Extending Transformer Decoder with Working Memory for Sequence to Sequence Tasks

Alsu Sagirova
Neural Networks and Deep Learning Lab, Moscow Institute of Physics and Technology
Dolgoprudny, Russia
alsu.sagirova@phystech.edu

Mikhail Burtsev
Neural Networks and Deep Learning Lab, Moscow Institute of Physics and Technology
Dolgoprudny, Russia
burtcev.ms@mipt.ru

Abstract

In this paper, we present the augmentation of the Transformer decoder with working memory. Transformers recently showed state-of-the-art performance on a wide range of NLP tasks. Transformer encodes existing elements of an input sequence but does not store implicit context-associated information. We propose to incorporate learnable working memory in Transformer to keep such information. Writing tokens from the vocabulary to the memory works like memorizing concepts related to the sequence. Such knowledge can be further employed during sequence processing.

We found that training of Transformer with working memory augmentation for fifteen epochs after memory-less pretraining for five epochs gives better BLEU results than training Transformer with working memory augmentation from scratch for twenty epochs.

Keywords: Transformer, MANN, Sequence to Sequence, Machine Translation
DOI:

Добавление рабочей памяти в декодер архитектуры Трансформер улучшает качество в задаче генерации последовательности

Алсу Сагирова
Лаборатория нейронных систем и глубокого обучения, Московский физико-технический институт (национальный исследовательский университет), Долгопрудный, Россия
alsu.sagirova@phystech.edu

Михаил Бурцев
Лаборатория нейронных систем и глубокого обучения, Московский физико-технический институт (национальный исследовательский университет), Долгопрудный, Россия
burtcev.ms@mipt.ru

Аннотация

В данной работе представлены способы добавления памяти в декодер Transformer. Transformer обрабатывает представления существующих элементов входной последовательности, но не хранит глобальную информацию, ассоциированную с входными данными, но не указанную явно в тексте. Мф предлагаем использовать обучаемую память, встроенную в Transformer для хранения необходимой информации. Запись в память токенов, представляющих собой слова из словаря, работает как механизм запоминания концепций, связанных с входной последовательностью. Такие знания могут быть использованы при дальнейшей обработке последовательности моделью.

Данное исследование показывает, что обучение с добавлением рабочей памяти в течение пятнадцати эпох с использованием предобучения без использования памяти в течение пяти эпох дает лучшие результаты метрики BLEU по сравнению с обучением модели с генерацией токенов в рабочую память в течение двадцати эпох.

Ключевые слова: Transformer, MANN, Sequence to Sequence, машинный перевод

1 Introduction

Transformer [1] is an encoder-decoder model successfully applied to various deep learning tasks. In particular, it is widely used for sequence-to-sequence learning [3]. The model key feature is an attention mechanism that enables access to a full context of the processed sequence during every token update. However, the Transformer architecture lacks storage for knowledge associated with input information and currently manipulates only with given input token representations. For a conceptual understanding of a text, it is essential to perceive context-related terms which are not mentioned. We hypothesize that the addition of working memory to accumulate knowledge token representations for future use during predictions generation will improve the model performance.

2 Model and Experimental Setup

We conduct experiments with three language pairs from the TED Talks Open Translation Project[1]: Portuguese-English, Russian-English, Turkish-English. For training and validation on each language pair, we use data samples that are no longer than 40 tokens.

We use the Transformer model with the following parameters: $N = 4$ layers in encoder and decoder, $d_{model} = 128, d_{ff} = 512, h = 8, P_{drop} = 0.1, batch = 64, warmup_{steps} = 4000$. For all experiments, we use best path decoding strategy for target sequence predictions.

In experiments we tested two schemes of memory addition: the first scheme is forcing different predictions to the memory slots while prepending target sequence with memory tokens. The second scheme is decoding tokens to the working memory such that memory positions are mixed with target predictions positions in the generated sequence. The architecture is depicted in Figure 1.

We calculate BLEU 4 [2] and METEOR [4] averaged for three runs on the validation set for each language pair to measure the models' performance.

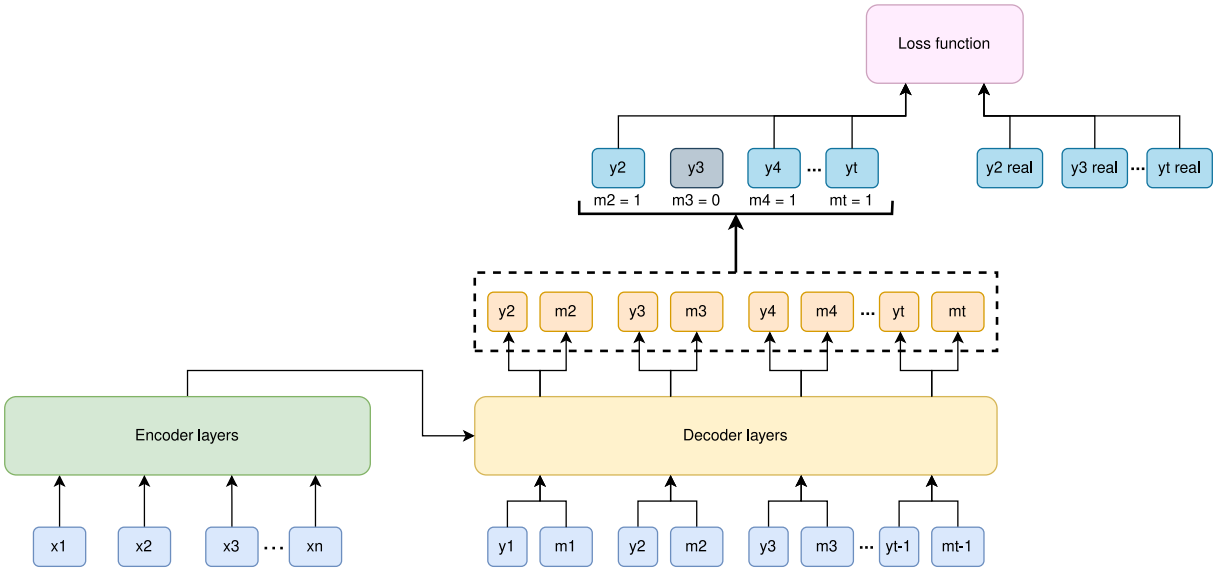

Figure 1: Transformer with decoder augmented with the working memory. The decoder input consists of tokens $y_1, \ldots, y_{t-1}$ generated to the moment and their memory flags $m_1, \ldots, m_{t-1}$. The final layer has an output size $= target\_vocabulary\_size + 2$ with the last two elements of the logits vector defining the memory flag value. The loss function takes target sequence predictions and real targets.

3 Experimental Results

BLEU scores for all our experiments are presented in Table 1. After 10 epochs of training for Pt-En and Ru-En datasets the original Transformer gives higher BLEU values than Transformer extended with

[1] https://www.tensorflow.org/datasets/catalog/ted_hrlr_translate

working memory and Transformer with additional leading memory tokens. These results and train loss plots in Figure 2, Figure 3 and 4 demonstrate that 10 epochs is inefficient for methods to converge and to learn how to incorporate memory contents into target predictions.

And 20 epochs training led to the convergence and improvement in scores for methods with generation of tokens to the working memory for all language pairs. When forcing different predictions in memory tokens, training with memory placed before real translation and masking first `[mem]` and `[start]` tokens gives better scores than working memory ($+0.71$ BLEU points for Pt-En pair, $+0.11$ BLEU for Ru-En data and $+0.3$ BLEU for Tr-En dataset). Working memory generation with nucleus sampling with $p_{nucleus} = 0.75$ outperformes enforcing different memory values in 20 epochs training setup for translations from Portuguese and Turkish.

The next set of results is for the case, when working memory generation with nucleus sampling decoding is trained on top of baseline model pretrained for 10 epochs. The comparison shows that combined training during 20 epochs outperforms pure working memory generation method for Pt-En and Ru-En datasets. Ru-En dataset is larger than Pt-En and Tr-En datsets and has smaller BLEU score performance gain between models with working memory and the baseline. In case of Tr-En we see that nucleus sampling with $p = 0.75$ works better with combined training and nucleus sampling with $p = 0.9$ gives higher BLEU value in pure working memory generation. We also test the working memory generation setting with ten leading `[mem]` tokens in encoder. Compared to the baseline with ten encoder `[mem]` tokens added before the input sequence, the proposed approach after 20 epochs gives higher BLEU scores for all three datasets.

Architecture	PT-EN		RU-EN		TR-EN	
	10 ep.	20 ep.	10 ep.	20 ep.	10 ep.	20 ep.
Transformer (baseline)	**27.02**	27.46	**19.86**	21.16	15.97	19.34
Leading mem tokens, different `mem` predictions	26.89	28.89	19.62	20.88	19.14	21.84
Working memory, different `mem` predictions	26.16	28.18	19.45	20.77	19.09	21.54
Working memory, $p_{nucleus} = 0.75$	24.71	28.71	19.29	20.80	19.44	21.97
Working memory, $p_{nucleus} = 0.9$	26.64	28.08	19.47	21.11	18.90	21.87
Baseline + Working memory, $p_{nucleus} = 0.75$	25.56	28.89	19.72	21.02	19.73	22.19
Baseline + Working memory, $p_{nucleus} = 0.9$	26.55	**29.87**	19.59	**21.25**	19.80	21.86
Baseline + Working memory, $p_{nucleus} = 0.9$ + 10 encoder `[mem]`	24.57	27.72	19.53	21.08	19.89	**22.32**
Baseline, 10 encoder `[mem]`	24.56	27.05	19.52	21.00	**20.10**	21.95

Table 1: BLEU scores (3 runs average value) for baseline and models with memory on TED validation set for three language pairs: Portuguese-English, Russian-English and Turkish-English. Scores are provided for 10 epochs training and for 20 epochs training. Best values are highlighted in bold.

Table 2 represents the METEOR scores. Models with working memory augmentation show the highest METEOR scores for all three language pairs. All best-performing methods outperform the baseline model for ≈ 8 points. For 20 epochs of training Ru-En dataset, both BLEU and METEOR highest values belong to the fine-tuning baseline with working memory and nucleus sampling decoding with $p_{nucleus} = 0.9$.

The working memory in decoder is designed to store generated tokens in memory. This allows us to analyse explicitly and interpret memory content. We provide examples of memory content for Pt-En translation in the Table 3.

Architecture	PT-EN		RU-EN		TR-EN	
	10 ep.	20 ep.	10 ep.	20 ep.	10 ep.	20 ep.
Transformer (baseline)	49.17	51.35	39.88	40.93	41.64	43.95
Leading mem tokens, different `mem` predictions	49.36	51.19	39.61	40.96	40.60	43.63
Working memory, different `mem` predictions	56.84	60.03	46.82	48.23	47.34	50.74
Working memory, $p_{nucleus} = 0.75$	56.52	**60.13**	46.64	48.14	47.92	50.75
Working memory, $p_{nucleus} = 0.9$	**57.39**	59.74	46.70	48.35	47.96	50.80
Baseline + Working memory, $p_{nucleus} = 0.75$	57.25	59.81	46.85	47.94	48.30	**51.14**
Baseline + Working memory, $p_{nucleus} = 0.9$	57.10	60.11	**47.02**	**48.45**	42.70	48.94
Baseline + Working memory, $p_{nucleus} = 0.9 +$ 10 encoder `[mem]`	55.55	59.02	46.75	48.25	**49.04**	51.07
Baseline, 10 encoder `[mem]`	49.04	51.49	39.49	40.99	41.86	43.81

Table 2: METEOR scores averaged for 3 runs for baseline and models with memory on TED Portuguese-English, Russian-English and Turkish-English validation sets. Scores are provided for 10 epochs training and for 20 epochs training. Best values are highlighted in bold.

Source sentence (Portuguese)	E emitam certificados falsos.
Target sentence(English)	And issue rogue certificates.
Working memory, different `[mem]` predictions	[start]and they showed false (certifi)(cer)(works)(official)(lab) (police)(blindness)(ai)(uns)(cou)certificate.[end]
Working memory, $p_{nucleus} = 0.75$	[start]and they emi(ted)(d)(ted)(tte)(ted)(ted)(ted)(d)(che) (ted)his false author.[end]
Working memory, $p_{nucleus} = 0.9$	[start]and (and)(and)(and)(and)(and)(and)(and)(and) (and)they issued false certificate set.[end]
Baseline + Working memory, $p_{nucleus} = 0.75$	[start] and they (launched)(launched)(launched)(launched) (launched)(launched)(launched)(launched) (called)(defe)launched false false .[end]
Baseline + Working memory, $p_{nucleus} = 0.9$	[start] and they (told)(covered)(went)(have)(discovered) (feature)(owned)(were)(feature)(diss)covered false certificate and they shed false protection.[end]
Baseline + Working memory, $p_{nucleus} = 0.9+$ 10 encoder `[mem]`	[start] and they (emi)(invited)(ast)(spark)(was)(inge) (put)(emi)(up)(was)emitted their false.[end]

Table 3: Examples of tokens stored in working decoder memory for translation from Portuguese to English. [start], [end] denote starting and ending tokens of the sequence correspondingly. Words or subwords in parentheses are tokens generated into the working memory. Remaining tokens represent the translation prediction.

4 Conclusion

In this work, we proposed, implemented and studied the working memory augmentation of the Transformer decoder. We performed experiments with three datasets from the TED Talks Open Translation Project for the machine translation task. We demonstrated that training Transformer first without memory and then with working memory outperforms the baseline. We also tested different decoding strategies and schemes of memory token generation. Our results confirm that storing generated tokens into memory improves the quality of model predictions.

References

[1] Attention is All you Need / Ashish Vaswani, Noam Shazeer, Niki Parmar et al. // Advances in Neural Information Processing Systems / Ed. by I. Guyon, U. V. Luxburg, S. Bengio et al. — Vol. 30. — Curran Associates, Inc., 2017. — Access mode: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] Bleu: a Method for Automatic Evaluation of Machine Translation / Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu // Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. — Philadelphia, Pennsylvania, USA : Association for Computational Linguistics, 2002. — Jul. — P. 311–318. — Access mode: https://www.aclweb.org/anthology/P02-1040.

[3] Raffel Colin, Shazeer Noam, Roberts Adam et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. — 2020. — 1910.10683.

[4] Lavie Alon, Agarwal Abhaya. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments // Proceedings of the Second Workshop on Statistical Machine Translation. — Prague, Czech Republic : Association for Computational Linguistics, 2007. — Jun. — P. 228–231. — Access mode: https://www.aclweb.org/anthology/W07-0734.

Appendix. Training loss plots.

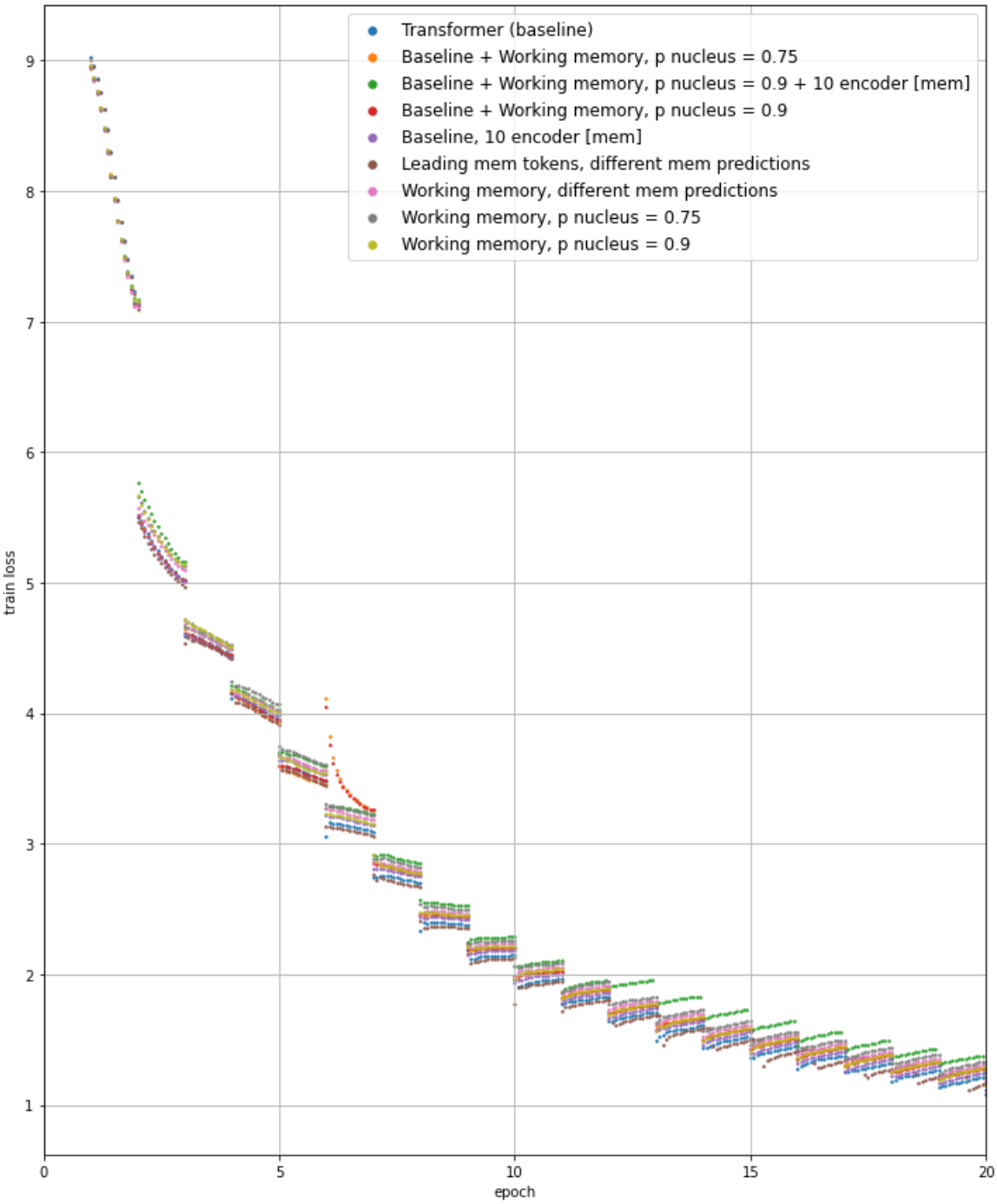

Figure 2: Train loss plots for TED Portuguese-English dataset experiments.

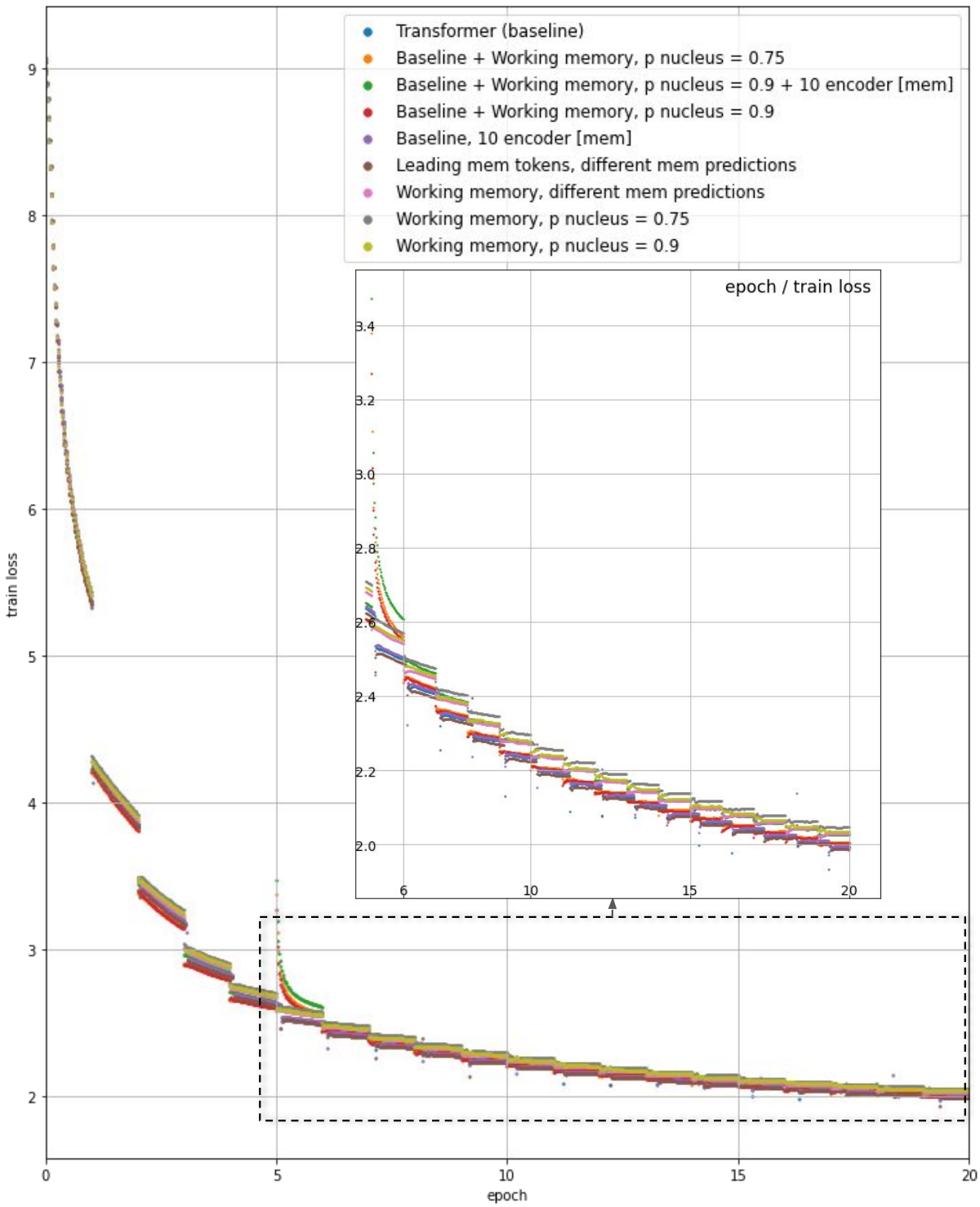

Figure 3: Train loss plots for TED Russian-English dataset experiments.

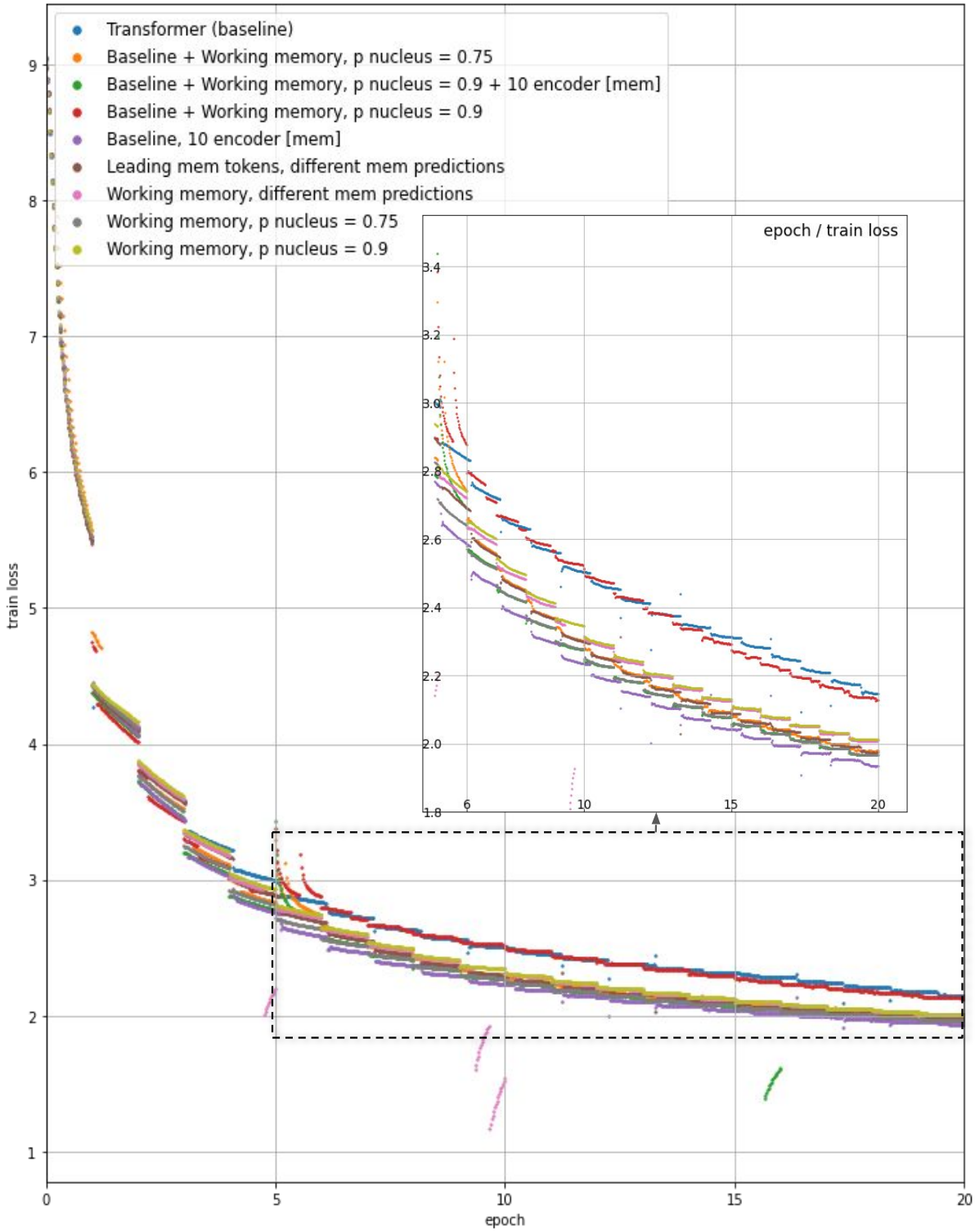

Figure 4: Train loss plots for TED Turkish-English dataset experiments.