Next-Token Prediction in Arabic LLM

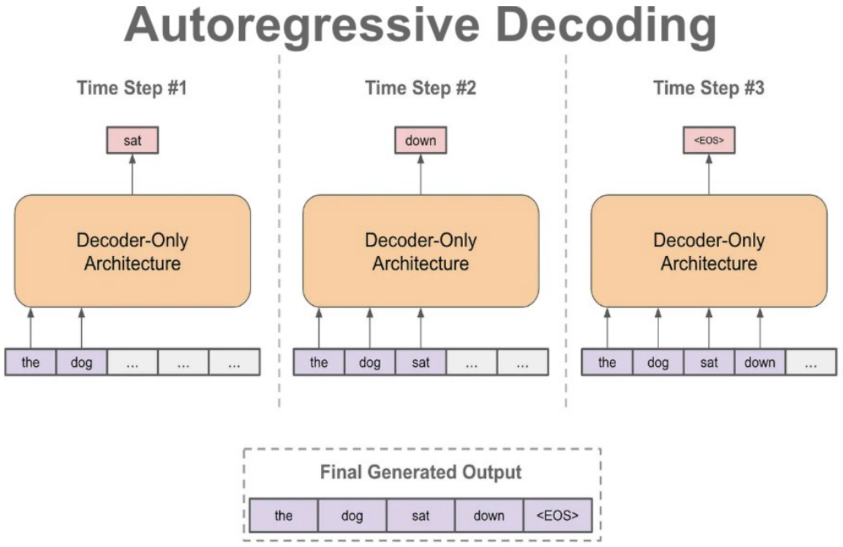
Project Description. Next-token prediction –also known as language modeling objective– serves as the backbone behind all modern advancements in large language models (LLMs), playing a pivotal role in their training on unlabeled text. Yet, the details and intricacies of how this next-token prediction operates, remain to be explored.

The next-token prediction objective is a self-supervised objective that is used heavily by LLMs for both pre-training and (supervised) fine-tuning. This objective involves the continuous classification of the token following the current token in a text sequence. And thanks to the utilization of a transformer, it can be applied across an entire sequence of tokens in a single forward pass during training. In essence, the approach involves iteratively executing two actions: 1) sampling some text from the dataset; and 2) training the model to predict the token that comes after each token within the sampled text.

This objective is self-supervised because we have access to an entire corpus of text. Thus, the ground truth next-token is always known. Consider a sequence of tokens given by $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$. Given a (decoder-only) transformer with parameters Θ, the language modeling objective tries to maximize the conditional probability of a token given preceding tokens. In other words, we want to maximize the probability of the model predicting the correct next token, given preceding tokens as context, or equivalently to minimize the loss function, using stochastic GD,

$$\mathcal{L}(\mathcal{X}) = -\sum_{i=1}^{n} \log\Bigg(\underbrace{\mathbb{P}(x_i \mid x_{i-k}, \cdots, x_{i-1}, \Theta)}_{\substack{\texttt{Conditional probabiblity of } i\texttt{th} \\ \texttt{token given } k \texttt{ preceding tokens} \\ \texttt{and model parameters } \Theta}} \Bigg),$$

where $\mathcal{L}$ is the language model loss over the full text corpus.



Therefore, LLM generating text, such as ChatGPT or Fanar, is essentially next-token prediction. As shown in the Figure, this follows the steps: 1) generate a current sequence of tokens; 2) generate the next-token using the LLM loss; 3) append this token to the sequence; and 4) repeat.

In this project, we are mainly interested in experimenting with the Wasserstein metric, which is natural in capturing the semantic structure of the data for a rich vocabulary like Arabic.

Project Type. 60% Engineering, 40% Research with a focus on contributing to a serious publication.

Internship Batch. Batch 1 from May 12 to July 12

Duties/Activities. Some code exists, but there is still some work to be done to make it usable, and to produce results.

Required Skills. Python programming.

Preferred Intern Academic Level. B.Sc. (3rd year or 4th year) or MS student.

Learning Opportunities. Students will be exposed to the exciting research in Large Language Models (LLMs), such as ChatGPT or Fanar for arabic LLM, and will learn the details of how these models work, i.e., the loss function and how text is encoded in the (decoder-only) transformer architecture. Also students will experiment with the Wasserstein word embedding, which is natural in capturing the semantic structure of the data for a rich vocabulary like Arabic.

Expected Team Size. It is preferable to have a team of 2 interns.

Mentors. Dr. Abdelkader Baggag, Dr. Sanjay Chawla