Soft prompting might be a bug, not a feature

Luke Bailey [*1] Gustaf Ahdritz [*1] Anat Kleiman [*1] Siddharth Swaroop [1] Finale Doshi-Velez [1] Weiwei Pan [1]

Abstract

Prompt tuning, or "soft prompting," replaces text prompts to generative models with learned embeddings (i.e. vectors) and is used as an alternative to parameter-efficient fine-tuning. Prior work suggests analyzing soft prompts by interpreting them as natural language prompts. However, we find that soft prompts occupy regions in the embedding space that are distinct from those containing natural language, meaning that direct comparisons may be misleading. We argue that because soft prompts are currently uninterpretable, they could potentially be a source of vulnerability of LLMs to malicious manipulations during deployment.

1. Introduction

Prompt tuning is a method for automatically generating effective prompts for improving the few-shot performance of large language models (LLMs) and other text-prompted generative models on a range of downstream tasks (Li & Liang, 2021; Lester et al., 2021; 2022). A small number of parameters that correspond to a fixed number of latent "tokens" is prepended to the input embedding of a frozen model. These parameters are then trained on task-specific data. The resulting "soft" prompts typically achieve better performance on the corresponding task than the best hand-engineered prompts and perform similarly to other more computationally expensive fine-tuning techniques. For these reasons, and because soft prompts are more portable than fine-tuned weights, they are growing in popularity in many areas. They have shown to be effective for common NLP tasks such as those present in the GLUE benchmark (Wang et al., 2018; Lester et al., 2021) and other domains ranging from genomic sequence modeling to image feature extraction and generation (Gal et al., 2022; Nguyen et al., 2023; Li et al., 2023a; Su et al., 2023).

Despite their effectiveness, little is understood about how soft prompts improve model performance. In Lester et al. (2021), it is suggested that soft prompt tokens may correspond to natural language tokens in the model's embedding space and may thus function similarly to hand-engineered prompts. Were this to be true, it should be possible to "decode" soft prompts back into natural sequences with similar properties, providing 1) a method for human decision-makers to audit soft prompts and 2) insights for prompt design in settings where soft prompting is not feasible (e.g. when only black-box model access is available).

Unfortunately, in this work we provide evidence that soft prompts *do not* correspond to natural language prompts in general. Our work shows that interpretation of soft prompts via naive mappings to natural language prompts can be highly misleading, as the resulting natural language prompts do not capture any of the effectiveness of the original soft prompts. Our findings can be summarized as follows:

(1) After decoding soft prompts to natural language prompts (Lester et al., 2021), we find that these decodings *do not* preserve the effectiveness of the original soft prompts.

(2) We find that the geometry of soft prompt embeddings is significantly different from natural language prompt embeddings, in terms of the magnitude and relative direction of the embedding vectors.

(3) We show that soft prompt embeddings are fairly robust to perturbations of magnitude, but are sensitive to changes in direction. This helps explain why mapping soft prompts to nearest natural language prompts results in a loss of performance. In contrast to this, we find natural tokens in the same examples are much more robust to these perturbations.

It is surprising that model inputs from parts of the embedding space that are so dissimilar to any natural tokens (and thus dissimiliar to the tokens the model was trained on) are able to elicit changes in model behaviour akin to full fine-tuning. We warn that, as prompt tuning becomes more widespread across domains, this difficult-to-interpret model pathology could potentially be exploited by malicious actors to create attacks that are subtler and more powerful than traditional adversarial examples composed of natural tokens alone.

[*]Equal contribution [1]Harvard University, Cambridge, MA. Correspondence to: Luke Bailey <lukebailey@college.harvard.edu>.

Workshop on Challenges in Deployable Generative AI at International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA. 2023.

2. Related work

Inspired by popular methods for specializing LLMs such as "zero-" and "few-shot" manual prompting (Wei et al., 2022; Lampinen et al., 2022), continuous prompt tuning was first introduced as an efficient alternative to "lightweight" fine-tuning (Houlsby et al., 2019). The method involves directly training blocks of task-specific parameters prepended to a frozen model's input after the embedding layer exactly as if they were natural token embeddings. In early work, a small number of additional parameters were similarly prepended to embeddings in each transformer layer (Li & Liang, 2021). Later, these were shown to be unnecessary (Lester et al., 2021; 2022). Prompt tuning can be very effective, achieving comparable performance to fine-tuning methods that train up to five orders of magnitude more parameters. Lester et al. (2021) briefly explore the interpretability of soft prompts by mapping soft tokens to the nearest natural tokens in the model's embedding space. They find that these nearest neighbors lie in clusters of task-relevant words and hypothesize that soft prompts may be priming the model to interpret the inputs in a task-specific domain, suggesting that soft prompts may be learning "word-like" representations. In this work, we provide evidence against this hypothesis.

Optimizing task-specific prompts directly in discrete token space is an active parallel area of research. Jiang et al. (2020) augments human-written prompts with heuristics and filters candidates based on downstream metrics. AutoPrompt does so with guidance from model gradients (Shin et al., 2020). More recently, Deng et al. (2022) uses reinforcement learning to directly optimize discrete prompts with only black-box access to the LLM being queried. The resulting prompts, while not necessarily interpretable, are still limited to the model's vocabulary and can not exhibit the unusual properties of soft prompts we highlight here.

There is existing work on improving and explaining the effectiveness of handcrafted prompting techniques like in-context learning (Brown et al., 2020) and chain-of-thought reasoning (Wei et al., 2022; 2023; Madaan & Yazdanbakhsh, 2022; Wang et al., 2022a). We find that soft prompts and natural language prompts lie in distinct regions in the embedding space, suggesting that soft prompting differs mechanically from existing natural-language prompting techniques and thus that these papers do not directly apply.

3. An empirical comparison of soft and natural-language prompts

We study soft prompts as described in Lester et al. (2021). Specifically, given a training set for a task and a frozen language model, we train a "prompt" of k new token embeddings prepended to the embeddings of each input, where k is a tunable hyperparameter. Thus, for a transformer model f with embedding dimension d and vocabulary V, a soft prompts s of k tokens is a $k \times d$ embedding matrix.

3.1. Natural-language decoding of soft prompts

We use the model embedding layer to create a bank of embeddings of all tokens in the model vocabulary (natural tokens). We then decode soft prompts to their nearest neighbor from this embedding bank. We consider two different nearest-neighbor-based decoding strategies: (1) decoding each soft token in s to its nearest natural-token neighbor by cosine similarity (NN_{CS}) and (2) doing the same using L2 distance (NN_{L2}). To evaluate the effectiveness of decoded prompts, after decoding a soft prompt s to a list of discrete tokens t in the model's vocabulary, we calculate the model performance with t prepended to every example in the testing set (replacing s). If prompts decoded in this fashion perform better than the baseline model with no task-specific prompting, then we can say that t "explains" at least some aspects of the effectiveness of s.

Formally, we denote the unprompted model by f, the model with trained soft prompts s by f_s, and the same model prompted with t by f_t. Given a test set $\mathcal{D}_{test}$ and some performance metric $\mathcal{L}$, we compare $\mathcal{L}(f_t, \mathcal{D}_{test})$ and $\mathcal{L}(f_s, \mathcal{D}_{test})$.

3.2. Geometric analysis of soft prompts

We also compare soft prompts and natural language prompts directly in the model's embedding space.

First, we compute simple properties of natural tokens and soft prompts, comparing magnitudes and proximities to natural-token nearest neighbors.

Next, we study the sensitivity of soft prompts to perturbations in magnitude. We consider transformations that reduce the L2 norm of all "token" embeddings in the soft prompt s by some factor k. Note that we only study *reductions* in magnitude because we find that the magnitudes of soft prompts already far exceed those of natural token embeddings. We then calculate the effect of the same perturbations on natural language tokens input to the prompt-tuned model.

We also consider gradual rotations of soft token embeddings in the directions of their discrete nearest neighbors in the model's vocabulary. Formally, for the ith soft token in s, s_i, and its nearest natural token embedding by cosine similarity e_i, we rotate s_i towards e_i by some angle θ. This rotation is done in the plane spanned by e_i and s_i. In the case where θ would lead to rotating some s_i beyond e_i, we simply rotate s_i to e_i and stop. Note that this case corresponds to simply replacing the soft token with its nearest natural token neighbor, as in the previous section. Again, we separately repeat these experiments on the natural language tokens in each example inputted to the prompt-tuned model.

4. Experimental results

We use soft prompts of length 100 trained in the encoder embedding space of the `T5-lm-adapt-base` model, a T5 checkpoint fine-tuned for language modeling and open-sourced by Lester et al. (2021). Prompts were trained on the SST-2 (sentiment analysis), RTE (paraphrasing), and MRPC (natural language inference) binary classification tasks (Wang et al., 2018), each reformulated as text-to-text tasks by replacing class labels of 0 and 1 with text labels such as "positive" and "negative." As in Raffel et al. (2020), if a model output does not correspond to one of the predefined class labels, the output is called a *category error* and is considered incorrect. Note that the unprompted base model consistently outputs category errors. Thus, one of the key features of the soft prompt is to guide the model's output to text that corresponds to class labels.

4.1. Nearest-neighbor decoding

Using the method outlined in Section 3.1, we evaluate the performance of discretized prompts created using NN_{CS} and NN_{L2} decoding. We find that in every case, while the model using the full soft prompt is performant (achieving 94.8%, 89.5%, and 81.9% accuracy in the SST-2, MRPC, and RTE tasks respectively), using the original model without soft prompting and with the NN_{CS} and NN_{L2} decoded prompts always lead to 0% accuracy and 100% category errors. Full results and the text decodings of the soft prompts are shown in Tables 2 and 1 of the appendix respectively. As noted by Lester et al. (2021), we see that soft prompts may lie comparatively closer to words relevant to the target tasks, evidenced by the target-labels of each tasks appearing in some of the decodings. Despite this, these decodings fail to preserve the efficacy of the original soft prompts. In fact, these neighbors often fail to produce valid responses, something trivially achieved by soft prompting and fine-tuning techniques. This suggests that the similarity between soft prompts and these nearest neighbor tokens may be superficial.

4.2. Geometry of soft prompts

To better understand why nearest-neighbor decoding of soft prompts fails, we compare soft prompt and natural language prompt embeddings in terms of magnitude and relative direction. The top panel of Figure 1 compares the distribution of the magnitude of natural token embeddings (from the `T5-lm-adapt-base` model) and soft token embeddings. We see that for all three tasks, soft tokens have magnitudes far exceeding those of natural token embeddings. The bottom panel of Figure 1 compares the distribution of cosine similarity (CS) to natural token nearest neighbors by CS for natural tokens and soft tokens. The two distributions are visibly different, as soft tokens have considerably smaller cosine similarities to their CS nearest neighbors than their natural counterparts. Together, these results indicate that the geometry of soft prompt embeddings are qualitatively distinct from that of natural prompt embeddings. They provide additional insight on why, in general, interpreting soft prompts in natural language may be challenging.

4.3. Robustness of soft prompts

Finally, we examined the robustness of soft prompts to perturbation in magnitude and direction (defined in Section 3). The results are shown in Figure 2. From the left panel, we see that, across all tasks, soft prompts are robust to reductions in magnitude. We note that reasoning about changes in magnitude in high-dimensional embedding space is difficult. To qualitatively ground our reductions in magnitude, we compare them to the L2 distances between natural token embeddings. On average across soft prompts, a reduction in magnitude of 3x reduces soft token magnitude by 700 units. We found the maximum L2 distance between a natural token embedding and its L2 nearest neighbor was 590 (see Figure 3 in the appendix), and the maximum L2 distance between any two natural token embeddings was 760. Thus, we can consistently perturb the magnitude of soft prompts by an average amount that exceeds the L2 distance between all natural tokens and their L2 nearest neighbors (and perturb by an average amount that approaches the largest distance between any two natural token embeddings) before seeing a significant performance drop. Conversely, decreasing the magnitude of each example's natural tokens, leaving the soft prompt untouched, has a much less dramatic effect, even *improving* performance on two of three tasks; on MRPC, a magnitude reduction factor of 6 gives an accuracy improvement of 8 percentage points. Full results are shown in Figure 4 in the appendix.

Figure 2 (right) shows our results for perturbations in soft prompt direction. We see a consistent drop-off in model performance when soft prompts are rotated beyond 30 degrees towards their CS nearest-neighbor. The cosine similiarity of two vectors that differ in direction by 30 degrees is $\cos(30^\circ) = 0.866$. This value is shown as a black dotted line in Figure 1. We found only 0.66% of natural tokens have a cosine similarity to their nearest neighbor of at least 0.866. Thus, we see a significant drop-off in performance even after perturbing the directions of soft prompts by a small angle: an angle that separates only 0.66% of natural tokens from their CS nearest neighbors. In contrast, these prompts are robust to a magnitude-perturbation larger than the distance between all natural tokens and their L2 nearest neighbors. From this, we conclude that soft prompts are far more robust to changes in magnitude than direction. In Figure 4 we perform the same perturbations on natural language tokens in model inputs. Unsurprisingly, we do not observe a comparable loss in accuracy.

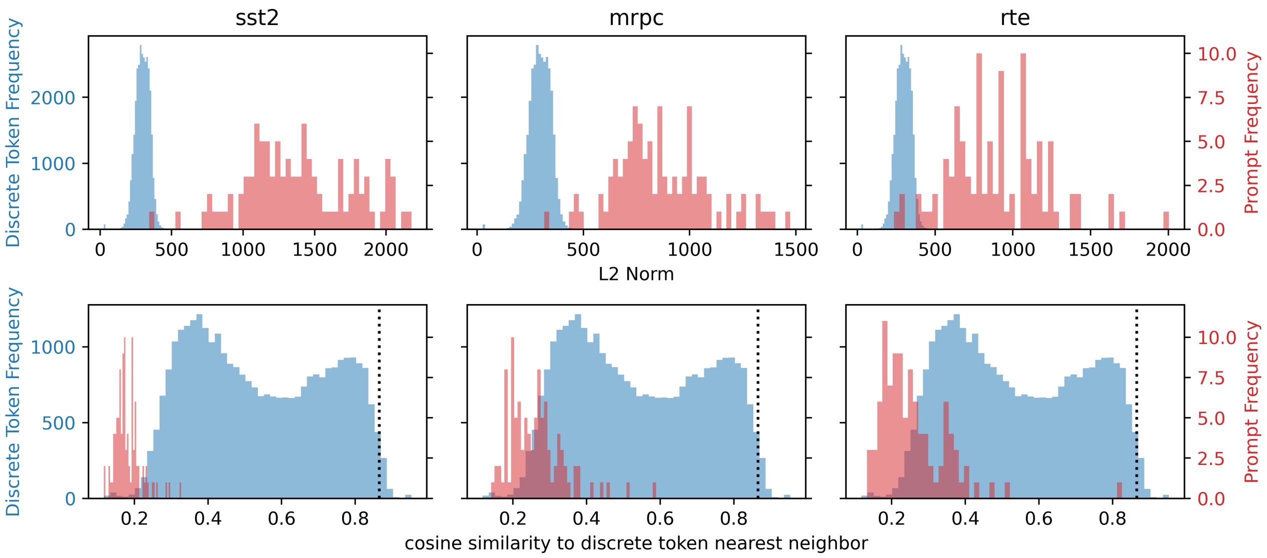

Figure 1. Soft prompts are far from discrete tokens in both magnitude and direction. Top: Histograms of the L2 norm of natural token embeddings (blue) and soft prompts (red) for `T5-lm-adapt-base`. Bottom: Histograms of the largest cosine similarity between natural tokens (blue)/soft prompt tokens (red) and any natural token embedding in the vocabulary. The vertical line indicates a cosine similarity of $0.866 = \cos(30^\circ)$. Note the y-axis scale is different for each embedding type.

We note that no soft token differs from its natural token CS nearest neighbor by less than 30 degrees. It stands to reason that NN_{CS} and NN_{L2} decoding both fail to produce performant discrete prompts; across all three tasks, soft prompts are not robust to the changes in direction effectively induced by such a decoding.

5. Discussion

Our findings raise several important questions about the interpretability and safety of soft prompts in LLMs. They also shed light on some intriguing characteristics of LLMs and provide directions for future research.

Our results point to a potentially undesirable feature of LLMs: that the behavior of these models on out-of-distribution inputs like soft prompts cannot be extrapolated from their performance on training data. The surprising, and often undesirable, generalization of complex models to OOD data has been studied in other deep models (Chakraborty et al., 2018; Kirichenko et al., 2020).

The difficulty of interpreting soft prompts also raises concerns about their usage in deployment. Specifically, it opens a potential avenue for adversarial attacks on LLMs that are difficult to detect and mitigate—the model can be maliciously manipulated to behave in undesirable ways through the clever usage of soft prompts that are difficult for humans to detect. We stress that, because soft prompts differ so dramatically from natural-language prompts, it is likely that they allow for unique attacks unlike those already possible with natural-language prompts alone (Wallace et al., 2021; Rao et al., 2023).

An obvious way to avoid soft prompt attacks would be to simply not allow users to input soft prompts into deployed models. In some scenarios, however, this may not be possible. We are seeing an increasing trend of incorporating multimodal inputs to LLMs by simply using a shared embedding space across inputs attached to a pretrained LLM (Su et al., 2023). Vision Language Models (VLMs) are a typical example, with models such as LLaVA (Liu et al., 2023), Mini GPT-4 (Gong et al., 2023), and BLIP-2 (Li et al., 2023a) simply taking images in the input, encoding them with a vision model, and projecting them to a sequence of soft tokens in the embedding space of an LLM such as FLAN-T5 (Chung et al., 2022) or LLaMA (Touvron et al., 2023). Such multimodal inputs could provide an entry point to out-of-distribution regions of the LLM's embedding space just like soft prompts, essentially allowing a user to conceal an adversarial soft prompt within an unassuming multimodal input. Rudimentary forms of this attack have already been shown by Carlini et al. (2023), who created adversarial images that, when inputted to VLMs such as Mini GPT-4 and LLaVA, caused the model to produce toxic outputs. In fact, their attacks were more effective when exploiting image inputs of multimodal models rather than attacking LLMs with natural language tokens.

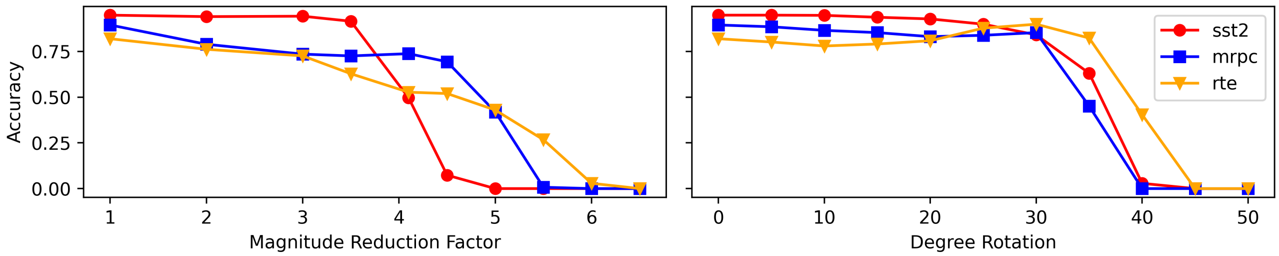

Figure 2. Left: Effect of reducing magnitude of soft prompts on model performance. Magnitude reduction is applied to all tokens in the soft prompt. Right: Effect on model performance of incrementally rotating soft prompt token vectors in the directions of their nearest neighbors by cosine similarity in the model's vocabulary.

Separately, "textual inversion," a technique analogous to prompt tuning, has become a popular tool for personalizing large text-to-image models, allowing users to train soft tokens corresponding to new concepts, objects, or styles (Gal et al., 2022). These tokens can be used in prompts just like real tokens, and unlike fully fine-tuned models, 1) can easily be combined with other soft tokens and 2) are lightweight enough to be easily distributed to other users. Indeed, there already exists at least one large community database of soft tokens trained for the Stable Diffusion, a popular text-to-image model (Rombach et al., 2022).[1] Without better tools for either understanding the effects of these soft tokens or transforming them into more interpretable counterparts, they too could potentially be used as attack vectors.

Under this light, our findings underscore the importance of more sophisticated investigations of how and why soft prompts work, in relation both to how they can be improved and also to the model pathologies they unveil. One potential approach is to utilize mechanistic interpretability techniques (Olah, 2022; Wang et al., 2022b; Nanda et al., 2023; Conmy et al., 2023) to determine what parts of the model are utilizing the soft prompt and how they contribute to the final output. Understanding the mechanisms by which models use soft prompts to specialize to target tasks can help us to understand how soft prompting may be maliciously exploited and how we may defend against such attacks. For example, we may find that soft prompts adversarially generated to cause a model to output toxic text consistently activate certain regions of the model. Removing or editing these regions may cause the model to be more robust to such attacks. Here the adversarial soft prompt is being used as a probe into the network to try and identify the circuits involved in unwanted model behavior. Previous work has used model interventions to change factual associations in LLMs (Meng et al., 2022) and reduce the toxicity of LLMs (Li et al., 2023b), but identified such interventions using different methods.

An important limitation of our work is that we only study prompts for a language model that performs relatively poorly in zero- and few-shot evaluations. In the future, we hope to repeat our experiments using larger and more capable models.

6. Conclusion

This work analyzes simple methods for interpreting soft prompts that are proposed in the literature. Our experiments reveal that soft prompts differ from natural language prompts in a number of key ways and that naive "decoding" techniques are unable to produce natural-language interpretations of soft prompts that can explain any of the latter's effectiveness. Together, the evidence suggests that soft prompts operate according to different underlying mechanisms than regular prompts. Neat analogies between the two are therefore misleading, or worse; the unique properties of soft prompts could potentially expose models to new types of malicious and difficult-to-detect manipulations in deployment. In short, we caution that soft prompts may be bugs rather than features of LLMs.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1750358 and Grant No. IIS-2107391. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. G.A. is supported by a Simons Investigator Fellowship, NSF grant DMS-2134157, DARPA grant W911NF2010021, and DOE grant DE-SC0022199. L.B. is supported by a grant from the Harvard College Office of Undergraduate Research and Fellowships.

[1] The "Stable Diffusion concepts library" can be accessed here.

References

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Carlini, N., Nasr, M., Choquette-Choo, C. A., Jagielski, M., Gao, I., Awadalla, A., Koh, P. W., Ippolito, D., Lee, K., Tramer, F., et al. Are aligned neural networks adversarially aligned? *arXiv preprint arXiv:2306.15447*, 2023.

Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., and Mukhopadhyay, D. Adversarial attacks and defences: A survey, 2018. URL https://arxiv.org/abs/1810.00069.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. *arXiv preprint arXiv:2304.14997*, 2023.

Deng, M., Wang, J., Hsieh, C.-P., Wang, Y., Guo, H., Shu, T., Song, M., Xing, E. P., and Hu, Z. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*, 2022.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv*, abs/2208.01618, 2022. URL http://arxiv.org/abs/2208.01618.

Gong, T., Lyu, C., Zhang, S., Wang, Y., Zheng, M., Zhao, Q., Liu, K., Zhang, W., Luo, P., and Chen, K. Multimodal-gpt: A vision and language model for dialogue with humans. *arXiv preprint arXiv:2305.04790*, 2023.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp, 2019.

Jiang, Z., Xu, F. F., Araki, J., and Neubig, G. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

Kirichenko, P., Izmailov, P., and Wilson, A. G. Why normalizing flows fail to detect out-of-distribution data. *Advances in neural information processing systems*, 33: 20578–20589, 2020.

Lampinen, A. K., Dasgupta, I., Chan, S. C., Matthewson, K., Tessler, M. H., Creswell, A., McClelland, J. L., Wang, J. X., and Hill, F. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*, 2022.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning, 2021. URL https://arxiv.org/abs/2104.08691.

Lester, B., Yurtsever, J., Shakeri, S., and Constant, N. Reducing Retraining by Recycling Parameter-Efficient Prompts. *arXiv preprint arXiv:2208.05577*, 2022. URL https://arxiv.org/abs/2208.05577.

Li, J., Li, D., Savarese, S., and Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023a.

Li, M., Davies, X., and Nadeau, M. Circuit breaking: Removing model behaviors with targeted ablation. 2023b.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation, 2021. URL https://arxiv.org/abs/2101.00190.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

Madaan, A. and Yazdanbakhsh, A. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*, 2022.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

Nanda, N., Chan, L., Liberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

Nguyen, E., Poli, M., Faizi, M., Thomas, A., Birch-Sykes, C., Wornow, M., Patel, A., Rabideau, C., Massaroli, S., Bengio, Y., et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *arXiv preprint arXiv:2306.15794*, 2023.

Olah, C. Mechanistic interpretability, variables, and the importance of interpretable bases. *Transformer Circuits Thread(June 27). http://www. transformer-circuits. pub/2022/mech-interp-essay/index. html*, 2022.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020. URL http://arxiv.org/abs/1910.10683.

Rao, A., Vashistha, S., Naik, A., Aditya, S., and Choudhury, M. Tricking llms into disobedience: Understanding, analyzing, and preventing jailbreaks, 2023. URL https://arxiv.org/abs/2305.14965.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models, 2022.

Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

Su, Y., Lan, T., Li, H., Xu, J., Wang, Y., and Cai, D. Pandagpt: One model to instruction-follow them all. *arXiv preprint arXiv:2305.16355*, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Wallace, E., Feng, S., Kandpal, N., Gardner, M., and Singh, S. Universal adversarial triggers for attacking and analyzing nlp, 2021. URL https://arxiv.org/abs/1908.07125.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Wang, B., Min, S., Deng, X., Shen, J., Wu, Y., Zettlemoyer, L., and Sun, H. Towards understanding chain-of-thought prompting: An empirical study of what matters. *arXiv preprint arXiv:2212.10001*, 2022a.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022b.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

A. Prompt Properties

We show the full results for decoding soft prompts using NN_{CS} and NN_{L2} and using resulting decoded tokens as prompts in Table 2. We see that when using no prompt and either of the decoded prompt inputs, model performance falls to 0 as the model starts exclusively outputting category errors. Narrowing the model output to the set of valid textual class labels is one of the important model behaviours that the soft prompt elicits, yet we see that the decoded prompts fail to recreate this behaviour on any of the input datapoints tested across all three tasks.

Table 1 shows the discrete decodings of the soft prompts produce by our two different nearest-neighbor decoding algorithms. We see, as reported by Lester et al. (2021), that the text target labels for each task seem to appear in some of the decodings (marked in red).

Task	NN_{CS} decoded soft prompt	NN_{L2} decoded soft prompt
SST-2	negative positive Englisch factors joc namedif Jacket the.... featured ugly⟨/s⟩ Arts Memoryall plein ProductsPCmila explaining Rwanda wasn stressfuling goddess the, filmmaker gebraucht în meeting drug Bestell positiveangoorganizarea the film mândr cough Mitchell 2013. authentication the memorable beginnt makes bizarre soybean digitalexpertise Nevada Dacia thecot intimacy Could specialist quizstellbar Duration precumly Whole story Hi") Mon..." stairs the the descoperitance the arbor acasă Em the." obiectealonsca truffleAccess the with the of. today, to the⟨extra_id_3⟩alegereaesprit definesTotodat	negative positive is factors 07 namedif Jacket the Issue featured ugly⟨/s⟩ Arts Memoryall beautifully coursesPC stăpân Record wines wasn stressfulING famous the, filmmaker fat binary meeting drug the negativeangoorganizarea the films mândr cough Mitchell 2013. authentication the memorable beginnt helps bizarre skincare digital bone Nevada Dacia thecot excellent could specialist InterviewIONAL the precumly THE storyvină") Jun..." situații the the grillance the Per acasă Em humidity." obiecte⟨extra_id_21⟩ ca cream oricum the⟨extra_id_5⟩ the⟨extra_id_12⟩. today, to the⟨extra_id_3⟩alegerea Once definesTotodat
MRPC	in equivalent 1000glasisse 1953 viele giving not_birthrun heart tous",varvalent We application zile, faitțe becoming Cl political China Pentru Blätter her einpier Ze Multimainlyations model ciudabasedaci ferm Attack oriner exactcha United 1973..."ent contsch injury 2019. videos partyarii the möglich muss circuit punukke President Editionhin prea communicationcher annual General litificationardkanis32 another Fort feed location of best accurateăriul21nic⟨/s⟩ 29 Informationen sharp Cop⟨extra_id_6⟩ban located numărul Kinder legend	in equivalent 100 createdisse 1953 viele giving not_erun heart tous",varpreședintele We application zile, faitțe becoming Cl political China Pentru Blätter her ein is Ze Multimainlyations model⟨extra_id_37⟩ basedaci catheter Attack oriner exactcha United 1973allyent contsch the 2019. videos partyarii the möglich muss circuit punukke President Editionhin prea communicationcher annual General litificationardkanis32 another Fort feed location of best accurateăriul21nic has29 Informationen sharp Cop⟨extra_id_6⟩ban located întreb more legend
RTE	tailment 300 football * 10%clomm surrounding Encette represent terminal Sugar market Winner than candidates casting theme premièreplan Foundation 9iert opened former Their Kuhalter• woman gold roll remarks not Group Spring not kick32 wurde modecuEntulente is Toyota fat Senezville Sauna soon not_rechtTrebuie goals⟨extra_id_3⟩ unsererdra) treptat expert' religious putin home investiții include 60 Year Green not not_ING....poți Primary do⟨/s⟩ainsiact celuifähigkeit pur conductedring type quarterly attempt rectangle century ignite Voi,	⟨extra_id_72⟩ ment 300 football * 10% strânsmm surrounding Encette represent terminal sugar the winner than candidates casting theme premièreplan Foundation 9iert opened former Their Ku 26,• woman gold roll-2 not Group Spring not kick32 wurde modecu Intulente is Ceci fat Senezville Sauna soon not_recht is goals⟨extra_id_3⟩ unsererdra) treptat expert' religious putin home investiții include 60 Year Green not not_ING,poți, do⟨/s⟩ainsiact celui stăpân pur conductedring type $9 attempt rectangle century renovateda,

Table 1. Result of decoding soft prompts nearest neighbors by cosine similarity (NN_{CS}) and L2 distance (NN_{L2}). We highlight in red appearances of target text labels (either in full or fragments) in the decodings. The labels are "positive" and "negative" for SST-2, "equivalent" and "not_equivalent" for MRPC, and "entailment" and "not_entailment" for RTE.

Prompt Type	SST-2	MRPC	RTE
Soft Prompt	0.9484	0.8946	0.8195
No prompt	0	0	0
L2 Norm nearest-neighbors	0	0	0
Cosine Similarity nearest-neighbors	0	0	0

Table 2. Accuracy of `T5-lm-adapt-base` model on testing set using different prompting schemes

On top of the various properties of soft prompts presented in the main text, we also investigated the distribution of L2 distances to natural token L2 nearest neighbors for soft prompts and natural tokens. The results are shown in Figure 3. Here we see that just like for CS nearest neighbors (shown in Figure 1), soft prompts have far larger L2 distances to nearest neighbors than natural tokens. Soft prompts are located in unique neighborhoods of the embedding space.

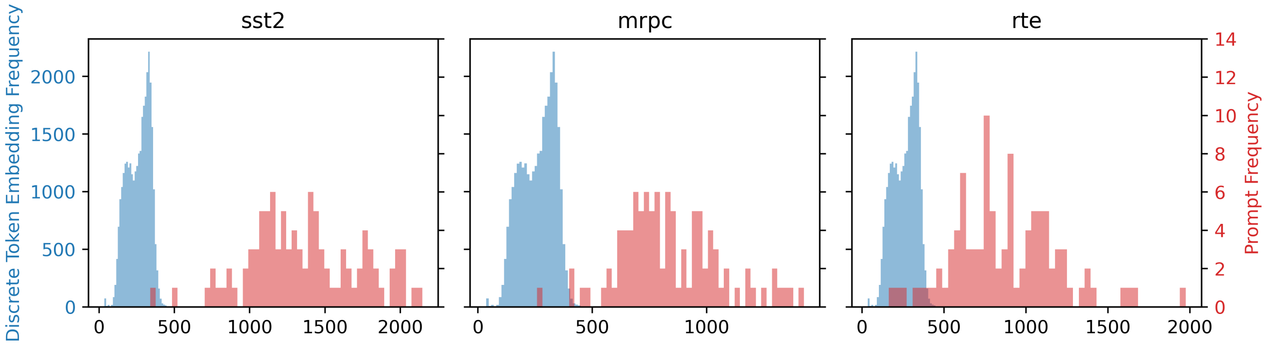

Figure 3. Histograms showing smallest L2 distance of discrete token embeddings (blue) and soft prompts (red) to discrete token embeddings. For discrete tokens, the largest difference between nearest neighbors is 756.83.

In Figure 4 (left), we provide the results of perturbing the magnitude of the embeddings of natural tokens in the prompt-tuned model without perturbing the soft prompt tokens. Compared to the soft-prompt perturbations in Figure 2, the effects of even extreme reductions in magnitude are either modestly negative or, surprisingly, even *positive*. We speculate that natural tokens somehow interfere with the inference-time action or optimization of the soft prompt. Accordingly, reducing them in magnitude may allow the soft prompt to have an even greater effect, more than compensating for the noise effectively being added to model embeddings. Clearly, the soft prompt has a powerful ability to distort the inner workings of the model, sometimes permitting downright unintuitive behavior. Rotating natural tokens towards their nearest neighbors in the vocabulary by cosine similarity (excluding the tokens themselves) has a slighter footprint on the prompt-tuned model's accuracy; results are shown in the right panel of Figure 4. Together, Figure 2 and Figure 4 demonstrate that soft prompts are quite brittle, apparently functioning differently from natural language tokens in model inputs.

B. Prompt Robustness

Figure 2 shows the effect of changing soft prompt magnitude and direction on task accuracy. Figure 5 also shows, at each magnitude reduction factor and degree rotation, the proportion of inputs that resulted in category errors and the non-category-error accuracy (that is the accuracy of the model when only considering valid outputs that correspond to textual class labels). Interestingly, in almost all of the plots, we see that the overall accuracy reduction before the drop-off comes from reductions in non-category-error accuracy, and the large breakdowns in accuracy are caused by increased category errors. Additionally, during the drop-off in model accuracy caused by an increasing number of category errors, the non-category-error accuracy often increases. This may be being caused by the model simply outputting category errors on inputs on which it is more uncertain, outputting class labels on inputs it finds easier to classify, or there is some more complex underlying model behavior at work.

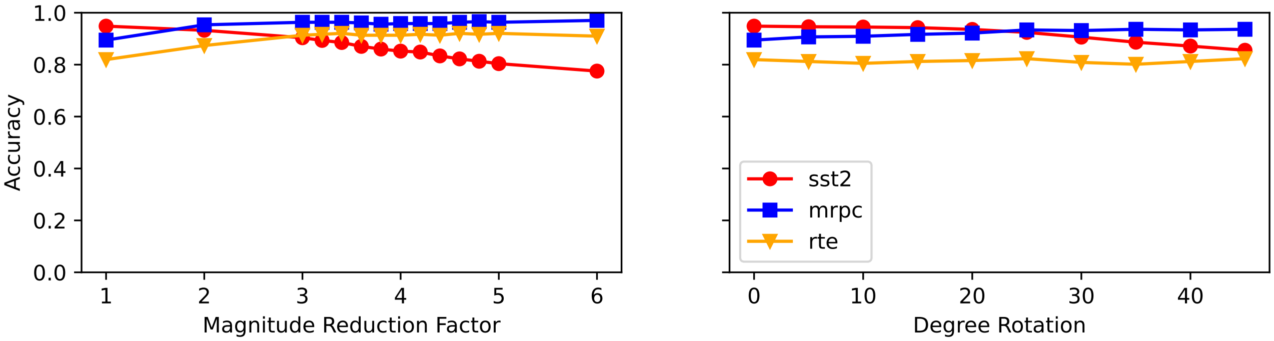

Figure 4. Left: Perturbing the magnitude of natural tokens in inputs to the prompt-tuned model has a much less destructive effect than perturbing the soft prompt in the same way. On two tasks, performance even improves. Concretely, compared to the baseline, a magnitude reduction factor of 6 raises MRPC accuracy by 7.6% and RTE accuracy by 9.0%, down from a peak of 11%. RTE accuracy declines by 17%. Right: Rotating natural tokens towards their nearest neighbors in the model's vocabulary by cosine similarity has comparatively modest effects on task accuracy.

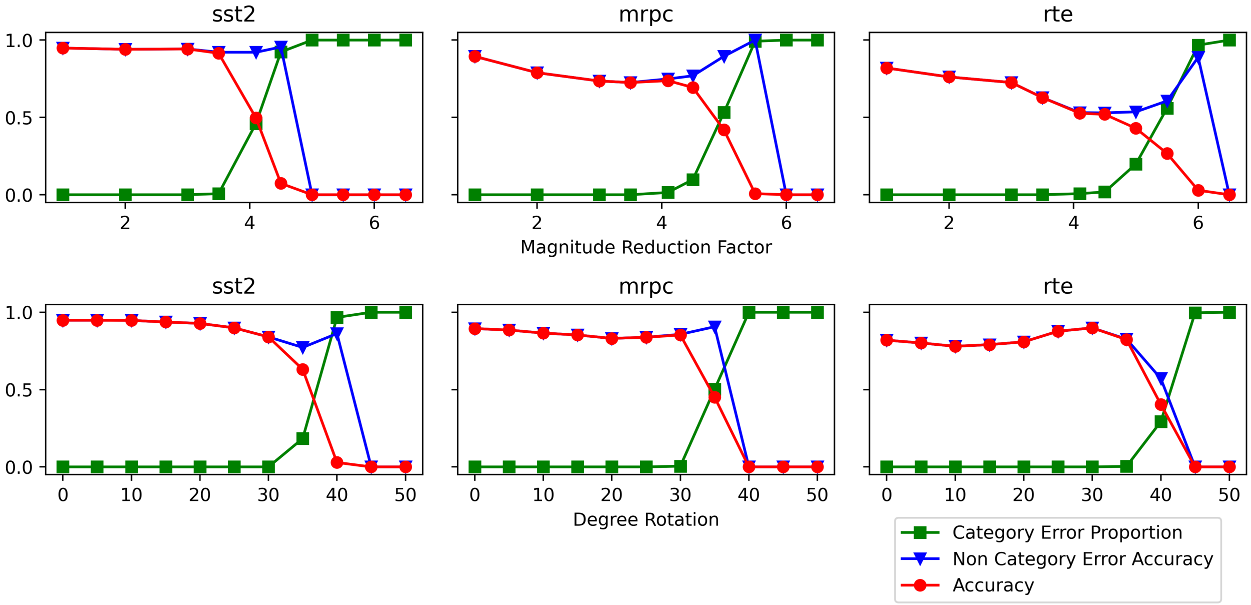

Figure 5. Effect of reducing soft prompt magnitude on model performance. Non-category-error accuracy shows the accuracy of the model when only considering valid outputs that correspond to textual class labels.