Advanced Methods in Text Mining
Winter Semester 2022 / 2023
Assignment 3 – Transformer
Final Version

Instructor: Dr. Rafet Sifa, Tobias Deußer
University of Bonn

Introduction

In this assignment, we will have an in-depth look at attention, transformer, and their application. This assignment will have 2 main parts totalling 100 points (pts) and a bonus part worth 50 bonus points. The coding exercises will be implemented using the programming language python 3.9.

1. Transformers from scratch (50 pts)

Implement a Transformer model as described in Vaswani et al. 2017 from scratch, i.e. do not use a model from e.g. Hugging Face, i.e. you should use the basic functionality of PyTorch and implement the attention mechanism on your own. The model should have an encoder, a decoder, positional encoding added to the input, an embedding layer, and a wrapper combining all of this.

You will use this model in the following question to evaluate and train it, so the training, masking, and batching will be discussed there.

2. Relation Extraction with Transformers (50 pts)

Consider the task of joint named entity recognition (NER) and relation extraction (RE), where the objective is to extract key information in form of entities, e.g. persons, locations, companies, numeric values, etc., from a document, sentence, or similar and establish in what relation they are to each other. Take this example from Deußer et al. 2022, in which we want to extract key performance indicators (KPIs) and their current year (cy) and prior year (py) values:

> "In 2021 and 2020 the total net revenue$_{\text{kpi}}$ was \$100$_{\text{cy}}$ million and \$80$_{\text{py}}$ million, respectively."
>
> total net revenue$_{\text{kpi}}$ – 100$_{\text{cy}}$, total net revenue$_{\text{kpi}}$ – 80$_{\text{py}}$,

With this definition of joint named entity recognition and relation extraction in mind, construct a full pipeline for the **KPI-EDGAR** dataset[1], introduced in Deußer et al. 2022. This pipeline should integrate the following steps:

1. Reading in the `.json` file.
2. Apply Sub-Word-Tokenization.
3. Train[2] two different encoder models (train until you reach the best performance on the validation set in terms of relation extraction F_1 score):

[1] Download the `.json` file from `https://github.com/tobideusser/kpi-edgar`. The *IOBES* tag are already included in there.

[2] We recommend using a proper training routine, batching, dataloader, and evaluation process. You can use libraries like PyTorch Lightning (`https://www.pytorchlightning.ai/`) to make this step easier.

   (a) An untrained transformer, preferably the one you implemented in the previous question. You do not need to implement the backpropagation algorithm, you are allowed to use the automatic differentiating function of PyTorch.

   (b) A BERT model, we recommend the `bert-base-uncased` version, available here: `https://huggingface.co/bert-base-uncased`

4. Implement a joint named entity and relation extraction routine, that first extracts named entities and them links them together.

5. Predicting on the test set and reporting your achieved accuracy and relation extraction F_1 score.

3. Theory (50 bonus pts)

In this section we will revise and investigate some theoretical insights of transformers.

1. Autoencoders: Explain what autoencoders are and how they are related to sequence-to-sequence modeling. How are they related to Transformers? (6 pts)

2. Normalization: What are the differences between the notions of batch and layer normalization? Explain your answer. (6 pts)

3. Autoregressiveness: Show how training and inference of RNNs and Transformers for sequence-to-sequence modeling are done in an autoregressive fashion. What trick do we use when training the decoder part? Explain your answer in detail. (8 pts)

4. Positional embeddings: Explain in your own words what positional embeddings are and why they are introduced? What alternative methods (other than the sin-cos based method we discussed in the lecture) are usually considered for incorporating positional embeddings? Explain your results verbally as well as formally. (10 pts)

5. Attention mechanism: What are the main differences between the attention models considered in RNNs and the Transformer-models? What is the main motivation to incorporate attention to RNNs? Why do we consider using multi-headed attention in Transformers? What is modeled through it? Explain your answer in detail. (10 pts)

6. Popular transformers: Explain in your own words what Bidirectional Encoder Representations from Transformers (BERT) Devlin et al. 2018 and XLNet Yang et al. 2019 are. How are they different from the transformer architecture we covered in the lecture? What are their differences? (10 pts)

References

Deußer, Tobias, Syed Musharraf Ali, Lars Hillebrand, Desiana Nurchalifah, Basil Jacob, Christian Bauckhage, and Rafet Sifa (2022). "KPI-EDGAR: A Novel Dataset and Accompanying Metric for Relation Extraction from Financial Documents". In: *Proc. ICMLA*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). "Xlnet: Generalized autoregressive pretraining for language understanding". In: *Advances in neural information processing systems* 32.