Style Vectors for Steering Generative Large Language Models

Kai Konen **Sophie Jentzsch** **Diaoulé Diallo** **Peer Schütt**
Oliver Bensch **Roxanne El Baff** **Dominik Opitz** **Tobias Hecking**
Institute for Software Technology, German Aerospace Center (DLR)
{first}.{last}@dlr.de

Abstract

This research explores strategies for *steering* the output of large language models (LLMs) towards specific styles, such as sentiment, emotion, or writing style, by adding *style vectors* to the activations of hidden layers during text generation. We show that style vectors can be simply computed from recorded layer activations for input texts in a specific style in contrast to more complex training-based approaches. Through a series of experiments, we demonstrate the effectiveness of *activation engineering* using such *style vectors* to influence the style of generated text in a nuanced and parameterisable way, distinguishing it from prompt engineering. The presented research constitutes a significant step towards developing more adaptive and effective AI-empowered interactive systems.

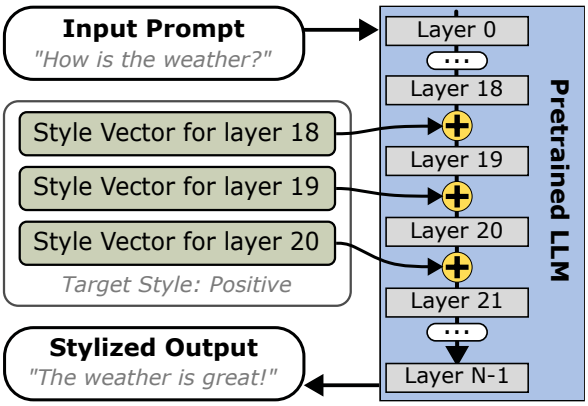

Figure 1: The LLM output is steered by adding style vectors to selected layers (e.g., layers 18-20) during a forward pass. For example, the answer of the LLM to the input prompt "How is the weather?" is steered towards a **positive** style, with a sample answer of "The weather is great!", a positive answer.

1 Introduction

Large language models (LLMs) pre-trained on vast corpora have marked a significant milestone in natural language processing, presenting remarkable language understanding and generation capabilities. Models like GPT-2 (Radford et al., 2019) and more recent variants such as GPT-3 (Brown et al., 2020) and GPT-4 (OpenAI, 2023) have become influential in transforming the landscape of text generation. LLMs have the potential to encode extensive public knowledge and can respond to a wide array of text prompts in a manner that often closely resembles human communication. OpenAI's ChatGPT, in particular, has garnered substantial attention, propelling discussions about generative AI from the scientific community into the broader public sphere (Brown et al., 2020; OpenAI, 2023). In this era of ever-advancing AI, it is becoming increasingly apparent that LLM-based artificial assistants will play a prominent role in both professional and personal contexts (Bender et al., 2021; Zhao et al., 2023). Examples of these are conversational information search (Alessio et al., 2023; Shah et al., 2023), human-AI co-creation (Yuan et al., 2022; Chung et al., 2022), or complex goal-oriented dialogues (Snell et al., 2022).

In these complex settings, text generation on a lexical level alone is not sufficient for effective human-AI interaction. Over and above that, a cognitive AI assistant should also be able to adapt to the human user on an affective and emotional level regarding engagement, regulation, decision-making, and discovery (Zhao et al., 2022). There is evidence that LLMs perform well on affective computing tasks, such as sentiment classification and personality prediction, and can have emotional dialogue capabilities to some extent. However, the resulting capabilities do not go far beyond simpler specialized models, presumably due to the LLMs' generality (Zhao et al., 2023; Amin et al., 2023). This limitation calls for mechanisms to better control implicit information and the style of an LLM's output.

Prompt engineering has been a promising ap-

proach in human-AI collaborative tasks, improving task efficiency and user collaboration (Wu et al., 2022). However, it is often highly task-specific and entails manually crafting prompts.

In this paper, we build upon and extend the works of Subramani et al. (2022) and Turner et al. (2023), which focus on steering the output of LLMs by modifying their internal states. In a series of experiments, using datasets of text samples labeled with sentiments and emotion categories, we show that one can derive a vector representation of a desired style class (e.g., *positive* sentiment) that, when added to the activation of certain layers of an LLM (in this work LLaMa (Touvron et al., 2023)), its output shows characteristics of this style class (see Fig. 1). Our experiments show that the effect of the changed models is more salient when prompted with subjective input (e.g.,"How do you define art?") rather than with factual input that allows little degrees of freedom (e.g., "What is the world's longest river?"). Our research aims to bridge the gap between the LLM's capabilities and the nuanced requirements of human-AI interactions, thus extending this novel dimension to the realm of controlling LLM outputs.

An open-source implementation of the algorithms used in this paper is available[1].

2 Background and Related Work

The introduction of transformer architectures in neural networks (Vaswani et al., 2017) has led to a massive leap in the development of contextualized language models, such as GPT (Brown et al., 2020). These novel large language models (LLMs) capture relations in the natural data and implicitly encode an unlimited number of more abstract concepts, such as sentiment or style. This quality has been exploited in several recent investigations and can be both a risk (Wagner and Zarrieß, 2022) and a chance (Schramowski et al., 2022).

Many approaches have been developed with the aim of controlling or affecting the output of LLMs, also referred to as *steering* LLMs (Brown et al., 2020; Zhang et al., 2022; Jin et al., 2022).

Traditionally, methods for producing text in a specific style fall under the domain of *stylized response generation* (Sun et al., 2022; Yang et al., 2020; Gao et al., 2019; Jin et al., 2020). Nonetheless, as common approaches of this class necessitate training and fine-tuning whole models, these methods are not applicable to state-of-the-art LLMs, given the immense parameter count and training costs of LLMs (Hu et al., 2021).

Another line of research worth mentioning that aims to employ alternative approaches to the traditional fine-tuning approach is the parameter-efficient transfer learning approach (Houlsby et al., 2019) using adapter modules, which seek to minimize trainable parameters. In contrast, in our work, we focus on a different efficiency aspect, not only on the minimal computational resources but also on the minimal data resources used.

A related but conceptually different approach to affect the output of LLMs is *text style transfer* (TST) (Jin et al., 2022; Reif et al., 2022). TST aims to transfer the style of a given text into a desired, different style. In contrast, steering LLMs deals with the task of generating a response in a desired style. We refer to Jin et al. (2022) for a detailed overview of TST.

Prompt engineering (Keskar et al., 2019; Radford et al., 2019; Shin et al., 2020; Brown et al., 2020; Lester et al., 2021; Li and Liang, 2021; Wei et al., 2022; Wu et al., 2022) focuses on controlling and directing the output of a language model by designing input prompts or instructions. By tailoring the natural language prompts, the model's output can be steered towards producing responses in the desired style.

Some recent approaches move in a new direction by modifying the layer activations of an LLM during the forward pass (Subramani et al., 2022; Turner et al., 2023; Hernandez et al., 2023). These approaches can be grouped under the term of *activation engineering*. Subramani et al. (2022) presented so-called steering vectors that, when added to the activations at certain layers of an LLM, steer the model to generate a desired target sentence x from an empty input. The rationale behind this is that the information needed to produce the target sentence is already encoded in the underlying neural network. Thus, the approach works without re-training or fine-tuning the model itself.

Starting with an empty prompt, i.e., beginning of sentence token <*bos*>, the vector $\mathbf{z}_{steer} \in \mathbb{R}^d$ is added to the activations of a defined layer of the model, where d is the dimension of the layer to generate the next of the T tokens of x. The objective is to find a steering vector $\hat{\mathbf{z}}_{steer}$ that maximizes the

[1]Find all resources at https://github.com/DLR-SC/style-vectors-for-steering-llms

log probability:

$$\hat{\mathbf{z}}_{steer} = \underset{\mathbf{z}_{steer}}{argmax} \sum_{t=1}^{T} log\ p(x_t|x_{<t}, z_{steer}) \quad (1)$$

It was demonstrated on a subset of sentences of the Yelp Sentiment dataset (Shen et al., 2017) that steering vectors can be used for shifting the style of a sentence x towards a dedicated target style using the vector arithmetic:

$$\hat{\mathbf{z}}_{target} = \mathbf{z}_{source} + \lambda\ \mathbf{z}_{\mathbf{\Delta}} \quad (2)$$

$\mathbf{z}_{source}$ is the steering vector that produces sentence x_{source}. $\mathbf{z}_{\mathbf{\Delta}} = \bar{\mathbf{z}}_{target} - \bar{\mathbf{z}}_{source}$ is the difference between the average of all steering vectors learned for sentences from the target and source domain. The steering vector $\hat{\mathbf{z}}_{target}$ can then be used to steer the model to generate a sentence x' that is similar to x but in the target style.

Moreover, layer activations have demonstrated utility in steering LLMs. Turner et al. (2023) exemplify that steering vectors, derived from contrasting activations for semantically opposed inputs like "love" and "hate" can guide LLM outputs during sentence completion. The difference in activations from such contrasting prompts at layer i can straightforwardly be added to another input's activations to steer outputs.

In this work, we add to this line of research a method that efficiently steers LLM outputs towards desired styles with notable control and transparency. In contrast to the aforementioned steering vector and TST techniques, it requires no additional optimization or prior knowledge about original styles. Unlike prompt engineering, our approach offers quantifiable adjustments in style, providing nuanced differences in responses without relying on vague intensity indicators in prompts, such as "extremely negative" versus "negative."

3 Methodology

We aim to modify the LLM activations for an input x to generate an output that is steered towards a specific style category $s \in S$. As shown in Eq. 3, this is achieved by finding style vectors $\mathbf{v}_s^{(i)}$ associated to s such that when added to the activations $\mathbf{a}^{(i)}(x)$ at layer i the output becomes steered towards s.

$$\hat{\mathbf{a}}^{(i)}(x) = \mathbf{a}^{(i)}(x) + \lambda \mathbf{v}_s^{(i)} \quad (3)$$

Style categories can be, for example, *positive* and *negative* for sentiment styles or different emotion classes such as *joy* and *anger*. The weighting parameter λ (Eq. 3) determines the influence

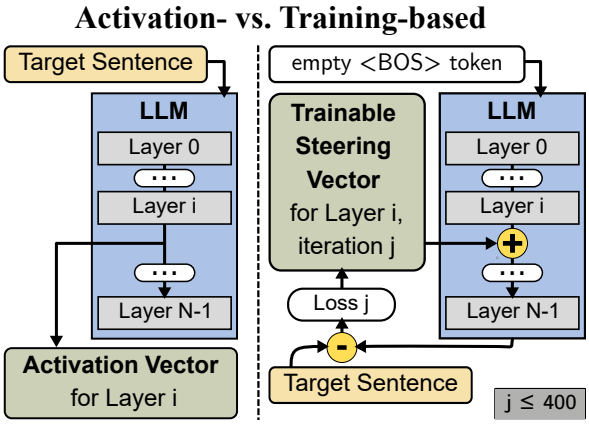

Figure 2: Extraction of an activation vector (left): The LLMs' values at layer i for a prompt in the target style are saved for later computation of style vectors. Trained steering vectors (right): The values of the vectors are optimized over $j = 400$ epochs such that the model produces a specified sentence in the target style from a simple beginning of a sentence (BOS) token.

strength of the style vector on the model's output and, thus, allows for more nuanced and controllable model steering compared to prompt engineering.

In this study, we compare two main approaches to calculate style vectors, namely *Training-based Style Vectors* (Sec. 3.1) and *Activation-based Style Vectors* (Sec. 3.2). Training-based style vectors are found from the generative steering vectors (Subramani et al., 2022). In contrast to this generative approach, activation-based style vectors are found by aggregating layer activations for input sentences from the target style (Turner et al., 2023). The basic assumption behind this is that LLMs internally adapt to the style of the input prompt when producing output, and thus, style vectors can be derived from its hidden states. These two methods are contrasted in Fig. 2 and introduced in more detail in this section.

3.1 Training-based Style Vectors

In the approach of Subramani et al. (2022) (see Sec. 2), an individual steering vector is learned for each target sentence. Thus, shifting the *source* style of an unsteered model output x towards a modified output x' (generated by steering vector $\hat{\mathbf{z}}_{x'}$) in the desired *target* style requires to compute a steering vector $\mathbf{z}_x$ that leads the unconditioned model to produce x (Eq. 2). This, however, leads to high computational costs and is impractical for online adaptation of an LLM prompted with arbitrary inputs. Furthermore, this vector arithmetic only works for style shifts when the source