Unnatural language processing: How do language models handle machine-generated prompts?

Corentin Kervadec and **Francesca Franzon**
Universitat Pompeu Fabra (UPF) / Barcelona
{name.lastname}@upf.edu

Marco Baroni
UPF and ICREA / Barcelona
marco.baroni@upf.edu

Abstract

Language model prompt optimization research has shown that semantically and grammatically well-formed manually crafted prompts are routinely outperformed by automatically generated token sequences with no apparent meaning or syntactic structure, including sequences of vectors from a model's embedding space. We use machine-generated prompts to probe how models respond to input that is not composed of natural language expressions. We study the behavior of models of different sizes in multiple semantic tasks in response to both continuous and discrete machine-generated prompts, and compare it to the behavior in response to human-generated natural-language prompts. Even when producing a similar output, machine-generated and human prompts trigger different response patterns through the network processing pathways, including different perplexities, different attention and output entropy distributions, and different unit activation profiles. We provide preliminary insight into the nature of the units activated by different prompt types, suggesting that only natural language prompts recruit a genuinely linguistic circuit.

1 Introduction

Neural language models (LMs) are parameterized probabilistic models that can assign a probability to any sequence of language tokens. Given that they are trained on huge amounts of natural language, we expect their statistics to mimic those of the latter. In this paper, we study what happens when a LM trained on English must process "unnatural language", that is, sequences that are extremely unlikely in English, as they are syntactically and semantically ill-formed.

We tackle this topic through the lens of *machine-generated prompts*, that is, automatically discovered input token sequences that optimize the model's performance in a target zero-shot task (Shin et al., 2020; Deng et al., 2022). It has indeed been widely observed that such prompts, while empirically effective, consist of nonsensical sequences of jumbled tokens. For example, using the popular AutoPrompt algorithm of Shin et al. (2020) and the OPT-1.3b language model (Zhang et al., 2022), we found that the prompt "*[X] Antarctica = sequelsStationrough [Y]*" outperforms reasonable human-crafted prompts such as "*[X] belongs to the continent of [Y]*" on the task of retrieving the continent a geographic body belongs to. Even more extremely, recent prompt generation methods find sequences of embedding vectors that do not correspond to items in the model vocabulary, but still outperform both human-crafted and machine-derived discrete prompts (Lester et al., 2021; Liu et al., 2023; Zhong et al., 2021). This state of affairs is paradoxical: why does a LM that has been trained to reproduce the statistics of natural language respond better to input sequences that are completely outside this distribution?

We present a detailed comparative study of how LMs internally process manually-crafted prompts and both discrete and continuous machine-generated prompts. While we do not solve the puzzle of why linguistically ill-formed machine-generated prompts are better than human prompts, we discover that there are fairly deep differences characterizing the various prompt types through all the processing stages of a LM, suggesting that the latter has fortuitously developed a distinct pathway to process unnatural language.

2 Related work

Understanding prompts. The advent of zero-shot prompting stimulated interest in the linguistic and semantic properties of prompts.[1] For example, Webson and Pavlick (2022) showed

[1]There is also related work on the effect of ablations such as word order permutations in the context of models fine-tuned for a specific task, such as natural language inference (e.g., Gupta et al., 2021; Pham et al., 2021; Sinha et al., 2021a,b).

that, with minimal fine-tuning, highly semantically irrelevant prompts can be as effective as prompts with pertinent semantic content. Starting with Wallace et al. (2019) and Shin et al. (2020), the fact that inscrutable machine-generated discrete prompts outperform natural language sequences has also attracted attention. For example, Deng et al. (2022) showed that constraining machine-generated prompts to be more "language-like" harms performance. Ishibashi et al. (2023) and Rakotonirina et al. (2023) studied how various ablations affect the performance of machine-generated prompts. The second study also demonstrated that it is possible to find discrete machine-generated prompts that are effective across a range of LMs. Khashabi et al. (2022) found that continuous prompts can be optimized to be near *any* arbitrary text in embedding space, while being equally effective. These studies focus on properties of the prompts themselves. We complement them with an analysis of how LMs respond when exposed to these prompts.

Understanding LMs More generally, understanding how LMs process unnatural linguistic input contributes to our understanding of their inner workings. Therefore, our study is also related to work on *interpretability* (Lipton, 2018), defined as the analysis of a trained model's decision policy. In particular, one can approach neural network interpretability by adopting a *mechanistic* paradigm, consisting in directly studying the weights and their activation in order to reverse-engineer the neural network. Successful mechanistic insights have been obtained in computer vision (Voss et al., 2021; Olah et al., 2020). Cammarata et al. (2020) is an example of mechanistic interpretability applied to Tranformer LMs. In this context, the Transformer feed-forward layers have been shown to behave like key-value memories (Geva et al., 2021). Notably, as shown in Dai et al. (2022), these memory slots, also called *knowledge neurons*, encode specific concepts acquired during pre-training. Even more interesting, manually editing these memories allows to causally control the prediction output (Meng et al., 2022), suggesting that they play a central role in language processing (see also Geva et al., 2022). In the present paper, we show that unnatural language processing is achieved by recruiting different knowledge neurons than the ones used for natural language processing.

3 Setup

3.1 Language model and tasks

OPT family LMs We conduct our analyses on OPT-350m and OPT-1.3b (Zhang et al., 2022), two pre-trained auto-regressive Transformer-based models trained on The Pile corpus (Gao et al., 2020), whose pre-trained weights are publicly available from HuggingFace. We choose auto-regressive models since LM development has increasingly shifted to this class, and OPT models since, in informal experiments, we found them to perform better on our tasks than comparable auto-regressive models available from HuggingFace (e.g., the GTP2 family). OPT models use a vocabulary set composed of 50,265 items.

Knowledge-retrieval tasks We base our experiment on the LAMA dataset (Petroni et al., 2019). Initially designed to probe factual knowledge and commonsense in LM, this dataset is a collection of $\langle r, s, o\rangle$ triplets describing a relation r between a subject s and an object o, e.g.: ⟨*continent of*, Lavoisier Island, Antarctica⟩. In particular, we use the TREx (ElSahar et al., 2018) subset, whose test set contains 41 relations, each with up to 1,000 tuples. All the machine-generated prompts are trained using the data collected by Shin et al. (2020), also containing 1,000 tuples per relation. Each relation defines a different knowledge retrieval task. We focused on these tasks because they require semantically contentful prompts (e.g., for the relation above, the prompt must carry some geographic information), as opposed to other setups where a prompt might simply have to describe the task at a meta-linguistic level ("*translate the following sentence into Chinese*"; "*does paragraph X entail paragraph Y?*", etc.).

3.2 Prompts

Terminology We refer to different methods to derive prompts as *prompt types*. We refer to the actual token sequences generated by a method for a certain task as *templates*.

Human prompts Human prompts (*human*) come from an augmented version of PARAREL (Elazar et al., 2021). PARAREL provides a set of near-paraphrase templates capturing each LAMA relation, e.g. "*[X] belongs to the continent of [Y]*". ParaRel enlarged the initial templates provided by LAMA using paraphrases from LPAQA (Jiang et al., 2020) and additional patterns mined from

Wikipedia. Each prompt was then evaluated by a set of human experts. We further manually augmented the set with more paraphrases, and we cleaned the prompt set, e.g., by removing templates not adapted to auto-regressive LMs.

Machine-generated prompts We compare *human* prompts with both discrete (*M-disc*) and continuous (*M-cont*) machine-generated prompts. The discrete ones are obtained using the popular Autoprompt (Shin et al., 2020) algorithm. For a given task, this algorithm generates a sequence of N tokens relying on a gradient-guided search in the discrete LM's vocabulary space. We set template length to $N = 5$, as it is the average human prompt length. The continuous machine generated prompts are obtained using Optiprompt (Zhong et al., 2021). For each task, Optiprompt generates a sequence of N continuous vectors through optimization in the LM's embedding space. Similarly to Autoprompt, we set $N = 5$. Machine-generated prompts are extracted using the LAMA-TREx training set (see above). 10 templates are obtained for each task by initializing training with different random seeds.

Template filtering We only use tasks for which we have, for each prompt type, at least one template reaching $> 10\%$ accuracy. We end up with 5.9 *human*, 8.3 *M-disc*, and 9.0 *M-cont* templates on average per task (across 21 tasks) for OPT-350m, and 6.3 *human*, 8.9 *M-disc*, and 10 *M-cont* templates on average per task (across 24 tasks) for OPT-1.3b.[2]

3.3 Diagnostic metrics

Accuracy We measure the effectiveness of a prompt type (*human*, *M-disc* or *M-cont*) by computing its micro-accuracy (following Zhong et al. (2021)), defined as the proportion of cases where the prompted LM succesfully assigned maximum completion probability to the ground-truth object. We average across templates and LAMA tasks. It is worth noting that, contrary to other works, we did not perform any filtering on the LM's output.

Input perplexity and output entropy We measure the average perplexity for each prompt type, defined as the exponentiated average negative log-likelihood of a `‘‘[subject] [template]’’` sequence, averaged across subjects, relations and templates. To characterize the LM probability distribution output, we also measure the average Shannon entropy of the output probability vector computed across all samples of the evaluation set.

[2]We attach the filtered template list as a supplementary archive.

Attention distribution We quantify how attention is distributed over input tokens following Ramsauer et al. (2021). For each attention head of each layer, we compute the average minimal number of attention values required to get a cumulative softmax probability mass of 0.90. This value ranges from 0% to 100%. Intuitively, given a row of an attention map of a transformer layer, it corresponds to the number (in %) of attention values you have to sum to reach 90% of the total attention. Because attention values are normalized, if the attention is flat then the score will be 90%. In contrast, if all the attention is focused on one token, then the score will be close to 1. This score decreases as the attention distribution becomes more peaky.

Knowledge neuron activation overlap Motivated by Geva et al. (2021) and Dai et al. (2022), who empirically demonstrated that Transformer feed-forward (FF) layers act as key-value memories, or *knowledge neurons*, we measure the activation overlap of the intermediate FF units (corresponding to memory keys) between different prompt types. More formally, for a given Transformer layer, let $x \in \mathbb{R}^d$ be the token-wise hidden representation contextualised by the self-attention operation. The FF layer can be expressed as:

$$u = f(x \cdot K^T + b_K) \quad (1)$$

$$FF(x) = u \cdot V + b_V \quad (2)$$

where $V, K \in \mathbb{R}^{d \times d_m}$ are the FF parameters, b_K, b_V their respective biases, and $f(\cdot)$ the non-linearity. K can be seen as a set of d_m keys, giving access to d_m “memory slots” stored in V. In order to quantify which knowledge neurons are being accessed during prompt processing, we look at the units $u \in \mathbb{R}^{d_m}$ corresponding to the weights associated to each key-value pair (Eq. 1). A high overlap means that the prompts activate the same knowledge neurons, indicating similar processing by the LM. On the opposite, a low overlap suggests that the prompts trigger different activation pathways in the LM. The measure is described in more detail in Appendix A.

Input similarity For each pair of same-task templates, we measure the cosine similarity of their embedded representations. We then compute the average to get similarities at the prompt-type level.