

Evaluating Linformer's performance on the WMT14 EN-DE machine translation task

Marco Pampaloni
Department of Computer Science
m.pampaloni2@studenti.unipi.it

1. Introduction

The Transformer architecture, since its introduction in 2017 [1], has revolutionized the field of natural language processing (NLP), reaching state of the art in various downstream tasks. Despite its massive parallelism capabilities, the Transformer struggles with longer sequence lengths due to its attention mechanism, which scales quadratically with the number of input tokens. This is a problem at both training and inference time.

For this reason, there has been a huge research effort in recent years to develop faster attention mechanisms, either exploiting the IO properties of the hardware these models run on [2], or by approximating the result of scaled dot product attention (SDPA).

The *Linformer* architecture [3] is an example of the latter approach. The authors first empirically show that the attention matrix is often low-rank, meaning that it could be approximated with an SVD decomposition by only keeping the largest singular values. This would of course introduce additional asymptotical complexity to the method, so the authors propose the adoption of linear projections on the keys and values matrices K, V to reduce their dimensionality and drive the computational complexity of the attention mechanism from $O(n^2)$ to $O(kn)$. The authors further show that the choice of k does not depend on the sequence length n , so that the scaling can be considered linear in n .

The standard multi head attention (MHA) introduced by [1] is computed as follows

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \\ \text{where } \text{head}_i &= \text{Attention}(QW^Q, KW^K, VW^V) \\ &= \underbrace{\text{softmax}\left(\frac{QW^Q(KW^K)^T}{\sqrt{d_{\text{model}}}}\right)}_{P_i} VW^V\end{aligned}$$

The Linformer attention first projects the key and value matrices into a space of dimension $\mathbb{R}^{k \times d}$ with projection matrices $E, F \in \mathbb{R}^{k \times n}$ and then computes MHA as before:

$$\overline{\text{head}}_i = \text{Attention}(Q, E_i KW_i^K, F_i VW_i^V)$$

This produces an attention matrix $\overline{P}_i \in \mathbb{R}^{n \times k}$, which is computed in time linear with n . Since the projection matrices E, F are fixed in size before training, an obvious downside of this approach is that the maximum sequence length n has to be known beforehand and the model cannot then scale to inputs with more tokens than this value. One workaround is to set the maximum sequence length n to a large number and handle shorter inputs by slicing the projection matrices along their columns before multiplying them with the inputs K and V .

This work aims at replicating the results of [1] and [3] and comparing the two architecture on the WMT14 EN-DE machine translation task.

1.1. Masking

In a standard Transformer architecture, sequences are usually batched together in order to exploit the model’s parallelism. This requires padding to be applied to the batch, but pad tokens should not contribute to the attention’s output. This can be achieved by masking the attention matrix, setting zeroing out its elements in correspondence with input pad tokens. This is often done by setting each corresponding element to $-\infty$ prior to applying the row-wise softmax.

In Linformer this method cannot be applied, as the attention matrix is linearly projected to a lower-dimensional space. Instead, we apply masking by zeroing out elements in Q, K, V corresponding to pad tokens. This ensures that the pad tokens do not contribute to the final attention result.

1.1.1. Causal masking

The standard Transformer was trained on the WMT14 EN-DE machine translation dataset, adopting an encoder-decoder architecture. The self-attention mechanism in the decoder’s layers need to employ causal masking in order for future (right) tokens not to contribute to the output of past positions. In the Transformer this again can be achieved by masking, in this case by adding an upper triangular matrix filled with $-\infty$ values to the pre-softmax attention matrix.

Linformer, which was originally developed for encoder-only architectures, does not allow for causal masking, because however you mask the resulting pre-activation attention matrix, future tokens leak into the past due to the linear projections of the K and V matrices.

The Linformer attention mechanism thus cannot be applied in the self-attention layers of the decoder, while it can safely be used in the cross-attention stages because of the lack of causal requirements. This hinders the full scaling potential of the encoder-decoder Linformer architecture, which is empirically shown in Sections 7.2 and 7.3.

2. Prior work

TODO

3. Data

As in [1], we used the WMT14 EN-DE [4] dataset comprised of about 4.5 million sentence pairs. For ease of use, we used the data hosted on a Kaggle repository¹, which conveniently collects all the english to german sentences in a single CSV file for training. Additionally, a validation (dev) and test dataset are provided. The validation set has not been used.

Because of resources constraints, we could not use or preprocess the entire dataset while keeping it in memory. Applying preprocessing on the fly during training would have slowed down the experiment significantly, rendering its results meaningless. Instead, we used half of the original training dataset and used the last 1% of it as a validation set.

We used the pretrained Huggingface² implementation of the BART [5] tokenizer, which employs Byte Pair Encoding (BPE) and was applied to the dataset prior to training as a preprocessing step. The vocab-

¹<https://www.kaggle.com/datasets/mohamedlotfy50/wmt-2014-english-german>

²<https://www.huggingface.co>

ulary of the tokenizer included about 50k tokens, in contrast to the vocabulary size of 37k from [1]. We truncated each sentence in the dataset to 256 tokens, padding on the right when necessary to enable training in batches. Each batch is randomly sampled from the training dataset and trimmed to the length of the longest non-padded sequence within the batch, significantly improving performance for batches with many short sequences (and, consequently, a high number of tokens).

The validation dataset is only used to log metrics during training and for scheduling the learning rate. The test dataset is instead used during the last evaluation step of each experiment.

TODO

- Average sequence length in training dataset

4. Architecture

This work focused on two architectures: the vanilla Transformer model and the Linformer. Both employed an encoder-decoder structure following [1]:

- 6 stacked encoder layers composed by Multi-Head self-attention, followed by a pointwise multi layer perceptron;
- 6 stacked decoder layers composed by Masked Multi-Head self-attention, followed by Multi-Head cross-attention and a pointwise multi layer perceptron.

In the standard Transformer model, each attention layer employs the scaled dot-product attention (SDPA) introduced by [1], both for the encoder’s self-attention and the decoder’s self-attention and cross-attention. The Linformer instead uses the linearized attention mechanism proposed by [3] everywhere except in the self-attention stage of the decoder’s layers, because of causality requirements (See Section 1.1.1).

Both the standard Transformer and the Linformer adopt residual connections and Layer Normalization applied after each attention block and feed forward layer as in [1]. Sinusoidal positional encoding and learned embeddings have been used for both architectures.

Table 1 shows the hyperparameters that have been set for both models, including every variant of Linformer tested.

Hyperparameter	Value	Description
d_{model}	512	Size of each embedding vector
h	8	Number of heads in multi-head attention (MHA)
d_k, d_v	64	Inner dimension of key and value vectors per head
d_{mlp}	2048	Hidden layer dimension of each pointwise MLP

Table 1: Hyperparameters shared by the tested models

One last notable change made to the architectures is the choice of the vocabulary size: BART tokenizer’s vocabulary has been resized to the next multiple of 8 in order to fully exploit the Tensor Cores³ of the Nvidia GPU (See Section 5) used during training, which accelerate matrix products when their sizes are divisible by 8.

TODO:

- Architecture diagram

³tensor cores: **TODO**

Hello World

5. Hardware

5.1. CPU bound

6. Experiments

6.1. Docker container (NVCR)

6.2. Mixed precision training

7. Results and Analysis

7.1. Model performance

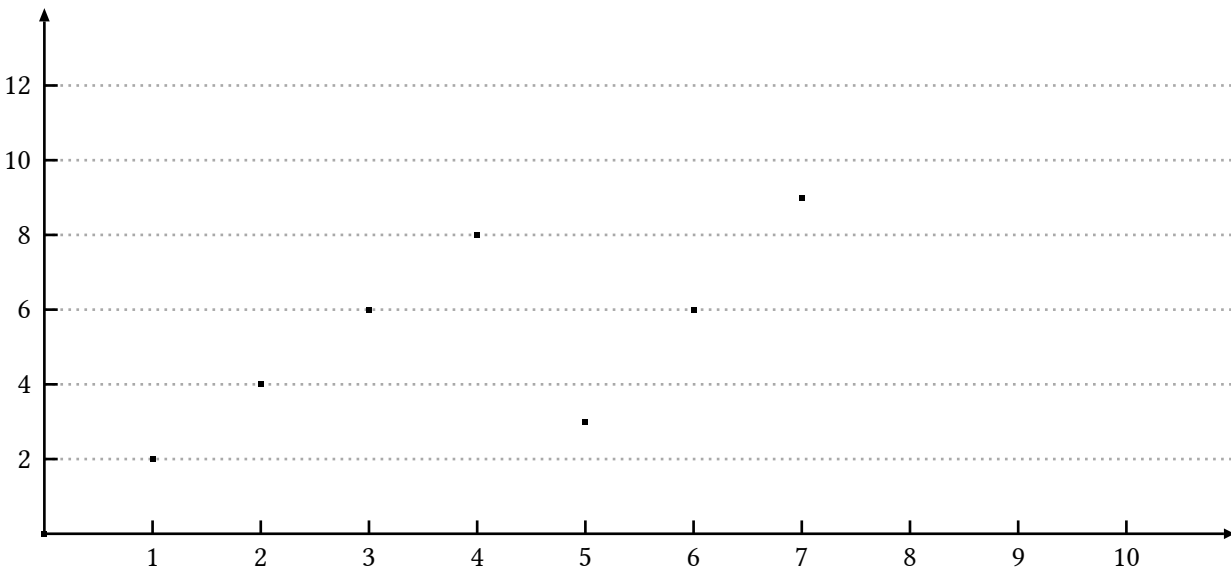
TODO

Model	PPL (test)	BLEU (test)
Transformer	3.41	29.92
Linformer (k=32)	3.96	30.08
Linformer (k=64)	3.84	27.74

Table 2: Linformer performance against a vanilla Transformer model on the WMT14 EN-DE (test) dataset. The Linformer has slightly worse perplexity than the Transformer, but their BLEU scores are comparable.

7.2. Training time

7.3. Inference time



Graph 1: Scatter Plot

8. Conclusions

Bibliography

- [1] A. Vaswani *et al.*, “Attention Is All You Need.” [Online]. Available: <http://arxiv.org/abs/1706.03762>
- [2] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, “FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness.” [Online]. Available: <http://arxiv.org/abs/2205.14135>
- [3] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, “Linformer: Self-Attention with Linear Complexity.” [Online]. Available: <http://arxiv.org/abs/2006.04768>
- [4] O. Bojar *et al.*, “Findings of the 2014 Workshop on Statistical Machine Translation,” in *Proceedings of the Ninth Workshop on Statistical Machine Translation*, O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, C. Monz, M. Post, and L. Specia, Eds., Baltimore, Maryland, USA: Association for Computational Linguistics, Jun. 2014, pp. 12–58. doi: 10.3115/v1/W14-3302.
- [5] M. Lewis *et al.*, “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.” Accessed: Jul. 25, 2024. [Online]. Available: <http://arxiv.org/abs/1910.13461>