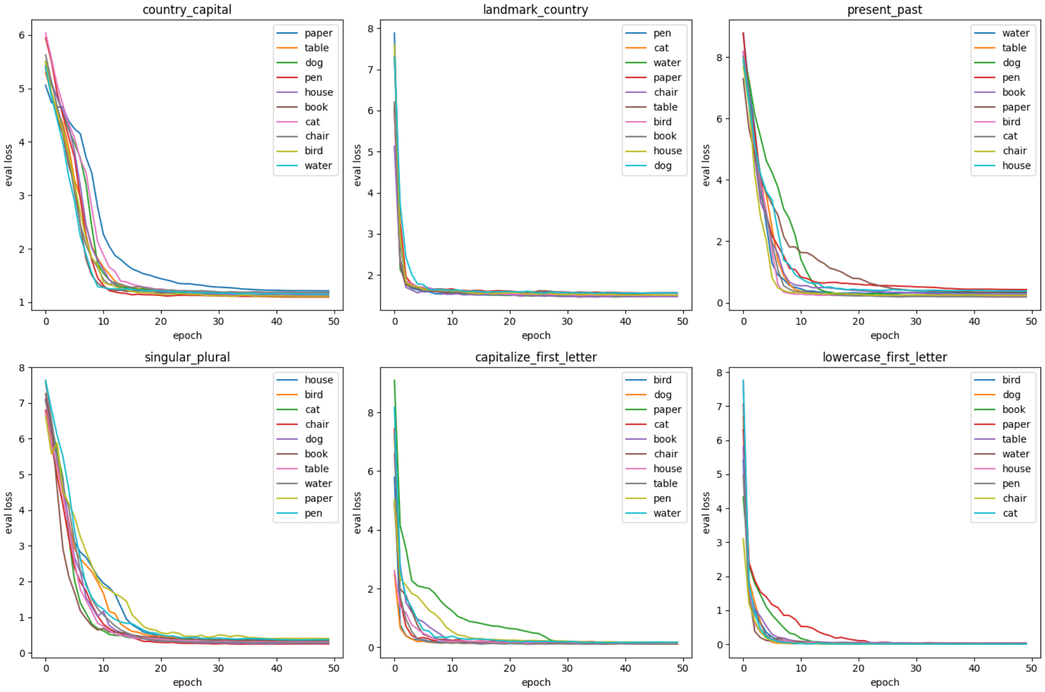

Figure 2: Loss on test dataset achieved by of the prompt-tuned models.

tokens trained for different tasks are in general closer to each other than embeddings trained with different initial words. The last claim is more evident if we look at tables 3.1 and 3.1. There it is evident that the distance between soft prompts trained with the same initial word is in general smaller than the distance between soft prompts trained for the same task. This means that a change in the initialization changes the location of a trained embedding more drastically than a change in a task it is being learned for.

In figure 3.1 it can also be seen that after projection of soft prompts into 2D space, the prompts seem to be grouped by the initial word, and not by the task they were trained for. However, this visualization should be taken with caution, since the algorithm used for projection is not precise.

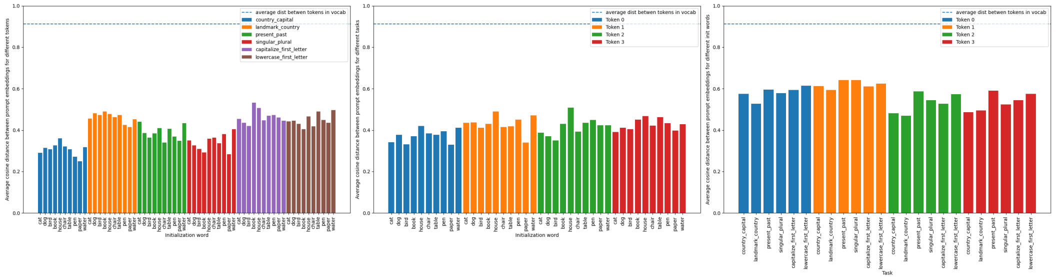

Figure 3: Average cosine distance between prompt embeddings in different groups. In the left graph the distance is measured and averaged between embeddings of soft prompts in different token positions, in the middle between embeddings of soft prompts trained for different tasks and in the right between embeddings of soft prompts trained with different initial words. The blue line indicates average distance between a sample of 100 random tokens from the vocabulary. The same figure plotted for l2 distance can be found in appendix A.

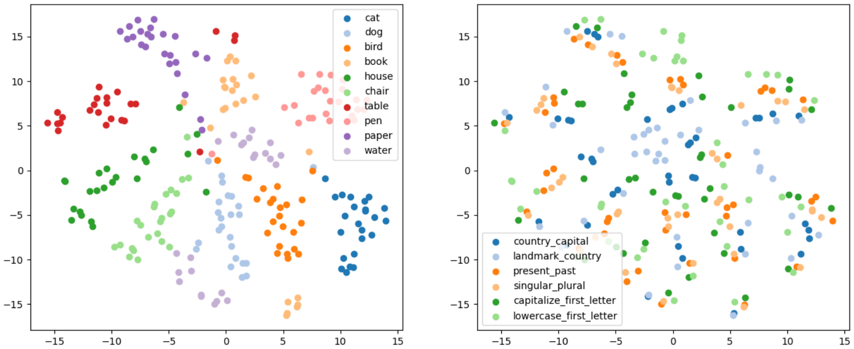

Figure 4: All the soft prompts trained for all the tasks with all initializations projected to 2D using TSNE and colored according to the initial word (left) or task (right).

Initial word	Average distance
cat	0.546
dog	0.547
bird	0.528
book	0.567
house	0.612
chair	0.544
table	0.580
pen	0.566
paper	0.506
water	0.584

Table 1: Average cosine distance between soft prompts trained from the same initial word.

Task	Average distance
country_capital	0.643
landmark_country	0.665
present_past	0.710
singular_plural	0.684
capitalize_first_letter	0.712
lowercase_first_letter	0.710

Table 2: Average cosine distance between soft prompts trained for the same task.

In general, these results suggest that the soft prompt embeddings tend to be closer to the embeddings of their initialization word than to the other soft prompt embeddings trained for the same task. This confirms the hypothesis that the location of soft prompts in the embedding space is poorly interpretable. However, the result also suggest that the distance between soft prompt vectors learned for the same task, but different initialization words 3.1 is still significantly smaller than both the average distance between random words in a vocabulary and the average distance between all the learned soft prompts (which is 0.741).

Moreover, we can calculate the Pearson correlation between distances between prompt token embeddings trained with different initialization to prove that these sequences of distances are highly correlated. We take all the soft prompt vectors learned with the same initialization word, but for different tasks and at different token positions, and calculate the distances between each pair of them, thus creating a sequence of distances for each initialization word. Then we calculate the Pearson correlation between each pair of sequences, and the minimum value of correlation is 0.742 (pvalue < 0.01 for all statistics). Thus, the distances between prompt embeddings learned for different tasks are highly correlated, which suggests that the information about the task is very likely to influence the geometry of soft prompts.

3.2 Locating soft prompt vectors in the embedding space: closest tokens

3.2.1 Analysis of ϵ-neighbourhoods

In order to better understand the geometry of soft prompts, we attempt to locate them in the embedding space. First, we find the embedding vectors that are close to the learned prompts in order to interpret the embeddings through their neighbours. The neighbours and the relative directions of the embeddings of tokens from the vocabulary can be highly interpretable, thus suggesting that this method might give interesting results. We find the neighbours not only for the learned prompt vectors, but also for a random sample of 100 embeddings from the model's vocabulary.

We select a value of ϵ - the maximum distance in terms of cosine distance that the embedding vector can be from the selected vector, for it to be considered a part of the ϵ-neighbourhood of the selected vector. We set the ϵ to 0.7 based on observation that this ϵ gives an interpretable neighbourhood for tokens in the vocabulary. For example the 0.7-neighbourhood for token *_interviewer* consists of tokens *_interview, _reporters, _reporter, _interviews, _journalist, _interviewed, _analyst, _interrog, _Interview, _investigator, _columnist, _psychologist, _caller, _filmmaker, _questionnaire, _commentator, _psychiatrist, _interviewing, _moderator, _narrator, _informant, _examiner, _reviewer, _presenter, _interviewer, Interview, _commenter, _announcer*.

With the selected ϵ almost all tokens from the sample we took from vocabulary, except for the tokens we used for initialization of prompt tokens and two other tokens (*isSpecial* and *cats*), have no soft prompts in their ϵ-neighbourhood. On the other hand, the words used for initialization have almost all of the soft prompt tokens initialized from this word in their ϵ-neighbourhood.

On average, ϵ-neighbourhoods of 100 random tokens from the vocabulary contain 38.62 embeddings, of which on average 36.66 embeddings are of other words in a vocabulary, and not soft prompts. This, the embeddings of soft prompts are located outside of the closest neighbourhood of 100 randomly chosen tokens from the vocabulary. On the other hand, the ϵ-neighbourhood of soft prompts contains on average 86.49 embeddings, among which on average 15.68 are embeddings of tokens from the model's vocabulary and 70.80 are embeddings of other soft prompts. Specifically, on average the ϵ-neighbourhood of a chosen prompt token contains 20.7 embeddings of the soft prompts that were trained with the same initialization word as the chosen prompt token, 17.99 embeddings of prompts that were trained for the same task as the chosen prompt, and 31.37 embeddings of soft prompts tokens taken from the same token position as the chosen prompt token. This suggests that soft prompt embeddings are located in more dense areas of the embedding space, very close to other soft prompt embeddings. Their ϵ-neighbourhood also often contains soft prompts initialized with the same word, trained for the same task or taken at the same token position.

We counted the number of times when prompt token embeddings trained for each task appear in the neighbourhood of the prompt token embeddings trained for any other task. As can be seen from the table 3.2.1, for each of the tasks, the tokens trained for this task have more tokens trained for the same task in the neighbourhood than tokens trained for any other task. Moreover, the hypothesis that the neighbourhood of a prompt token trained for a specific task contains more prompt tokens trained for the task that is close by meaning to that specific task than tokens trained for other tasks holds for *country_capital* and *landmark_country*. Unfortunately, it does not hold for other tasks, but the reason for that might be that the model's internal notion of similarity between tasks is different from that of humans.

3.2.2 Qualitative analysis

The words from the vocabulary that are contained in the ϵ-neighbourhood of any of the soft prompt tokens fall into four categories: 1) modifications and inflections of the words used for initialization (*water, waters, Water, table, TABLE*, etc); 2) individual letters from

	cc	lc	pres_past	sing_plur	capitalize	lowercase
cc	242	92	49	68	30	19
lc	92	158	14	24	13	10
pres_past	49	14	96	74	22	24
sing_plur	68	24	74	148	37	30
capitalize	30	13	22	37	112	24
lowercase	19	10	24	30	24	84

Table 3: The number of times soft prompts trained for the tasks in each column are present in the 0.7-neighbourhood of the soft prompts trained for the tasks in each row. The abbreviations in the table stand for: cc - *country_capital*, lc - *landmark_country*, pres_past - *present_past*, sing_plur - *singular_plural*, lowercase - *lowercase_first_letter*, capitalize - *capitalize_first_letter*

uncommon alphabets (*ß, Đ, đ, Ē, ĕ, Ę*, etc); 3) unreadable groups of symbols (*?????-?????-*, *ÃĥÃĤÃĥÃĤ*, etc); 4) Other (*StreamerBot, _teasp, embedreportprint, _newcom*, etc). The words that fall into the last category are not easily interpretable: for example, the *StreamerBot* token was close to the prompt trained for landmark_country task. Tokens from all of these categories are poorly interpretable and barely related to the tasks the tokens were trained for.

3.2.3 Distance to the closest token

In terms of cosine distance to the closest token, on average tokens from the sample of the vocabulary and words from the vocabulary that we used for initialization are located at distance 0.424 to the closest embedding to them, and at distance 0.428 to the closest embedding of the word from the vocabulary. As for the trained soft prompts, they are generally located at distance 0.358 to the closest embedding to them, and at distance 0.53 to the closest embedding from the vocabulary (13 of the 240 prompt token embeddings do not have any embeddings from the vocabulary in their 0.7-neighbourhood, the average is taken without taking those tokens into account). Thus, soft prompts are located closer to each other, than words from the vocabulary, but further from the words from the vocabulary.

4 Conclusion

In this work we explored the geometry of soft prompt embeddings trained with prompt tuning for different tasks and different initialization. The results show that the location of the prompts in the embedding space depends highly on the initialization. Also, the trained prompts are located further from tokens from the vocabulary than they are from each other.

However, we also found that the soft prompts trained for the same or in some cases even closely related task are more often located closer to each other than the soft prompts trained for different tasks. Moreover, if we compute the correlations between sequences of distances between prompt tokens trained with the same initialization, these sequences turn out to be highly correlated, suggesting that there is a significant impact of the task the prompt is trained for on the location of the token in the embedding space. These findings suggest that soft prompts might actually be more interpretable than we think, but in terms other than closeness to the embeddings of words from the vocabulary.

References

1. Luke Bailey et al. "Soft prompting might be a bug, not a feature". In: ().
2. Neil Houlsby et al. "Parameter-efficient transfer learning for NLP". In: *International conference on machine learning*. PMLR. 2019, pp. 2790–2799.

3. Daniel Khashabi et al. “Prompt Waywardness: The Curious Case of Discretized Interpretation of Continuous Prompts”. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2022, pp. 3631–3643.
4. Brian Lester, Rami Al-Rfou, and Noah Constant. “The Power of Scale for Parameter-Efficient Prompt Tuning”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 3045–3059.
5. Xiang Lisa Li and Percy Liang. “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)*. 2021, pp. 4582–4597.
6. Alec Radford et al. “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8 (2019), p. 9.
7. Eric Todd et al. “Function Vectors in Large Language Models”. In: *International Conference on Learning Representations*. ICLR. 2024.

A Appendix

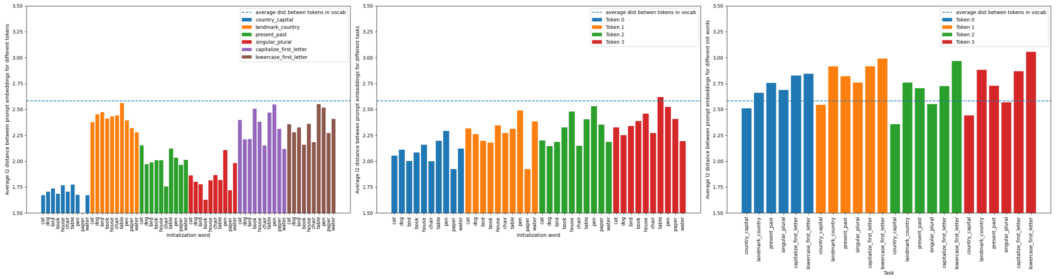

Figure 5: Average l2 distance between prompt embeddings in different groups. In the left graph the distance is measured and averaged between embeddings of soft prompts in different token positions, in the middle between embeddings of soft prompts trained for different tasks and in the right between embeddings of soft prompts trained with different initial words. The blue line indicates average distance between a sample of 100 random tokens from the vocabulary.