August 2023

AUTHOR-SPECIFIC PREFIX-TUNING FOR PERSONALIZATION OF LARGE LANGUAGE MODELS

Elizabeth A Hutton

AUTHOR-SPECIFIC PREFIX-TUNING FOR PERSONALIZATION OF LARGE LANGUAGE MODELS

AUTHOR:
Elizabeth A Hutton

ABSTRACT

This article outlines a novel approach to the personalization of large language model (LLM) outputs to an individual writer in an artificial intelligence (AI)-assisted writing application without the need for fine-tuning or prompt-engineering. With this approach, an individual's writing style may be encoded through a compact, learned model which maps writing samples to an "author-embedding" which may be prepended to the input of an LLM (in the manner of prefix-tuning) to steer the model to generate content in the writing style of that individual. The presented techniques involve several processing steps, including the selection of an optimal subset of an author's writing samples, the training of an author embedding model, and the use of author-embeddings as a prefix to an LLM.

INTRODUCTION

Large language models (LLMs) have become increasingly popular in artificial intelligence (AI)-assisted writing applications given their ability to generate human-like text. For a user who seeks AI-written material (for example, emails, presentation scripts, branded copy, etc.) that resembles their unique writing style, there are a limited number of options. Existing state of the art language model personalization techniques involve either fine-tuning[1] or in-context learning[2], which are both impractical for the AI-assisted writing application for reasons explained below.

The first option is to train fine-tuned models, which can achieve high levels of writing personalization. Fine-tuning is a process wherein a pre-trained model is further trained on a dataset of writing samples from an individual user. The training requires substantial amounts of user data which may be difficult to collect and store, or which may not be available at all as would be the case for new users. Furthermore, the fine-tuning approach requires that a model be trained and maintained for every user, which does not scale well as the number of users grows. For these reasons, it is impractical to host fine-tuned, per-user models for the personalization of AI-assisted writing.

The second option involves using pre-trained LLMs in a few-shot learning setting (a.k.a in-context learning) where examples of a user's writing style are injected into a prompt. It is possible to steer a model's generations towards certain writing styles using carefully crafted prompts without modifying the underlying model, however there are significant drawbacks to such an approach. Even in the simplest zero-shot setting where the prompt does not include examples of the desired behavior, prompt engineering is a challenging process of trial and error. In the AI-assisted writing use case, a prompt might look something like "Write a response to the following email {email message} in the style of the user. Examples: {user writing samples}." Notably, prompt engineering iterates on the prompt template itself and not the text that is injected. Not only is prompting with multiple examples computationally inefficient (response time scales quadratically with the length of an input and repeated calls using the same prompt wastes resources), but it will also not reliably produce longform, personalized writing. Even if it were possible to select a few optimal examples of an individual's writing style for inclusion in a prompt, this would make each prompt prohibitively long.

To tackle the inefficiencies of prompt-based, in-context learning, researchers at Stanford University have proposed "prefix-tuning"[3] wherein a compact task-specific representation, rather than a natural language prompt, is prepended to a model input. The prefix model is a learned mapping of a task to a continuous vector and comprises only a fraction of the trainable parameters of the core LLM (which remains unchanged). This approach has not yet been applied to the task of generating text in an individual's writing style, which requires a different process for training the prefix model. Existing approaches are only applicable when there is a one-to-one mapping (of, for example, a task to a prefix), but numerous writing samples would be required to effectively capture an individual's writing style, requiring the model to learn a many-to-one mapping.

DETAILED DESCRIPTION

To address the above-described deficiencies, the techniques presented here leverage author-specific continuous vector prefixes to generate text in a specific writing style using LLMs. This approach offers a number of advantages over the existing approaches that were described above. Among other things, the presented techniques do not require the training and storage of fine-tuned models, nor do they require prompt engineering. Instead, the approach involves selecting a representative subset of an individual’s writing to efficiently encode style information, training a prefix model on the selected writing samples to generate author-specific vectors, and employing those author-specific vectors as an input prefix to an LLM to generate text in an individual’s writing style.

This approach includes several processing steps which must be completed before training the author embedding model. The first step involves the selection of an optimal subset of writing samples. When a user has a large corpus of writing samples available, a representative subset of those samples may be selected for use with the prefix model. To select an optimal subset, a number of different steps may be completed.

The first step comprises preprocessing. During this step, a user's writing samples may be preprocessed to facilitate any subsequent analysis. Such a preprocessing activity may include tasks such as tokenization, lowercasing, removal of stop words, and stemming or lemmatization.

The second step is feature extraction. During this step, various features that represent a user's writing style may be extracted from the preprocessed text. The developed feature vectors may be evaluated to ensure that they effectively capture individual differences in writing style, rather than content. Such features may include lexical features (e.g., vocabulary richness, word frequency distributions, n-gram patterns, and common phrases), syntactic features (e.g., sentence length, sentence structure, and part-of-speech patterns), and stylistic features (e.g., tone, sentiment, use of figurative language, and domain-specific terminology).

The third step involves clustering the samples. The developed feature vectors from the previous step may be used to cluster the writing samples of a user into groups with similar characteristics. This can be achieved using one of several unsupervised machine learning clustering algorithms where the optimal number of clusters may be determined based on a clustering evaluation metric and the elbow method.

The fourth step is subset selection. From each of the above-described clusters, one or more representative samples may be chosen (based on their proximity to the cluster centroid or other criteria that signify their representativeness). The selected samples form a representative subset which capture the diversity and unique characteristics of the user's writing style. By continually refining the subset selection based on the similarity between the generated text and the user's writing style, this algorithm can further improve the personalization of the LLM-generated text. While the selection of the writing samples that are to be injected is an iterative process, this is not considered the same as prompt engineering, which primarily operates on the prompt template itself. Unlike prompt engineering, the prefix-tuning approach does not require careful prompt selection and testing because the underlying prefix model will learn a correct mapping from the prompt of one’s choosing to the desired model output.

Once an optimal subset of writing samples has been selected, the next step is to train the author prefix model. This activity involves the prefix-tuning strategy described above, where instead of learning to encode task information, the model learns to encode writing styles. Figure 1, below, displays the relevant components for modeling the interaction between the author prefix model, the input, the LLM, and the resulting writing outputs.

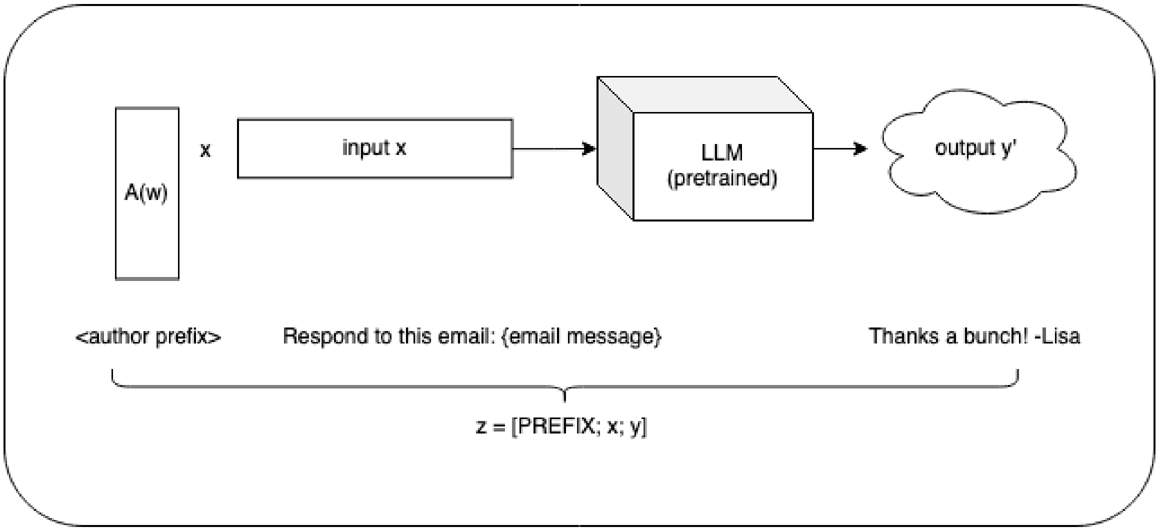

Figure 1: End-to-End Pipeline for Training Prefix Model

As shown in Figure 1, above, the training process comprises several elements. Element *A(w),* the prefix model, may take the form of a small neural network which models the relationship between input writing samples, *w*, and an author-embedding or *PREFIX*. The element *x* may be any task-specific content (such as, for example, an email thread) that is required for prompting an LLM. The LLM may be fed a combination of *PREFIX* with *x* and the desired output *y* to generate *y'*. Rather than updating any of the weights in the LLM, any error between *y* and *y'* may be backpropagated via gradient descent to update the weights of the prefix model, *A(w)*. Each user of such an AI-assisted writing application will have a unique prefix, which may be updated periodically as new writing samples become available or as the prefix model is retrained. The prefix model is also capable of generating meaningful representations even for unseen users (i.e., users that are not in the training set). Compared to any LLM, or even to fine-tuned adaptor models, such a model is significantly more compact and easier to store.

Once the author embedding model has been trained, it may be used as a prefix to generate unique writing with an LLM. When a user requests a new generation task, their unique prefix may be prepended to the input context and then fed into the LLM to generate a response that is in their writing style.

In summary, the novel techniques presented herein enable the efficient personalization of LLM outputs to an individual writer in an AI-assisted writing application without the need for fine-tuning or prompt-engineering. With such an approach, an individual's writing style may be encoded through a compact, learned model which maps writing samples to an "author-embedding" which may be prepended to the input of an LLM (in the manner of prefix-tuning) to steer the model to generate content in the writing style of that individual. The presented techniques include several processing steps, including selecting an optimal subset of an author's writing samples, training an author prefix model, and using author-embeddings as a prefix to an LLM.

[1] Syed, B., Verma, G., Srinivasan, B. V., Natarajan, A., & Varma, V. (2020). "Adapting Language Models for Non-Parallel Author-Stylized Rewriting." Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), 9008-9015. https://doi.org/10.1609/aaai.v34i05.6433

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, et al. "Language Models are Few-Shot Learners," in Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, pp. 1877-1901, 2020.

[3] X. L. Li and P. 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), Online, Aug. 2021, pp. 4582-4597. Association for Computational Linguistics. [Online]. Available: https://aclanthology.org/2021.acl-long.353. DOI: 10.18653/v1/2021.acl-long.353.