FACULTY
OF ELECTRICAL
ENGINEERING
CTU IN PRAGUE

MASTER THESIS

Bc. Alexander Gažo

Algorithms for Document Retrieval in Czech Language Supporting Long Inputs

Department of Computer Science

Supervisor of the master thesis:	Ing. Jan Drchal, Ph.D.
Study programme:	Open Informatics
Study branch:	Artificial Intelligence

Prague 2021

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Gažo Alexander** Personal ID number: **495795**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Computer Science**

Study program: **Open Informatics**

Specialisation: **Artificial Intelligence**

II. Master's thesis details

Master's thesis title in English:

Algorithms for Document Retrieval in Czech Language Supporting Long Inputs

Master's thesis title in Czech:

Metody document retrieval nad českými texty vhodné pro zpracování dlouhých vstupů

Guidelines:

The task is to develop methods of document retrieval to be deployed in the fact-checking scenario. Focus on Czech corpora and the ability to deal with long input strings.
1) Familiarize yourself with methods of document retrieval aimed for the fact-checking task. Focus on the Czech language and neural network approaches. Aim for modern model architectures that are able to deal with long inputs (Longformer, Reformer, etc.)
2) Work with the Czech Wiki FEVER and ČTK data, both supplied by the supervisor.
3) Select an appropriate method and modify it for the supplied data. Consider student-teacher training based on models pretrained on different languages and training from scratch.
4) Evaluate and compare the methods.

Bibliography / sources:

[1] Thorne, James, et al. "FEVER: a large-scale dataset for fact extraction and verification." arXiv preprint arXiv:1803.05355 (2018).
[2] Thorne, James, et al. "The fact extraction and verification (fever) shared task." arXiv preprint arXiv:1811.10971 (2018).
[3] Chang, Wei-Cheng, et al. "Pre-training tasks for embedding-based large-scale retrieval." arXiv preprint arXiv:2002.03932 (2020).
[4] Beltagy, Iz, Matthew E. Peters, and Arman Cohan. "Longformer: The long-document transformer." arXiv preprint arXiv:2004.05150 (2020).
[5] Kitaev, Nikita, Łukasz Kaiser, and Anselm Levskaya. "Reformer: The efficient transformer." arXiv preprint arXiv:2001.04451 (2020).

Name and workplace of master's thesis supervisor:

Ing. Jan Drchal, PhD., Centrum umělé inteligence

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **03.08.2021** Deadline for master's thesis submission: **13.08.2021**

Assignment valid until: **19.02.2023**

Ing. Jan Drchal, PhD. Supervisor's signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

I declare that I carried out this master thesis independently and only with the cited sources, literature, and other professional sources.

I understand that my work relates to the rights and obligations under Act No. 121/2000 Sb., the Copyright Act, as amended.

In Prague on ..... signature

I would like to thank my supervisor Ing. Jan Drchal, Ph.D., for his support and friendly approach. Thank you to my colleagues, with whom we could have endless passionate discussions regarding our project, motivating each other all the while enjoying the time spent together. A great deal of gratitude belongs to my girlfriend Tina, who helped guide me towards the best possible outcome, and last but not least, I would like to thank all my family and friends for their everlasting support.

Title: Algorithms for Document Retrieval in Czech Language Supporting Long Inputs

Author: Bc. Alexander Gažo

Department: Department of Computer Science

Supervisor: Ing. Jan Drchal, Ph.D., Artificial Intelligence Center FEE CTU

Abstract: The document retrieval task is a well-studied problem of finding the relevant subset of documents to the provided search query. Recent advances in the field of Natural Language Processing (NLP), namely the transformer architecture (Vaswani et al., 2017) and BERT model (Devlin et al., 2018) provide a new approach to document retrieval. The document retrieval in this thesis is motivated by the Czech fact-checking task, which is an important challenge in the modern world. In this thesis, we apply the latest research achievements to the transformer's attention mechanism (Bahdanau et al., 2015), decreasing the space and time complexity, allowing for longer input sequences (documents). We then study whether the processing of whole articles, unlike only theirs paragraphs, improves the performance of the retrieval models.

Keywords: document retrieval, fact-checking, long-inputs, Czech language, NLP, BERT, TFIDF

Název práce: Metody document retrieval nad českými texty vhodné pro zpracování dlouhých vstupů

Autor: Bc. Alexander Gažo

Katedra: Katedra počítačů

Vedúci práce: Ing. Jan Drchal, Ph.D., Centrum Umělé Inteligence FEL ČVUT

Abstrakt: Úloha vyhľadávania dokumentov (document retrieval) je dobre známy problém nájdenia relevantnej podmnožiny dokumentov k vyhľadávanemu dotazu. Nedávny pokrok v oblasti spracovania prirodzeného jazyka (NLP), konkrétne architektúra transformera (Vaswani et al., 2017) a model BERT (Devlin et al., 2018), poskytujú nový prístup k vyhľadávaniu dokumentov. Vyhľadávanie dokumentov v tejto práci je motivované úlohou overovania faktov v českom jazyku, ktorá je dôležitou výzvou pre moderný svet. V tejto práci aplikujeme najnovšie výskumné výsledky na mechanizmus pozornosti (attention) transformera (Bahdanau et al., 2015), znižujúc priestorovú a časovú zložitosť, čo umožňuje prácu s dlhšími vstupnými sekvenciami (dokumentami). Na záver skúmame, či spracovanie celých článkov, na rozdiel od iba ich odsekov, zlepšuje výkonnosť vyhľadávacích modelov.

Kľúčové slová: vyhľadávanie dokumentov, overovanie faktov, dlhé vstupy, český jazyk, NLP, BERT, TFIDF

Contents

List of Figures

List of Tables

Introduction

Motivation

At its early stages, the internet was envisioned to become the pinnacle of joint human effort to gather and easily retrieve expert knowledge on virtually any topic. However, with many laypeople connected to the internet, extensive and aggressive advertising, and adversarial agents such as foreign powers or simply malicious individuals, the information on the internet is becoming harder to be trusted. The information overload created by these agents results in an "opaque" state of the internet, where relevant and accurate information is hard to find in the mass of similarly sounding, non-professionally written sources. Fact-checking the claims manually, therefore, becomes very expensive and time-consuming - bordering on unfeasible. Nevertheless, multiple projects focused on fact-checking have emerged. Various platforms such as Instagram[1] and Twitter[2] have incorporated fact-checking mechanisms, mainly on viral post and posts of politicians.

In Czechia and the Slovak Republic, a popular project is Demagog[3], whose goal is to verify politicians' claims. The claim verification is carried out manually using primary sources. Similar foreign projects are PolitiFact[4], Factcheck.org[5], and Washington Post Fact Checker[6]. Multiple past "public debates", such as the European migrant crisis or the current coronavirus pandemic, have highlighted the need for such systems since there is a need for accurate and up-to-date information. The process is very labor-intensive, and thus, there is a natural demand for automatization.

The recent advances in natural language understanding, mainly the introduction of transformer architecture (Vaswani et al., 2017) and the BERT model (Devlin et al., 2018), led to new research on the use of neural methods in fact-checking. The FEVER[7] paper (Thorne et al., 2018a) has led this effort since 2018, focusing on creating a dataset meant for training neural models. They succeeded in creating a sizeable human-annotated dataset and were able to train a pipeline model on it. The model first retrieved relevant documents (the document retrieval task) and then labeled the initial claim based on these documents. With better models released every year, the long-term goal is to create a model capable of correctly assessing a claim's truthfulness and provide satisfactory evidence. However, creating helpful tools for journalists to assist them in the fact-checking scenario is the goal for now.

On the other hand, the advances also provide new ways of creating false information on a large scale. Such an example is the recently introduced GPT-3 model (Brown et al., 2020), which is able to generate human-sounding English texts. The potential ability of adversaries to flood the internet with fake news articles emphasizes the need for scalable fact-checking tools.

Automatic fact-checking should not be thought of as the miracle cure to the fake news problem, which is much more complex and deserves a society-wide approach.

[1] https://about.fb.com/news/2018/05/hard-questions-false-news/
[2] https://blog.twitter.com/en_us/topics/product/2021/introducing-birdwatch-a-community-based-approach
[3] https://demagog.cz
[4] https://www.politifact.com/
[5] https://www.factcheck.org/
[6] https://www.washingtonpost.com/news/fact-checker/
[7] https://fever.ai/

AI in Journalism

This thesis is one of the multiple theses written by the fact-checking team at ČVUT, led by Ing. Jan Drchal, Ph.D., as part of the AI in Journalism project, supported by the Transformation of Journalisms Ethics in the Advent of Artificial Intelligence (TL02000288)[8] grant from the Technology Agency of the Czech Republic. Our team focuses on creating a Czech fact-checking dataset from the ground up, and developing usable Czech models for the fact-checking task, inspired by FEVER (Thorne et al., 2018a) and Binau and Schulte (2020). The dataset is based on Czech news articles provided in cooperation with the Czech News Agency[9] – we refer to the completed dataset as the ČTK dataset. Our colleague Ullrich (2021) describes the creation of the ČTK dataset, which consisted of building a Czech annotation platform, working with annotators (students of the Faculty of Social Sciences at Charles University, one of our partners), and analyzing and cleaning up the gathered data. The works from colleagues Dědková (2021) and Rýpar (2021) deal with various aspects of document retrieval – the use of hybrid (multi-stage) models and the performance of different embedding paradigms, respectively.

Transformer Models

The original transformer architecture (Vaswani et al., 2017) and BERT-model (Devlin et al., 2018) are based on feeding fully-connected feed-forward networks with token representations aggregated from the whole text, meaning that the information from the input tokens (words or subwords) is adjusted according to their whole context. This mechanism, named "attention" by the authors (Bahdanau et al., 2015), introduces quadratic time complexity by "attending" to all the input tokens for each input token. The input of BERT is thus usually limited to 512 tokens as a design choice. Working with this restriction, we decided to split the ČTK articles into paragraphs and perform document retrieval on them, theoretically losing joint article meaning.

In this thesis, we study this practice and compare it to working with full articles using BERT-based models with altered attention mechanisms such as Nyströmformer (Xiong et al., 2021), Longformer (Beltagy et al., 2020) and Reformer (Kitaev et al., 2020). The changes allow for longer inputs without increasing the computation cost, compromising in other areas.

Thesis Outline

This thesis focuses on the document retrieval part of the fact-checking pipeline. Specifically, it deals with models suitable for processing long Czech documents.

The first chapter deals with fact-checking as a formal task and the past advances in this field. The next chapter formally defines document retrieval and introduces traditional as well as novel approaches. The FEVER dataset, its Czech-translated version, and the ČTK dataset are in-depth described in the third chapter. We also analyze the length of the documents upon which the datasets are built. We dedicate the last two chapters to the proposed solutions and the results of the evaluation.

[8] `https://starfos.tacr.cz/cs/project/TL02000288`

[9] Česká Tlačová Agentúra (ČTK)

1. Fact-Checking

1.1 Task Description

We see fact-checking as a crucial part of journalism, being the means of covering important events truthfully. The goal is to compare the reported fact or claim to the current state of the world or its assumed truthful approximation, in our case, the corpus of ČTK news articles (ČTK infobank) and the Czech Wikipedia. We refer to this textual world-state representation as the knowledge base. From the comparison, we can "check" the claim's truthfulness, proclaiming it true, false, or unverifiable. Thus, we can think of the task of fact-checking as a classification problem with labels True, False, and Not Enough Info (NEI). The claims are usually one or a few sentences long strings stating some fact or facts. There are platforms, such as Demagog.cz, which also use a category labeled "misleading", reserved for statements that are technically true but imply additional, false meaning. For now, our models do not use this category, although it may be used in further research.

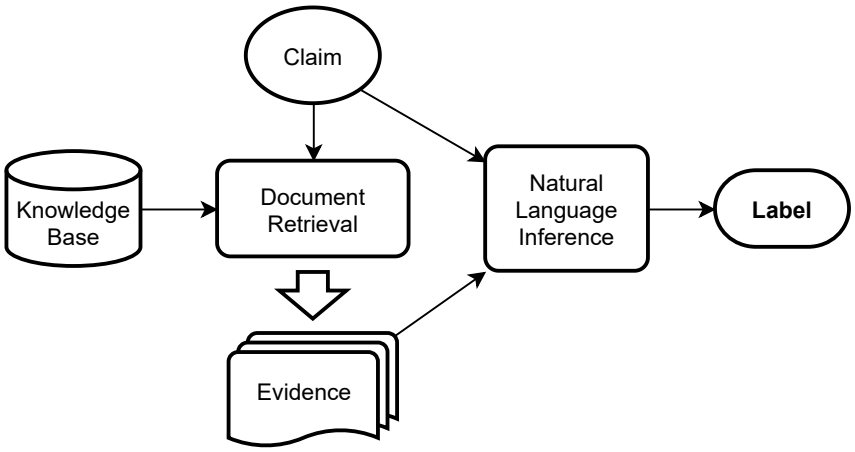


Figure 1.1: Fact-checking pipeline diagram.

To label a claim true or false, we also need to provide evidence validating or refuting the claim. For our uses, the evidence is one or multiple news articles in ČTK infobank, which we consider an adequate approximation to the actual world state. The infobank naturally provides a somewhat distorted image of reality since the journalists already assume some world knowledge, and the article form may further compress the original information. Therefore, approaches based on language comprehension face the challenge of inferring real-world knowledge from somewhat distorted data.

The above-described approach can be formulated as a sequence of subtasks (pipeline) depicted in Figure 1.1. The first step is to retrieve a collection of relevant documents from the knowledge base. This step is called document retrieval. Then, the Natural Language Inference task is performed, classifying the claim based on the selected documents.

1.2 Related Work

There exist several traditional fact-checking projects such as Demagog[1], PolitiFact[2], Factcheck.org[3], and Washington Post Fact Checker[4], with focus on fact-checking politicians' claims as well as general viral news.

Figure 1.2: Example claim, verdict, and its justification from the Czech Demagog project. Available at a permanent address `https://demagog.cz/vyrok/19500`.

Regarding automatic fact-checking, as of the time of writing, we are not aware of any other Czech fact-checking datasets other than (Přibáň et al., 2019).

Regarding the English language, Thorne et al. (2018a) created the Fact Extraction and Verification (FEVER) dataset; the first large-scale dataset focused on open-domain fact-checking. The pipeline (Figure 1.1) also appears in (Thorne et al., 2018a), with the addition of the Sentence Selection step, where sentences forming the evidence are extracted after the document retrieval step.

The authors of the FEVER dataset also announced a shared task (Thorne et al., 2018b) in which "The task challenged participants to classify whether human-written factoid claims could be `SUPPORTED` or `REFUTED` using evidence retrieved from Wikipedia." The results presented in (Thorne et al., 2018b) confirmed the pipeline approach, as the best performing submissions adhered to the design. The authors

[1] `https://demagog.cz`
[2] `https://www.politifact.com/`
[3] `https://www.factcheck.org/`
[4] `https://www.washingtonpost.com/news/fact-checker/`

continue organizing shared tasks with various modifications, such as (Thorne et al., 2018c), with claims designed to mislead the models, and the current task[5], focusing on a combined structured (tables present in articles) and unstructured (articles' texts) knowledge base.

[5] `https://fever.ai/task.html`(accessed 26th July 2021)

2. Document Retrieval

The term document retrieval refers to the task of finding relevant information to user queries in a large set of records (documents). One can think of document retrieval as a search in a vast database of documents. In this view, web search services, such as Google, are also a form of document retrieval, with the database of documents being all the accessible web pages on the internet.

For our uses, the query is the claim to be fact-checked, and the document database is a collection of relevant documents, which we call the knowledge base.

In this chapter, we define the task of document retrieval and introduce traditional and novel (neural) approaches to solving it. After describing the traditional word-weighting approaches, we examine crucial concepts behind the proposed neural models. We then describe the framework which allows us to use the neural models for the document retrieval task.

2.1 Formal Description

The task can be formally described (Chang et al., 2020) using a scoring function (sometimes referred to as ranking function)

$$f_{\mathcal{D}} \colon \mathcal{D} \times \mathcal{Q} \to \mathbb{R} \tag{2.1}$$

that maps a document-query pair (d, q) to a score $f(d, q)$. Then, given a query, the documents in the query-document pairs with the highest scores are considered to be the proposed solution to the task.

As mentioned above, this definition also fits the descriptions of a range of other tasks such as open-domain question answering (Chen et al., 2017) or recommendation systems.

2.2 Approaches to Document Retrieval

The organizers of the **T**ext **Re**trieval **C**onference (TREC) (Craswell et al., 2021) classify retrieval methods into three categories (with examples):

- Traditional – TFIDF, BM25
- Neural Networks using Language Modeling (NNLM) – BERT-based
- Neural Networks – others

We explore NNLM and traditional approaches, emphasizing the NNLM approach while using the latter as the baseline. The term "Neural Networks" is reserved for methods that use neural networks only in some parts of its implementation (such as word embeddings, see Subsection 2.4.1) and do not rely explicitly on language modeling. We do not explore such methods in this thesis, and therefore, by neural approach, we refer to the NNLM methods. While the two work on entirely different principles, we will use both to generate an indexed version of the knowledge scope.

This chapter further introduces the most common document retrieval methods and new models with great potential while explaining the main points of the theoretical background.

2.3 Traditional Approaches

Traditional approaches covered are weighting schemes, assigning a query-dependent score to each word in each document in the database.

2.3.1 TFIDF

The traditional approaches are motivated by the intuition that relevant documents will contain the same words as those present in the query. Longer documents are at an advantage since there is a higher chance of the relevant words being present. Therefore, the term count is often normalized by the number of all terms in the document. This simple metric is called term frequency (TF).

TF can be ineffective if some of the terms in the query are very common in the document database. This issue is resolved by introducing inverse document frequency, which informs how common the term is across all documents. The base version of IDF:

$$\mathrm{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \ . \tag{2.2}$$

Multiplying these two metrics, we get the TFIDF weighting scheme. The weight of term t in document d in document database D is then

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D) \ . \tag{2.3}$$

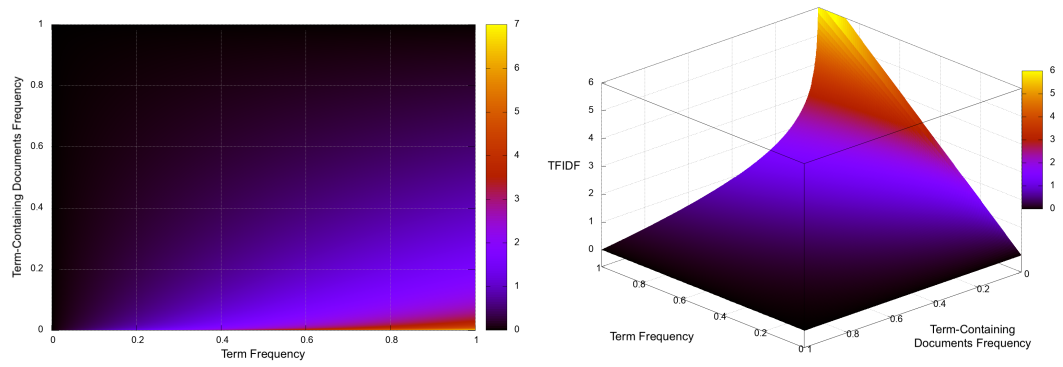


Figure 2.1: Visualization of TFIDF depending on the term frequency and the frequency of the documents containing the term.

We have explained how to compute a term's weight ("importance") for a specific document. The intuitive formula for query score $f(q, d)$ is then

$$f_D(q, d) \approx \sum_{t \in q} \mathrm{tfidf}(t, d, D) \ . \tag{2.4}$$

Such a metric can be viewed as a weighting scheme.

Since

$$t \notin d \ \Rightarrow \ \mathrm{tf}(t, d) = 0 \ \Rightarrow \ \mathrm{tfidf}(t, d, D) = 0 \ ,$$

we know that only words in the corpus' vocabulary are important, and thus we can precompute the TFIDF values for each document and word from the corpus' vocabulary. We obtain $|D| \times vocabulary\_size$ matrix V of TFIDF values. Here, to simplify length normalization, we normalize the matrix V so that each row has a unit norm. This step removes the need for normalization in the TFIDF formula, and we may use raw term

count instead of the normalized frequency. To get the relevance of each document to the query q, that is $f_D(q, d)$, we first represent the query q as a bag of words vector (BOW), corresponding to the columns of our precomputed TFIDF matrix V, ignoring words that appear only in the query. The resulting score is then the normalized (not to favor long queries) dot product of a row of the matrix and the BOW representation of the query q denoted $\vec{q}$. We can obtain the scores for every query document pair by matrix multiplication:

$$f_D(q, d) = \frac{V\vec{q}}{|\vec{q}|} \in \mathbb{R}^{|D|} , \tag{2.5}$$

provided that matrix V is row-normalized (euclidian norm of each row is equal to one). Please note that this is equivalent to computing the cosine similarity for each document and query vector pair.

This function is one of the first and is still widely used (Beel et al., 2016) weighting schemes for document retrieval.

Over the years, multiple versions of the TFIDF approach have appeared, differing slightly in formulas or weights of the factors. One of such versions is Best Match 25.

2.3.2 Best Match 25 (BM25)

A more complex term weighting scheme is Best Match 25 (Robertson et al., 1995):

$$\text{BM25}(q, d) = \sum_{t \in q} \text{idf}(t, d) \cdot \frac{(k_1 + 1) \cdot \text{c}(t, d)}{k_1(1 - b + b \cdot (L_d / L_{avg})) + \text{c}(t, d)} \cdot \frac{(k_3 + 1) \cdot \text{c}(t, q)}{k_3 + \text{c}(t, q)} , \tag{2.6}$$

where $\text{c}(t, d)$ is the raw count of the term t in the document d. L_d and L_{avg} are the current document's length the average document length, respectively.

The first term is IDF. The second term is TF with two tuning parameters k_1 and b. Parameter k_1 corresponds directly to TF scaling, while parameter b corresponds to scaling by the inverse of the document's length. The third term follows the same idea as the second term, but the b parameter equivalent is unnecessary since we only have one fixed query. Schütze et al. (2008) note that this term is only useful for longer queries q, such as whole paragraphs.

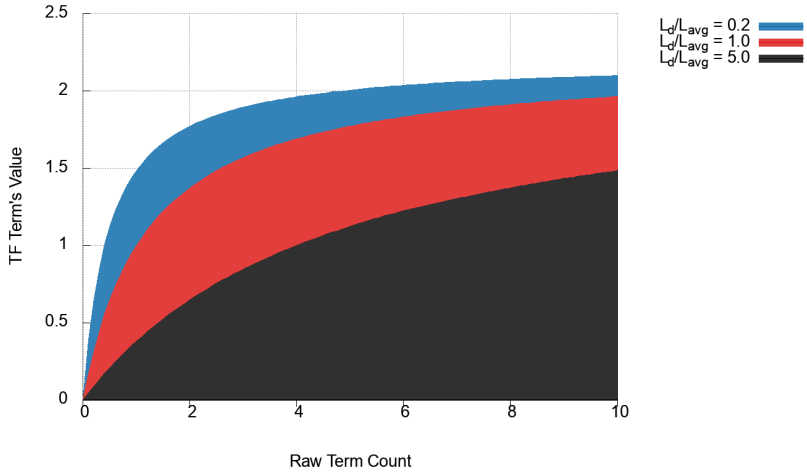


Figure 2.2: TF term's relation to the length of the current document for $k = 1.2$ and $b = 0.75$.

Since this weighting scheme introduces tunable parameters, it can be trained on data. If the data is not available, Schütze et al. (2008, Section 11.4.3) recommend using $k_1, k_3 \in [1.2; 2]$ and $b = 0.75$.

Final Thoughts

Some of the apparent disadvantages of traditional approaches are the lack of semantic meaning that comes from the independence on the word order and the exact word matching. The former may be improved by using n-grams and the latter using character-level features instead of words, especially in inflected languages such as Czech. On the other hand, TFIDF is, to this day, a very well-performing low-computation cost ranking function, and as reported by Yang et al. (2019), BM25 (if tuned well) is still a solid baseline capable of beating even much more complex neural models. Fine-tuned BM25 was demonstrated[1] to outperform every document-retrieval submission from the original FEVER shared task (Thorne et al., 2018b).

2.4 Neural Approaches

Neural approaches using language modeling are neural models that were trained to predict the correct word given a preceding text. Recurrent neural network (RNN) architecture was traditionally used for this task. The RNN was fed encoded tokenized input, returning a hidden state that was next fed into the RNN again along with the next input token's representation. Its hidden states could also be used to encode the input into a fixed-length vector representation, which can be further used to either classify the input or use another RNN (decoder) to generate another sequence. Intuitively, the vector contains the meaning of the original input. Such an approach is the Sequence to Sequence (seq2seq or encoder-decoder) model architecture (Sutskever et al., 2014) with significant use-case in machine translation.

As an improvement to the seq2seq models, the attention mechanism was introduced by Bahdanau et al. (2015). The authors viewed RNNs as a bottleneck to improving the architecture's performance. They proposed an automatic method of enriching parts of the input with other relevant parts of the input, eliminating the need to process the input as a whole. The attention mechanism is in-depth described in Subsection 2.4.2.

The research paper "Attention Is All You Need" by Vaswani et al. (2017) improved the RNN-based seq2seq by introducing the transformer architecture, depicted in Figure 2.3, by substituting the RNN with multiple "encoder blocks", that is, attention layer and feedforward network with skip-connection. This resulted in simpler architecture (RNN architectures tend to become very complex when trying to avoid the vanishing gradient problem (Pascanu et al., 2012)), parallelization of the computation, and, most importantly, an improved performance. The improvement showed that the attention mechanism was crucial to the past achievements of the seq2seq models, hence the paper's name.

Then **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT) (Devlin et al., 2018) model was introduced. It was "designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers". The model can then be fine-tuned on a specific task using a small amount of labeled data and one additional output layer.

This was a brief description of how a sequence of research papers – "Sequence to Sequence Learning with Neural Networks" (Sutskever et al., 2014), "Neural Machine Translation by Jointly Learning to Align and Translate" (Bahdanau et al., 2015), "Attention Is All You Need" (Vaswani et al., 2017), and "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (Devlin et al., 2018) – led to a shift from RNN-based NLP approaches to, now SOTA, BERT-based approaches. The

[1] `https://github.com/castorini/anserini/blob/ad4caeb59ec512d0ce07412e6c4b873a8b841da4/docs/experiments-fever.md`

resulting research focus on BERT-based models is sometimes referred to as BERTology.

In the following sections, we will explain the concepts mentioned above in greater detail.

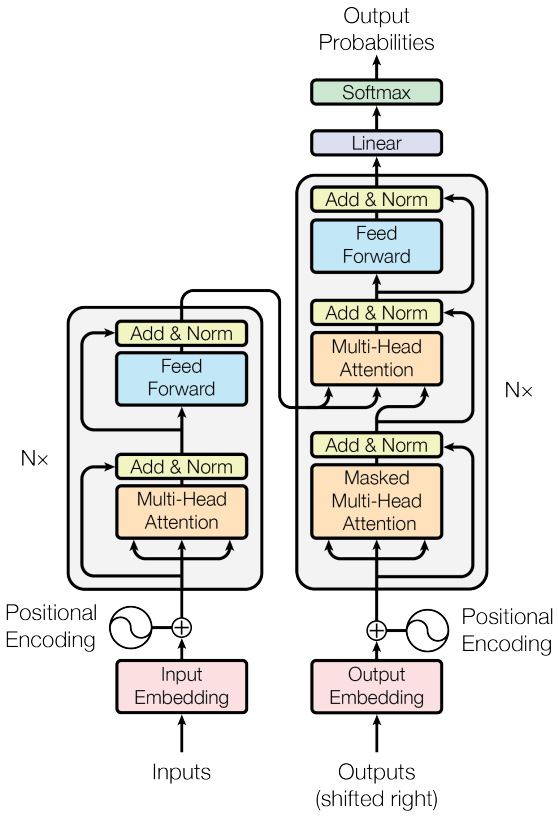


Figure 2.3: The transformer model architecture. Reprinted from (Vaswani et al., 2017).

2.4.1 Word Vectors

Unlike traditional approaches, neural networks require numbers as their inputs. Therefore the words (tokens) have to be encoded into a numeric representation with a fixed length. The word encoding is just another linear layer (with $vocab\_size \times embedding\_dim$ weight matrix), which can be trained from scratch, or one could use pre-trained (for example (Pennington et al., 2014)) word vectors (the rows of the weight matrix).

2.4.2 Attention Mechanism

The main feature of the transformer models is the attention mechanism. It adjusts each token's embedding based on all the other tokens present in the current input. This is done by computing three different linear projections of the original input tokens and then, based on certain similarities, combining them back together. The attention mechanism is also referred to as self-attention since all the tokens in one input attend to the same input.

The projections are computed using trainable matrices W_Q, W_K, and W_V, producing queries Q, keys K, and values V. Then, we compute the dot product ("similarity") between the keys and the queries QK^T, illustrated in Figure 2.5. The result is normalized by the inverse factor of the square root of the projection dimension. Then softmax is applied, and the result is used as the weights for the weighted average of all value

vectors. To put it into an equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \ . \tag{2.7}$$

Therefore, for every token, the result is the weighted average of all the value vectors, based on the similarity between the queries and the keys. The result is called "scaled dot-product attention".

The computation is performed multiple times in parallel with independent projection matrices to make the mechanism more robust. The results are then concatenated and projected into the final dimensionality. This extended attention is called multi-head attention. The intuition is that different heads might learn to attend to different patterns and meaning subspaces.

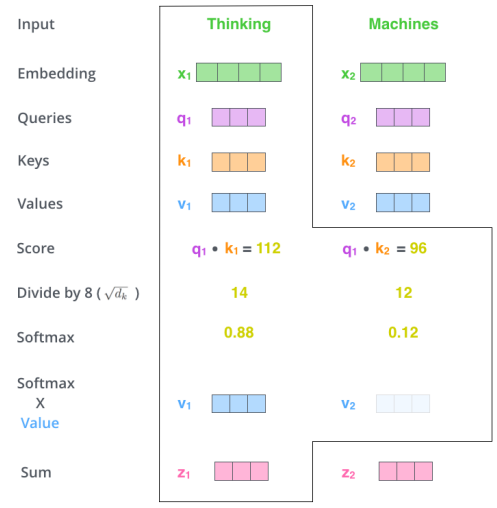


Figure 2.4: Example of self attention mechanism with projection dimension 64 and the input "Thinking Machines". Reprinted from (Alammar, 2018).

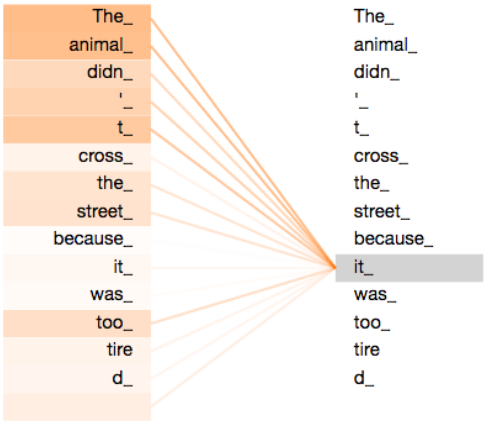


Figure 2.5: Self-attention mechanism example, where the token for the word "it" is paying attention to "the animal", effectively being substituted for it. The orange color represents the values from QK^T. Reprinted from (Alammar, 2018).

2.4.3 Transformer and BERT

As mentioned earlier, the transformer architecture no longer uses RNNs. The omission meant that the input is not longer sequentially processed. While allowing for better performance through parallelization, the position information is lost, and therefore the information has to be added to each input token. The solution is embedding the position into the same dimension as the word (token) vectors and then adding the positional embedding to the token's embedding. The position embedding is fixed, non-trainable, and defined as a vector of trigonometric functions applied to the position index, depicted in Figure 2.6. The authors provide a detailed explanation in (Vaswani et al., 2017, Section 3.5).

While seq2seq architecture can work with arbitrary sequence types, BERT is a language model working with text input. Specifically, it is just the encoder part of the seq2seq architecture, meant for computing real-valued fixed-length vector representation of the input tokens, that can further be used by other neural models (usually shallow, fully connected feedforward networks). The encoder is trained on a large amount of unlabeled text data (corpus) in a phase called pre-training. Due to a large number of the model's parameters ($\approx$ 110 million in its base form), this phase consumes

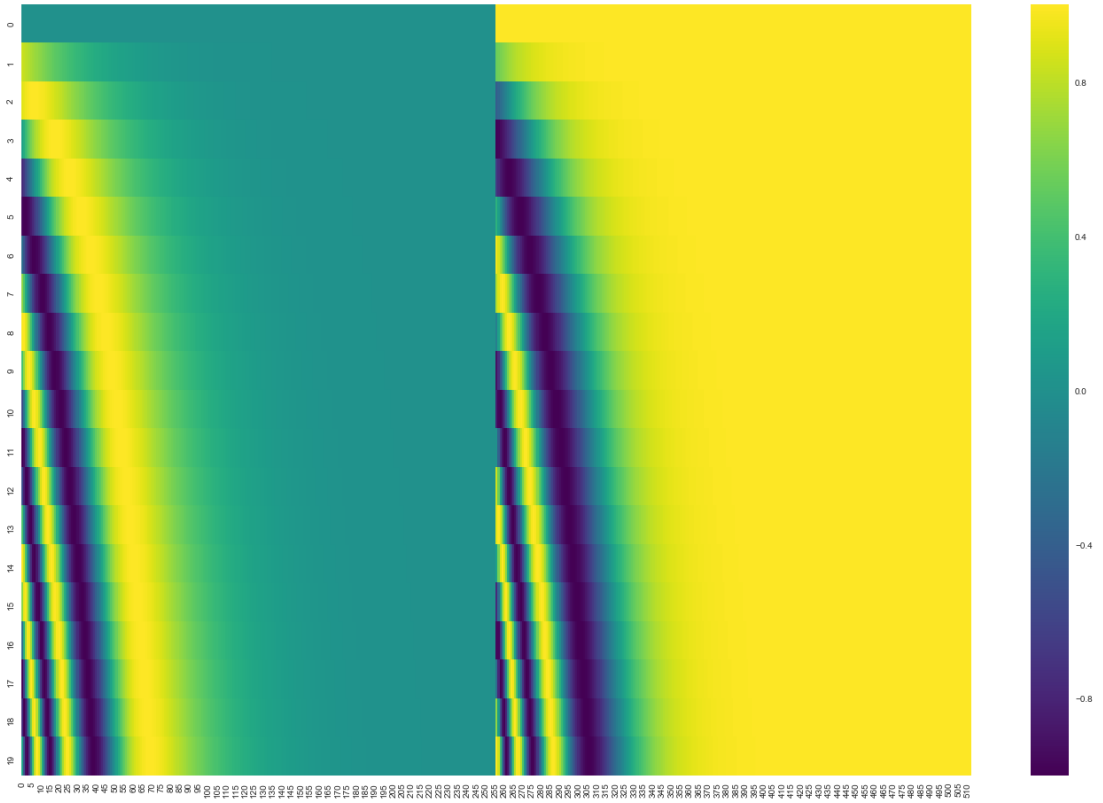


Figure 2.6: Transformer positional encoding as described in (Vaswani et al., 2017). Input length is 20 tokens with the embedding dimension of 512. For better visualization, the originally alternating sin and cos functions are divided to the left and right half, respectively. Reprinted from (Alammar, 2018).

a significant amount of power. The model can then be fine-tuned, that is, trained for the task we want to apply it on, such as Named Entity Recognition (NER) (Sang and Meulder, 2003), Natural Language Inference (NLI) (Bowman et al., 2015a; Williams et al., 2017), or Question Answering (QA) (Rajpurkar et al., 2016). This is usually done by adding one fully connected layer after the encoder output and relatively short training (of all parameters) with the usage of a smaller amount of labeled data.

Intuitively, the pre-training "teaches" the language and word meanings to the model while fine-tuning only "explains" the final task to it.

2.4.4 Encoder Utilization in Document Retrieval

We have, so far, introduced the encoder model, but we have yet to describe how to use it in document retrieval. We can look at the neural-model-generated encodings in a way similar to the traditional approaches.

Traditional approaches result in a sparse representation of the knowledge base with the feature dimensionality equal to the corpus's vocabulary size (the vocabulary can also contain n-grams or character n-grams if it is advantageous). Therefore, it can be considered a weighting scheme, assigning "importance" to each word present in the document.

Neural approaches in this paper generate a fixed-length real-valued vector representation for both the query and all the documents in the knowledge base. The dense representation is inherently hard to interpret, and the meaning can be derived only from the relative position of different documents' representations (vectors). Hence the $f(q, d)$ score from Equation 2.1 is the distance between the vector representations of q

and d. The distance metric used may be cosine similarity, dot product, or a fast approximation such as FAISS (Johnson et al., 2021). This approach allows us to preprocess the whole knowledge base, and thus during runtime, only the query representation needs to be computed. The disadvantage is that we do not utilize the attention mechanism to consider each query-document relation individually and, today less relevant problem, the need for additional space to store the precomputed representation.

Neural models can also be used directly as the scoring function as in Equation 2.1 and use the document-query pair (d, q) as their input (the cross-attention model). This can, according to Chang et al. (2020), lead to better performance. However, since the query is provided at runtime, the evaluation cannot be precomputed and has to run for every document each time we enter a new query. Because of that, and because of the large number of documents in the knowledge base, this approach is unfit for real-world application.

There exists a hybrid approach capable of utilizing the cross-attention model. It runs the model on a small subset of the corpus, typically pre-selected by a non-neural model. The neural model only reranks the gathered sentences. Our colleague Dědková (2021) provides a detailed look into the use of such methods in Czech document retrieval.

Generating Representations

Generating the dense representations, also called embeddings since we project the input into a vector space with a specific dimension, is a well-studied area of research. We restrict ourselves to neural methods.

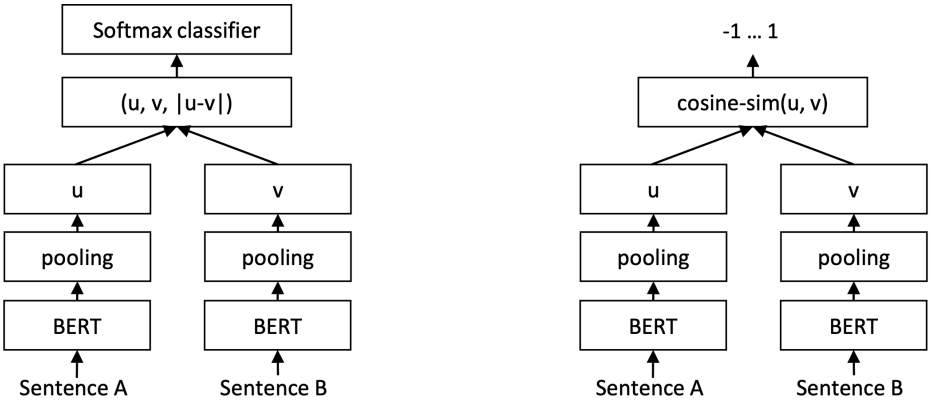


Figure 2.7: The left figure depicts SBERT architecture for classification, and the right depicts regression architecture. Both BERTs in each figure share their weights. Reprinted from (Reimers and Gurevych, 2019).

There are multiple ways to generate the input's embedding from the model-processed tokens. Sentence-BERT (Reimers and Gurevych, 2019) adds a pooling layer, either `MAX` or `MEAN`, and fine-tunes the model using two different approaches, depending on the task (see Figure 2.7). They compare their method against averaging all the output tokens' representations, selecting the out representation of special input token `[CLS]` situated at the front of the input, and against averaging input tokens' GloVe (Pennington et al., 2014) word vectors. The authors note that if the fine-tuning step is not performed, the BERT model provides little to no improvement to averaging input tokens' word vectors.

The authors of (Chang et al., 2020) proposed a different approach, designed directly for document retrieval, in their case, over Wikipedia articles. They apply a fully-

connected linear layer to the [CLS] token's representation and also pre-train the BERT model on three additional document retrieval tasks:

- **Inverse Cloze Task (ICT)** - Training the model to embed sentences close to the paragraph from which they were selected. The selected sentences are omitted from the paragraph in the training step.
- **Body First Selection (BFS)** - Training the model to embed sentences close to the paragraphs in the same article. The sentences are selected in such a manner as to reasonably expect this task to succeed. In the case of training on Wikipedia, the sentences are selected from the first paragraph (summary).
- **Wiki Link Prediction (WLP)** - Training the model to embed sentences close to Wikipedia articles from which there exists a hyperlink to the article of the sentence.

The authors reported significant improvements compared to using only MLM for the pre-training phase.

2.4.5 Language Support

The models described in previous sections are capable of multilingual learning. In representation learning, we strive to train models, which for two similar texts in different languages provide similar representations. Some (Lample and Conneau, 2019) use additional training tasks utilizing parallel data to achieve this property. Multilingual models are expected not to outperform monolingual models. However, a large multilingual model XLM-RoBERTa (XLM-R) (Conneau et al., 2019) was able to perform competitively with monolingual models on the English language, as well as on the Czech language (Macková and Straka, 2020). XLM-R performs well even without explicitly training on a multilingual task, using only unlabeled data. It was trained on a vast English corpus and one hundred small corpora in different languages. Lample and Conneau (2019) further confirm that training a model in this way provides competitive results. Macková and Straka (2020) hypothesize that the greedy SGD forces the model to exploit existing similarities between the learned languages since by exploiting these similarities, the model saves its capacity allowing for better performance during training. Other recently introduced multilingual models are SlavicBERT (Arkhipov et al., 2019) – tuned for NER and trained using Russian news and Bulgarian, Czech, Polish, and Russian Wikipedias – and mBERT – a BERT representation model trained on multilingual data.

The available monolingual Czech models are RobeCzech (Straka et al., 2021) and Czert (Sido et al., 2021), both language representation models based on BERT architecture. Authors of RobeCzech compare their model with other models mentioned in this section on five NLP tasks, claiming improvement over SOTA in all of them.

As of this writing, we are not aware of pre-trained models supporting long inputs in the Czech language. In the next section, we present methods for utilizing pre-trained models without the need for training from scratch.

2.4.6 Distillation

Knowledge distillation (Bucila et al., 2006; Hinton et al., 2015) is a compression method in which we train a compact model (student) to mimic the predictions of a more complex model (teacher). The main idea is to feed both the teacher and the student the same input and training only the student using the error between the student's and the teacher's predictions as the objective function. The method has been successfully

applied to various model ensembles, producing a single model able to perform competitively (Hinton et al., 2015). Sanh et al. (2019) apply the method to BERT, producing DistilBert – a BERT model initialized from pre-trained BERT but only with half the number of layers. Trained DistilBERT retains 97% of the original model's performance, according to the authors.

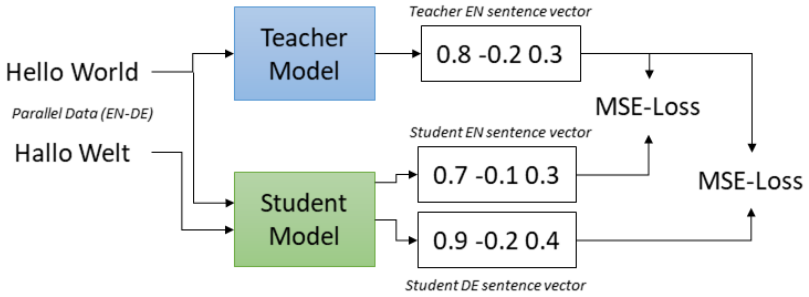


Figure 2.8: Multilingual distillation training example on EN-DE parallel sentences. Reprinted from (Reimers and Gurevych, 2020).

The method can also be utilized to teach new languages to a model using parallel language data, that is, a collection of the same sentences in multiple languages. Reimers and Gurevych (2020) propose "multilingual distillation" in which the teacher model is a well-performing monolingual representation model. The teacher model is fed only sentences of its "native" language, while the student model is fed sentences in all the languages. The student model is trained using the sum of all the losses over all the languages as the objective function. The process is depicted in Figure 2.8. The aim is to train the student model to generate the same representations of sentences with the same meaning, regardless of their written language.

2.4.7 Transformer's Computational Limits

While the benefits of the attention mechanism are undeniable (Vaswani et al., 2017), the mechanism also introduces quadratic time and space complexity. This is the result of the "one on one" approach coupled with the non-linear softmax operation, which prevents various optimization techniques from being applied. The attention mechanism thus acts as a bottleneck in applications, where a longer input is required. Transformer models, therefore, generally limit the input length to 512 tokens, padding shorter inputs and truncating longer inputs. That might be adequate for most applications, but in document retrieval, valuable information might get lost (see Section 3.4).

In addition, other parts of the transformer architecture consume a large amount of memory. During training, the activations of every encoder layer have to be stored for back-propagation, and feedforward layers in the encoder blocks are themselves nontrivially large.

2.4.8 BERTology

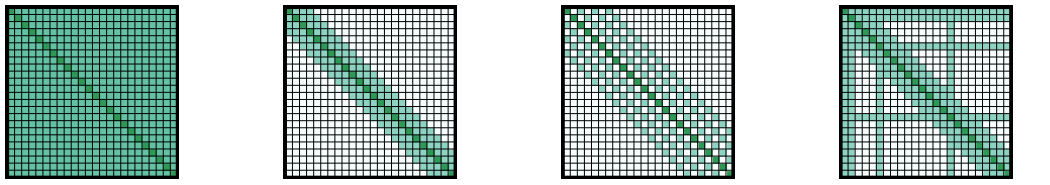
The research focus on BERT-like models resulted in many papers introducing minor and major changes to the BERT's architecture. We will focus on modifying the attention mechanism so that longer inputs are not the limiting factor during computation.

Longformer

The Longformer architecture (Beltagy et al., 2020) represents the simplest form of the attention mechanism modification. Instead of computing the whole QK^T, only certain regions (usually specific diagonals) are calculated, thus reducing the model time complexity allowing for longer inputs.

The authors proposed attention windows that ensured that each token attended only to a fixed number of the neighboring tokens. Given multiple encoder blocks of the transformer architecture (Figure 2.3), each token eventually attends to all the input tokens, similarly to CNNs. Additionally, a "dilated" window attention was used, where the window contains gaps of parametrized size. The authors claim that using different sizes of dilation per attention head proved to be beneficial.

In different applications, the model performs better with different attention patterns. It might require that some positions attend to the whole input; for example, in QA, it is essential to attend to the question. Thus, the authors use another pattern in the attention mechanism. It arises by allowing some positions to attend to the whole input and making the whole input sequence attend to them, as in Figure 2.9.



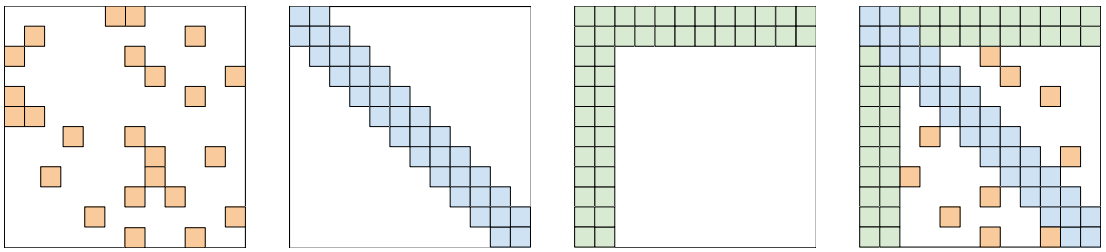
(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

Figure 2.9: Original attention and Longormer attention patterns. Reprinted from (Beltagy et al., 2020).

To utilize the benefits of this sparsified attention, the authors had to implement a custom CUDA kernel capable of computing parts of matrix multiplication effectively.

BigBird

The BigBird model (Zaheer et al., 2020) adopts a similar approach. The model also combines windowed and global attention with the addition of random attention. The random attention pattern is generated per input and in a way to attend to exactly r tokens in one row. Again, the intuition is that thanks to the pattern of the attention matrix, the omitted parts are reachable in just a few layers.



(a) Random attention (b) Window attention (c) Global Attention (d) BigBird

Figure 2.10: BigBird attention patterns with parameters $r = 2, w = 3, g = 2$, the last being the number of first tokens to attend to globally. Reprinted from (Zaheer et al., 2020).

The authors prove theoretical guarantees for the model, namely that the attention pattern is able to approximate an arbitrary attention matrix. However, in the

worst case, the number of layers needed is n, making the guarantee nonetheless rely on empirical evidence.

BigBird and Longformer both propose pattern-based attention, granting them $\mathcal{O}(n)$ complexity. However, they do introduce additional meta-parameters, which in order for the model to perform on BERT's level, have to be set to large values.

The paper also discusses techniques to make the computation of the patterns suitable for GPU, such as block-attention, where sparse tokens are "enlarged" to bigger blocks, and reshaping the final matrix to a rectangle shape.

Reformer

The Reformer model (Kitaev et al., 2020) offers two changes to the transformer model. The first is the usage of locality-sensitive hashing (LSH) (Andoni et al., 2015) in the attention mechanism, and the second is the use of reversible residual networks (RevNets) (Gomez et al., 2017).

The main idea behind the Reformer's attention is that softmax output from Equation 2.7 is influenced most by the largest values of QK^T. Since the output of QK^T represents the dot products between the rows of Q and K, the largest values are those, for which are the vectors from Q and K closest to each other. The problem of finding the closest vectors is still in $\mathcal{O}(n^2)$ since we need to compute the cosine distance between all the vector pairs – we would need to compute QK^T nonetheless. Here the authors utilize LSH for finding the closest vectors. Hash function $h(x)$ is "locality-sensitive" if it maps nearby vectors to the same hash with high probability. The simplified base idea is that under random linear projections into $n_{buckets}$ dimensional space (randomly initialized $n_{buckets} \times d$ matrix), nearby vectors preserve their orientation, thus sharing the dimension index where they show the highest values. Multiple random linear projections are performed in parallel to reduce the probability of similar vectors differing in the maximal indices. When the LSH computation is finished, each vector is assigned to a "bucket". Then the matrix Q is sorted so that vectors from the same buckets are in a sequence. What follows is the computation of full attention, but only on parts of the sorted matrix. The parts are selected so that each bucket attends to all of the vectors in the bucket and one previous bucket, see Figure 2.11. LSH does reduce attention computation time, but it introduces a large constant $c = n_{buckets}^2$ to the memory and time complexity since $n_{buckets}$ is typically 128. The authors also noted that setting $Q = K$ did not negatively impact the model's performance, and therefore LSH is applied only to the matrix Q.

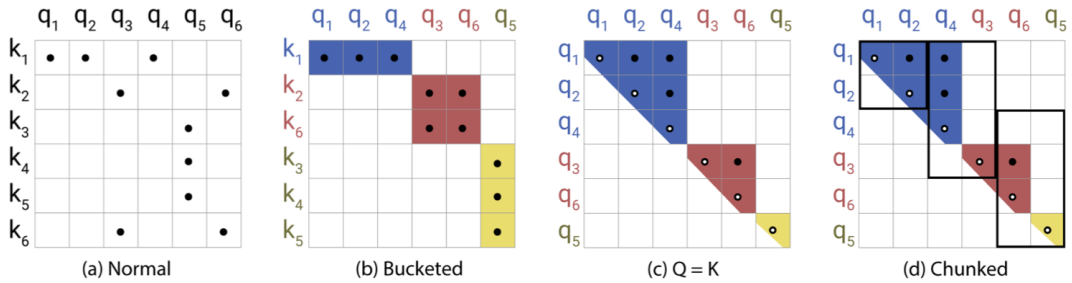


Figure 2.11: Reformer attention buckets. Reprinted from (Kitaev et al., 2020).

Another way to look at the LSH bucketing is to compute the dot product of the input vectors with a sequence of $\frac{n_{buckets}}{2}$ random vectors and their negative (oppositely-facing) counterparts, with the same dimension as the input vectors. Since cosine-similar

vectors produce the highest values of dot products, we can be confident that vectors are similar if they shared the highest dot product values with the same random vectors.

In order to reduce the memory complexity, the Reformer model applies three different methods. The first one is the use of RevNets (Gomez et al., 2017) discarding the need for storing layers' activations during training (Figure 2.12). The second one is the use of "chunking" – feeding only "chunks" of input at a time through the linear layers. The last one is the swapping of the parameters of the currently unused layer from GPU to CPU. This would not be efficient in normal transformers, but since Reformer is able to work with inputs even 64,000 tokens long, the authors claim that "the amount of compute done with the parameters amortizes the cost of their transfer."

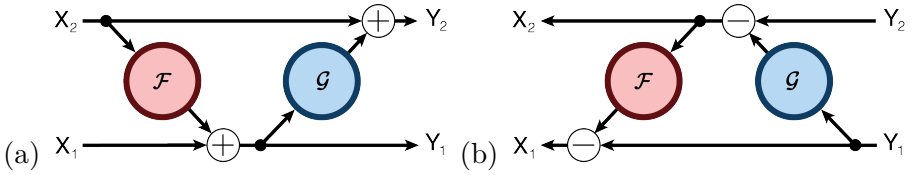


Figure 2.12: The forward **(a)** and the backward **(b)** computations scheme of a residual block. Reprinted from (Gomez et al., 2017).

Linformer

Linformer model (Wang et al., 2020) proposes a projection method to make the attention mechanism linear. The authors hypothesize that the softmax part of the attention is low-rank, and therefore it is possible to perform an approximate calculation in lower dimensions. The theory stands on the distributional Johnson–Lindenstrauss lemma (Johnson and Lindenstrauss, 1984), using its formulation from (Arriaga and Vempala, 2006). It states that given $k \leq n$, a random gaussian projection from an n-dimensional space into k-dimensional space with zero-mean and $\frac{1}{k}$ standard deviation, preserves vectors' lengths and pairwise dot products with high probability. They prove that for the softmax part, the projection dimension k can be set to $\min\{\Theta(9d\log(d)/\epsilon^2), 5\Theta(\log(n)/\epsilon^2)\}$ while keeping the approximation error reasonably low (Wang et al., 2020, Theorem 2).

The projection matrices E, F are defined in the proof as $E = \delta R, F = e^{-\delta}R$ for $R \in \mathbb{R}^{k\times n}$ with iid entries from $\mathcal{N}(0, \frac{1}{k})$ and a small constant δ. The attention mechanism itself is changed only by introducing non-trainable projection matrices when computing the keys and the values:

$$\text{LinAttn}(Q, K, V) = \text{Attention}(Q, EK, FV) = \text{softmax}\left(\frac{Q(EK)^T}{\sqrt{d}}\right)FV\ . \tag{2.8}$$

The resulting time and space complexity is $\mathcal{O}(kn)$.

The authors also tested multiple techniques of sharing the projection matrices, concluding that the model's results were preserved when all the attention heads and layers used the same E, F matrices, even when $E = F$. They also proposed using different projected dimensions k for different heads and layers, motivated by Figure 2.13 (right), although no tests were performed.

We want to note that (Wang et al., 2020, Theorem 1) provides no insight since it holds for arbitrary matrix P. However, the authors do provide anecdotal evidence of the attention mechanism's low rank (see Figure 2.13).

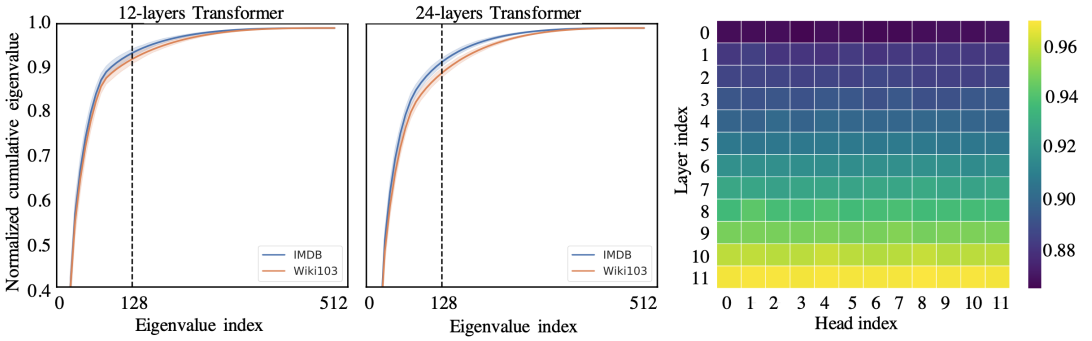


Figure 2.13: Anecdotal evidence of the low rank of a pre-trained transformer's attention mechanism. Higher cumulative sum of eigenvalues indicates the amount of information in the included indices. Reprinted from (Wang et al., 2020).

Performer

The Performer model (Choromanski et al., 2020) looks at the self-attention mechanism, specifically at the softmax substep $A = \exp(\frac{QK^T}{\sqrt{d}})$, as a randomized kernel function:

$$A_{i,j} = K(q_i, k_j) = \mathbb{E}[\phi(q_i)^T \phi(k_j)] \ , \tag{2.9}$$

where K is a kernel function $K\colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined for a feature map $\phi\colon \mathbb{R}^d \to \mathbb{R}^r$. The authors approximate the attention by using the randomized feature map to define matrices $Q', K' \in \mathbb{R}^{n \times r}$, with the rows equal to $\phi(q_i^T)^T$ and $\phi(k_i^T)^T$, respectively. Then, by calculating in the parenthesized order

$$AV = (Q'(K'V)) \tag{2.10}$$

the authors reduce the time complexity from $\mathcal{O}(n^2 d)$ to $\mathcal{O}(nrd)$. The paper then deals with finding suitable randomized feature maps with adequate theoretical assurances. The authors propose a general form

$$\phi(x) = \frac{h(x)}{\sqrt{m}}(f_1(\omega_1^T x), \ldots, f_1(\omega_m^T x), \ldots, f_l(\omega_1^T x), \ldots, f_l(\omega_m^T x)) \ , \tag{2.11}$$

where $f_1, \ldots, f_l\colon \mathbb{R} \to \mathbb{R}$, $g\colon \mathbb{R}^d \to \mathbb{R}$ and $\omega_1, \ldots, \omega_m \overset{\text{iid}}{\sim} \mathcal{D}$ for some distribution $\mathcal{D} \in \mathcal{P}(\mathbb{R})^d$.

The authors conclude their theoretical research by guaranteeing "unbiased or nearly-unbiased estimation of the attention matrix, uniform convergence and low estimation variance" for $h(x) = \exp(\frac{||x||}{2})$, $l = 2$, $f_1 = \cos$, $f_2 = \sin$, $\mathcal{D} = \mathcal{N}(0, \mathbf{I}_d)$, and ensuring that $\omega_1, \ldots, \omega_m$ are perpendicular to each other, (Choromanski et al., 2020, Theorem 4). The parameter m does depend on the L2-norm of the queries and keys, the dimensionality of the embeddings, and the required precision, but does not depend on the input sequence length n.

Nyströmformer

Nyströmformer's (Xiong et al., 2021) base idea is to approximate the attention computation by subsampling the original Q and K matrices and using a Nyström-based method (Wang and Zhang, 2013) to approximate the full attention. The subsampling is done by splitting the Q and K matrices into equally sized segments and then averaging each part into single vectors, calling them landmarks.

The attention computation can be written as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix} V \ , \tag{2.12}$$

where $A_S \in \mathbb{R}^{m \times m}$ and other matrices are of appropriate shapes, with m being the number of samples (landmarks). The authors note that using (Wang and Zhang, 2013), one could approximate the attention by expressing the matrix C_S using the other matrices as $C_S = F_S A_S^{\dagger} B_S$. The $\dagger$ sign indicates the Moore-Penrose inverse.

Such usage of the Nyström approximation will not help, as the samples can only be obtained after the attention matrix is computed – the softmax operation needs the whole matrix QK^T (or at least its rows). Nevertheless, the authors do compute softmax over only the subsampled parts of the QK^T matrix and use the resulting softmax-ed matrices to compute the approximated attention. The final approximated attention is obtained by

$$\widehat{\text{Attn}}(Q, K, V) = \left(\text{softmax}\left(\frac{Q\tilde{K}^T}{\sqrt{d}}\right) \text{softmax}\left(\frac{\tilde{Q}\tilde{K}^T}{\sqrt{d}}\right)^{\dagger}\right) \left(\text{softmax}\left(\frac{\tilde{Q}K^T}{\sqrt{d}}\right) V\right) \ , \tag{2.13}$$

where $\sim$ indicates a subsampled matrix. The parenthesization ensures that the computation remains efficient. The final memory and time complexity is $\mathcal{O}(n)$, provided $m \ll n$.

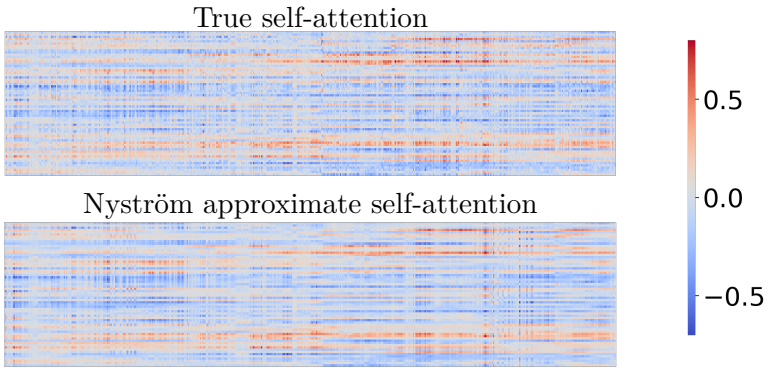


Figure 2.14: Anecdotal comparison of the true attention and Nyström approximation. Reprinted from (Xiong et al., 2021).

The only theoretical guarantee presented directly in the paper is that if the landmarks are equal to the original keys and queries, then the approximation converges to the true attention. However, this holds trivially. Proper theoretical guarantees are located in the appended repository[2]. There, the difference between true attention and the approximation is bounded by:

$$||\text{Attn} - \widehat{\text{Attn}}||_\infty \leq (1 + ||A_S^{\dagger}||_\infty + ||A_S^{\dagger} - Z^*||_\infty)||V||_\infty \ , \tag{2.14}$$

where $||\cdot||_\infty$ is the maximum absolute column sum, and $Z*$ is the result of the GPU-compatible iterative algorithm (Razavi et al., 2014) used by the authors to compute the Moore-Penrose inverse. The authors report good results on various tasks, even though the softmax operation is performed on submatrices only. The adverse effects might be reduced by the fact that landmarks are calculated as segment means, and therefore, the original information is present during the softmax step. Figure 2.14 provides an anecdotal comparison between full self-attention and the approximated attention.

[2] `https://github.com/mlpen/Nystromformer/`

The authors note that thanks to the theoretical results from (Oglic and Gärtner, 2017) and (Wang et al., 2020), even a small number of landmarks suffices to provide a good approximation. The first paper states that the Nyström approximation error decreases with the matrix's rank. In the second paper (Linformer), the authors suggest that the attention mechanism produces low-rank matrices.

3. Datasets

As mentioned, the primary dataset, usage-wise and inspiration-wise, was the FEVER dataset (Thorne et al., 2018a). Deriving from the authors' methods, we (Ullrich, 2021) were able to create our own FEVER-like database using Czech News Agency's infobank. In this chapter, we describe the FEVER dataset and the two datasets that our team has produced.

3.1 FEVER Dataset

The **F**act **E**xtraction and **Ver**ification (Thorne et al., 2018a) dataset is a large-scale dataset based on Wikipedia articles. The dataset was created by extracting sentences (claims) from the English Wikipedia articles and classifying them by human annotators as Supported, Refuted, or NotEnoughInfo. If the claim is verifiable (Supported, Refuted), then the evidence, either single or multiple paragraphs or even articles proving or disproving the claim, is also recorded. The complete FEVER contains 185,445 annotated claims generated using 50,000 popular articles.

The creation consisted of two steps. The sentences were first manually extracted from popular Wikipedia articles. Only the first paragraphs, usually containing the summary, were used for this step. Then, to create a more diverse set of claims, the annotators had the option of producing new claims by mutating the existing ones in various ways (generalizing, specification, entity substitution, non-trivial negating, and rephrasing). The negation used has to be non-trivial because simple negative words and phrases like "no" and "it is not true that" can be leveraged by the later steps of the fact-checking pipeline to immediately classify claims as Refuted instead of trying to understand the text.

More complex claims were created by providing related Wikipedia articles (articles hyperlinked from the original article) as another source of information while mutating the claim.

In the second step - annotation - the annotators were asked to label the generated claims and provide additional evidence when needed. The whole process was streamlined to not spend longer than five minutes on a single claim throughout all stages.

The quality of the dataset was thoroughly tested in the paper. We have, however, noticed an unaddressed issue described in Section 3.3.1. One of the methods for improving the dataset was annotating the generated claim by multiple people to reduce mislabeling.

3.1.1 FEVER CS

For use in baseline training and pre-training, we localized the original FEVER dataset into the Czech language. The localization consisted of translating the original claims using a translation service and mapping the English Wikipedia knowledge base used by FEVER to available Czech Wikipedia articles while removing claims with missing Czech evidence. The process is in-depth described by Ullrich (2021, Chapter 3).

Because of the large-scale usage of machine translation without the means for robust evaluation and the lack of one-to-one correspondence between Czech and English Wikipedia articles, this dataset serves only as a baseline to verify the robustness of our models on a larger scale, primarily for the document retrieval part of the pipeline.

Split	FEVER CS			FEVER EN		
	Supported	Refuted	NEI	Supported	Refuted	NEI
train	53,542	18,149	35,639	80,035	29,775	35,639
dev	3,333	3,333	3,333	6,666	6,666	6,666
test	3,333	3,333	3,333	6,666	6,666	6,666

Table 3.1: Label distribution of the resulting FEVER CS.


```
{
  "id": 120449,
  "verifiable": "VERIFIABLE",
  "label": "SUPPORTS",
  "claim": "Venuše se nazývá 'sesterskou planetou' Země.",
  "evidence": [
    [
      [
        141476,
        156671,
        "Venuše (planeta)",
        8,
        "Venus"
      ]
    ]
  ],
  "claim_en": "Venus is called the \"sister planet\" of Earth."
}
```


Figure 3.1: FEVER CS data example, containing one evidence set refering to the Venus Wikipedia page.

3.2 ČTK

The basis for the ČTK dataset is the collection of Czech news articles provided in collaboration with the Czech News Agency. Inspired by Thorne et al. (2018a) and Binau and Schulte (2020), our colleague Ullrich (2021) created a Czech version of the claim extracting and labeling software tool[1] running over the ČTK infobank's articles. It was designed to be used by layman annotators, who were students of our partner - the Faculty of Social Sciences at Charles University.

Split	Supported	Refuted	NotEnoughInfo
train	1132	519	473
dev	100	100	100
test	200	200	200

Table 3.2: Label distribution of the resulting ČTK v2.1 dataset.

The dataset creation consisted of two harvests. After reviewing the results of the first one, we were able to rewrite instructions in the tool to guide the annotators to create higher-quality claims and labels with fewer conflicts. The second harvest concluded with ≈ 3,500 labeled claims, with more than half being labeled two or more

[1] available at `https://fcheck.fel.cvut.cz/`

times (see Figure 3.4). As of this writing, this figure is not final as the dataset needs to be manually cleaned and have conflicts resolved.


```
{
  "id": 2143,
  "label": "REFUTES",
  "claim": "Kocianovo kvarteto nikdy nezískalo žádné ocenění.",
  "evidence": [
    [
      "T200706010229101_1"
    ]
  ],
  "source": "T200706010229101_1"
}
```


Figure 3.2: ČTK data example, containing one evidence set refering to the first paragraph of the ČTK infobank article with the id `T200706010229101`.

Paříž/Praha 2. června (ČTK) - Za vynikající kompletní nahrávku kvartetů Paula Hindemitha pro francouzskou firmu Harmonia Mundi obdrželo 3. června 1997 české Kocianovo kvarteto Velkou cenu francouzské Akademie Charlese Crose.

Figure 3.3: ČTK infobank entry corresponging to the evidence id in Figure 3.2.

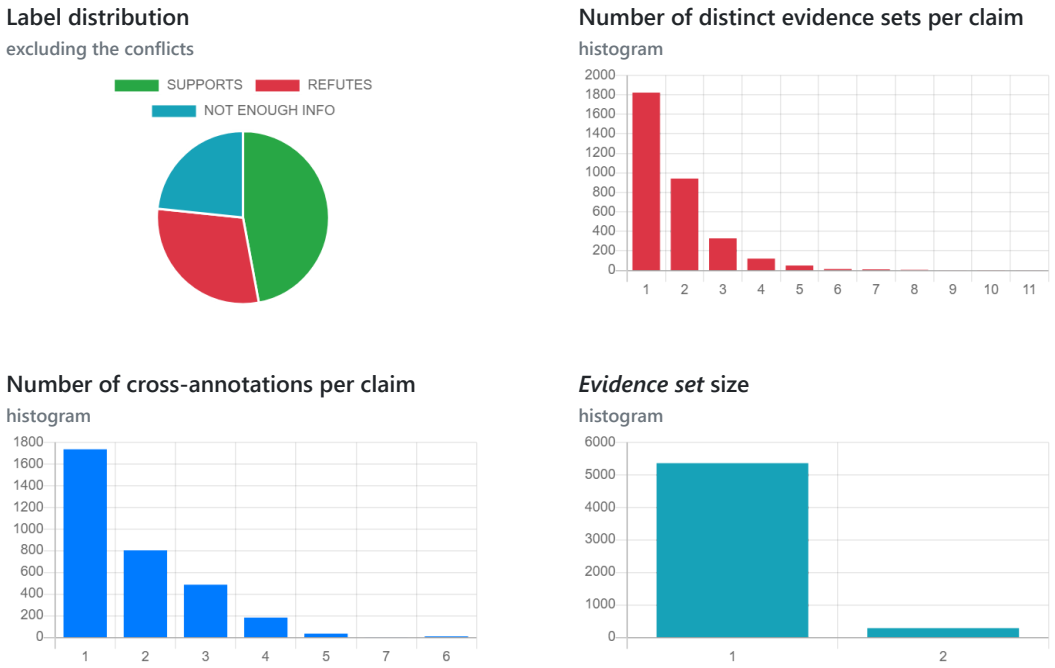


Figure 3.4: Visualizations of various properties of the collected ČTK dataset. Figure reprinted from (Ullrich, 2021).

3.3 Data Quality

The claim mutation in FEVER and ČTK datasets can be a source of unintentional cues, which the model can exploit to ”guess” the label without understanding the claim. For example, Thorne et al. (2018a) pointed out that their annotators had difficulties negating the claims beyond trivial negations.

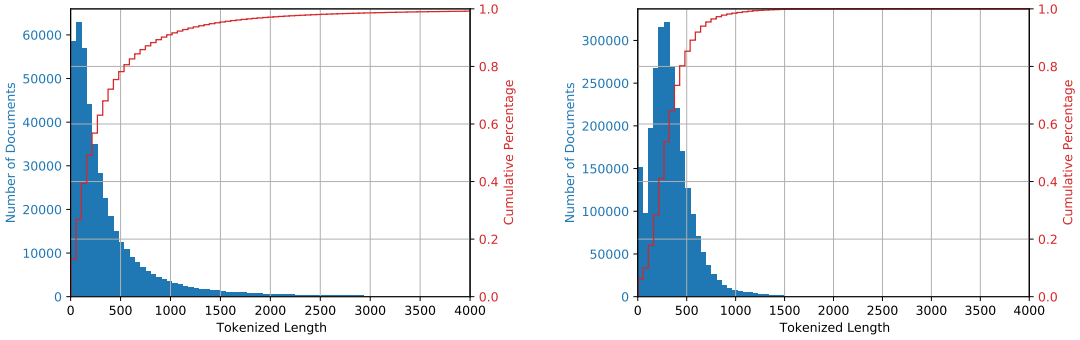
Rýpar (2021) conducted a series of evaluations on the ČTK and FEVER CS datasets using dataset-weighted cue information (DCI), and cue productivity and coverage (Niven and Kao, 2019), inspired by Derczynski et al. (2020). The results concluded that the original FEVER dataset contained simple negative clues, which were also present in FEVER CS, although with limited impact only. No significant bias was confirmed, but the current size of the ČTK dataset allows for thematic clusters which impact the dataset more than they should.

3.3.1 Data Leakage

As pointed out by Ullrich (2021), the original FEVER dataset contains an unaddressed quality – at least 80 % of verifiable dev claims share an evidence-set document with some train claim. The effects of this ”leakage” are unknown and call for further research.

3.4 Knowledge Base Data Length

The above-discussed datasets are built upon two distinct knowledge bases, namely ČTK infobank and Czech Wikipedia. Note that ČTK infobank has been filtered from economic tables and sports news. The focus of this thesis is to explore the benefits of models supporting long inputs. Figure 3.5 depicts the resulting distribution of tokenized (using RobeCzech (Straka et al., 2021)) lengths of each of the knowledge bases:



(a) Czech Wikipedia articles.

(b) Filtered ČTK articles.

Figure 3.5: Histograms of the tokenized lengths of whole articles in both Czech Wikipedia and ČTK infobank.

The histograms show that the standard input length of 512 tokens forces us to truncate ≈ 20% of Wikipedia articles. For the ČTK dataset, the evidence is paragraph based and therefore, the truncation affects only the overall context of the article. Given the nature of the long documents in each knowledge base, namely:

- the first paragraph of the Wikipedia articles is always the summary,
- the most informative paragraphs of ČTK news articles are typically self-contained,
- the long articles in ČTK infobank are either draft program statements of the government or news summary already contained in other articles,

we can expect that the effect of extending the input might be marginal. Nonetheless, non-trivial information may still be truncated, and thus by leveraging the lower time complexity of the models discussed in Section 2.4.8, we may produce positive results.

4. Proposed Solutions

4.1 Baselines

As the traditional baseline, we have decided to use BM25, as proposed by Yang et al. (2019). We use the Pyserini[1] python toolkit for information retrieval (Lin et al., 2021) – a python interface for the Anserini[2] library (Yang et al., 2017, 2018), originally implemented in Java, building on the Lucene[3] library. The Pyserini library is a highly optimized library implementing multiple search algorithms. In Pyserini, BM25's implemention uses bigrams features.

We also compare our results with the results achieved by Rýpar (2021), who researched document retrieval for documents up to one paragraph long ($\approx$ 230 tokens), using mBERT and ColBERT (Khattab and Zaharia, 2020) models.

4.2 Neural Models

From the models described in Section 2.4.8 only Longformer (Beltagy et al., 2020), BigBird (Zaheer et al., 2020) and Reformer (Kitaev et al., 2020) are available in the HuggingFace Transformers library (Wolf et al., 2020), and none are trained in the Czech language.

We have chosen to experiment with BigBird and Nyströmformer. We chose BigBird over Longformer, as the BigBird architecture essentially performs the same way as Longformer. Another reason is that during experimentation with Longformer, we ran into scalability issues with embedding the documents. The time required to generate the representation of the entire Czech Wikipedia was estimated to take $\approx$ 300 hours (for the ČTK infobank $\approx$ 1500 hours), despite using the authors' guide[4] and implementation.

The Performer (Choromanski et al., 2020), Linformer (Wang et al., 2020), and Reformer (Kitaev et al., 2020) models, although promising, were shown to be outperformed by the Nyströmformer model by its authors (see Appendix 2). Given the resource intensity of the task[5] and the lack of a unified code base, we have decided to follow the results and implement the Nyströmformer model only.

We tested three approaches to the task:

1. Train the BigBird model using multilingual knowledge distillation as described in Section 2.4.6 using the SBERT library (Reimers and Gurevych, 2019).

2. Train Nyströmformer on MLM task from RobeCzech (Straka et al., 2021) checkpoint using the provided code base[6].

3. As a soft baseline, evaluate the pre-trained RobeCzech model and SBERT-fine-tuned (see Figure 2.7) RobeCzech model on the document retrieval task.

Following the discussion in Section 3.4, we have chosen the input length of the long models to be 2048.

[1] `https://github.com/castorini/pyserini/`

[2] `https://github.com/castorini/Anserini`

[3] `https://lucene.apache.org/`

[4] `https://github.com/allenai/longformer/blob/caefee668e39cacdece7dd603a0bebf24df6d8ca/scripts/convert_model_to_long.ipynb`

[5] linear slow-down in an ideal scenario means that for sequences of length 2048 or 4096 tokens the model nevertheless performs $\approx$ 8 to 16 times slower

[6] `https://github.com/mlpen/Nystromformer/`

4.3 Evaluation Metrics

Document retrieval is best evaluated by the model's ability to match the evidence set with the retrieved documents. By the retrieved documents, we mean the sequence of k documents from the knowledge base with the highest scores. We can match the evidence set in a variety of ways, motivating multiple measures of performance:

- **Precision** - We try to "fit within" the evidence set with retrieved documents.
- **Recall** - We try to "cover" the evidence set.
- **F1** - The harmonic mean of precision and recall.
- **Mean Reciprocal Rank (MRR)** (Voorhees, 2000) - We try to retrieve the relevant documents at the first positions.

The specific formulas are:

$$\text{precision} = \frac{|\{\text{evidence documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} , \tag{4.1}$$

$$\text{recall} = \frac{|\{\text{evidence documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{evidence documents}\}|} , \tag{4.2}$$

$$\text{F1} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} , \tag{4.3}$$

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} , \tag{4.4}$$

where Q is a sequence of queries and rank_i is the rank of the first relevant document retrieved for the query q_i.

The most relevant metrics for us will be the recall and the MRR since we aim to provide all the necessary knowledge ideally among the first results. Since the precision and the recall depend on the number k of retrieved documents, we evaluate the models with varying k and denote each evaluation with the suffix $@k$.

5. Experiments

5.1 BM25

We used grid search for $k_1 \in [0.2, 2.0]$, $b \in [0.1, 1.0]$ with the step size of 0.1, over the training splits. Only the entries labeled supported and refuted were used since only such entries provide evidence. We performed fine-tuning over whole training splits for both ČTK and FEVER CS datasets. The resulting parameters were chosen according to the best recall achieved on the training data, calculated by summing up the difference between the best-achieved recall@k and the configuration's recall@k. The monitored levels of the parameter k were 1, 5, 10, 25, 50, 100, and 500. The resulting values of the hyperparameters are displayed in Table 5.1. We note that the difference between the selected values and neighboring values was insignificant (see Table 5.2).

datset	k_1	b
ČTK	1.0	0.5
FEVER CS	1.4	0.3

Table 5.1: Fine-tuned parameters for BM25.

datset	k_1	b
ČTK	[0.9, 1.1]	[0.4, 0.5]
FEVER CS	[1.3, 1.5]	[0.3, 0.4]

Table 5.2: Neighborhoods of parameter values with the best performance.

5.2 ColBERT and mBERT baselines

We used the ColBERT and mBERT models pre-trained and fine-tuned by Rýpar (2021). Rýpar (2021) performed document retrieval on a paragraph level, and therefore to better compare the model's quality in the context of this thesis, we modify the predicted sequences by substituting paragraphs with the document they originate from. The created duplicates in the predictions are omitted in a forward pass, starting from the top and deleting each article we have already encountered. The change in the resulting metric, however, turned out to be marginal.

5.3 Nyströmformer

For Nyströmformer, we used the compact containerized implementation published by the authors[1]. The training was done using the MLM task on Czech data, namely Czech Wikipedia[2], a very large Czech corpus "czes" czes (2011), and the Corpus of contemporary written (printed) Czech "SYN v4" (Křen et al., 2016). The reason for such a large amount of data is that when using inputs of length 2048 tokens, we obtain a relatively small number of instances, and the fact that large corpora tend to produce better-performing models (Conneau et al., 2019).

The model was not trained from scratch; instead, we used the pre-trained RobeCzech model as the starting point. RobeCzech's tokenizer was used, and all the weights, except for the attention mechanism, were initialized as a copy of RobeCzech's weights. However, we had to address an issue regarding RobeCzech's tokenizer, where the size of the word in its vocabulary was greater than the number of possible tokens. This initialization proved to increase the speed of training. The issue occurred as the vocabulary mapped multiple tokens to `[UNK]`, a token representing unknown values.

[1] `https://github.com/mlpen/Nystromformer/`
[2] `https://dumps.wikimedia.org/`

Using a simple wrapper around the tokenizer, which returned the correct vocabulary size, solved the issue.

The training itself was performed for 3000 steps on four Tesla V100-SXM2-32GB graphics cards, with batch size 8, embedding dimension 768, landmark count 512, and a decaying learning rate 8e-5. Figure 5.1 shows that for Nyströmformer, the training does not significantly slow down after increasing the input length.

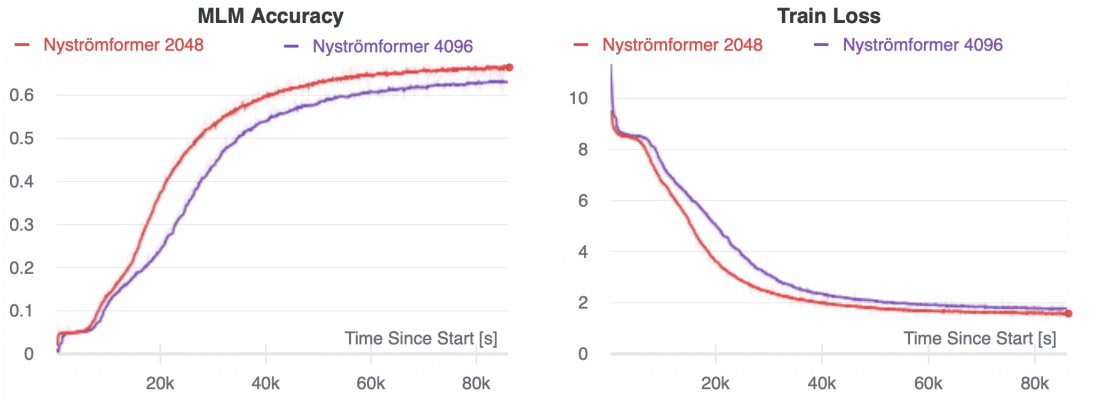


Figure 5.1: Comparison of the speed of training for two different input lengths. The experimental setup is the same except for doubled batch size and 1.6 times higher learning rate for the shorter-input model.

After training, we applied mean-pooling over generated token embeddings to generate the final representations.

5.4 BigBird

We have used the pre-trained BigBird model along with its implementation from the HuggingFace Transformers library (Wolf et al., 2020). We applied multilingual knowledge distillation as described in Section 2.4.6 using the SBERT library (Reimers and Gurevych, 2019). Both the teacher and the student model were initialized as the pre-trained BigBird model. The training was performed using sentence-parallel Czech-English corpus Czeng 2.0 (Kocmi et al., 2020). The corpus contains human-translated sentences in both the Czech and the English languages. After training for 25 mini-epochs on a single V100 GPU, with a learning rate of 2e-5, and a batch size of 16, the evaluation error stopped decreasing.

The second stage was the training of representation generation, described in Section 2.4.4, again using SBERT. The model was tasked with generating similar or dissimilar representations of sentences in the Stanford Natural Language Inference (SNLI) Corpus (Bowman et al., 2015b), depending on the label. The dataset consists of pairs of sentences labeled `entailment`, `contradiction`, and `neutral`. The training was performed for 30 epochs.

We theorize that it would be beneficial to train the model on longer inputs than single sentences since BigBird, with its long input capability, cannot train the weights corresponding to the farther parts of the input. Chaining multiple sentences together from the Czeng 2.0 dataset would not be helpful, as the sentences rarely form a large block of coherent text. We would thus need a large parallel dataset with whole works translated. We are, however, not aware of such a Czech-English dataset.

5.5 RobeCzech Baseline

As a simple baseline we decided to apply the above mentioned method of "entailment training" on the SNLI dataset to the RobeCzech model. We performed the exact same training as the second stage of BigBird training.

5.6 Results

model	P@1	P@2	P@5	P@10	P@20	P@500
BM25	24.33	15.57	8.03	4.79	**2.85**	0.18
mBERT	9.80	7.41	4.07	2.56	1.63	0.18
ColBERT	**24.37**	**17.09**	**8.69**	**4.87**	2.76	**0.19**
RobeCzech	0.97	0.73	0.63	0.39	0.28	0.05
RobeCzech 50ep (SNLI)	5.84	5.47	3.60	2.38	1.53	0.12
RobeCzech 80ep (SNLI)	5.84	5.23	3.21	2.19	1.40	0.12
BigBird (SNLI)	0.97	0.61	0.54	0.29	0.23	0.03
	R@1	R@2	R@5	R@10	R@20	R@500
BM25	24.15	29.51	38.54	45.61	54.15	**80.00**
mBERT	9.82	14.86	20.40	25.69	32.75	72.80
ColBERT	**24.43**	**34.26**	**43.58**	**48.87**	**55.42**	79.35
RobeCzech	0.98	1.46	3.17	3.90	5.61	24.63
RobeCzech 50ep (SNLI)	5.61	10.24	15.85	21.22	27.80	55.61
RobeCzech 80ep (SNLI)	5.85	10.24	14.63	19.51	25.37	54.88
BigBird (SNLI)	0.98	1.22	2.68	2.93	4.63	13.41
	F@1	F@2	F@5	F@10	F@20	F@500
BM25	24.24	20.39	13.29	8.67	**5.41**	0.36
mBERT	9.81	9.89	6.79	4.66	3.11	0.36
ColBERT	**24.40**	**22.80**	**14.50**	**8.86**	5.27	**0.38**
RobeCzech	0.97	0.97	1.05	0.71	0.53	0.11
RobeCzech 50ep (SNLI)	5.72	7.14	5.87	4.29	2.91	0.25
RobeCzech 80ep (SNLI)	5.85	6.93	5.27	3.94	2.65	0.24
BigBird (SNLI)	0.97	0.81	0.89	0.53	0.44	0.06
	M@1	M@2	M@5	M@10	M@20	M@500
BM25	21.22	23.63	25.82	26.70	27.23	27.60
mBERT	11.02	13.67	15.18	15.78	16.28	16.82
ColBERT	**23.31**	**28.07**	**30.35**	**31.17**	**31.61**	**32.04**
RobeCzech	1.13	1.53	1.95	2.01	2.15	2.32
RobeCzech 50ep (SNLI)	5.79	7.88	9.57	10.28	10.72	11.10
RobeCzech 80ep (SNLI)	5.79	8.36	9.73	10.42	10.75	11.13
BigBird (SNLI)	1.29	1.45	1.77	1.78	1.93	2.01

Table 5.3: Models' performace on the ČTK dataset.

5.7 Discussion

The selected methods were unsuccessful in surpassing the baselines. The best performance out of the proposed methods has been achieved by the RobeCzech model, which

model	P@1	P@2	P@5	P@10	P@20	P@500
BM25	31.13	20.36	10.41	5.80	3.09	0.14
mBERT	63.88	**40.59**	**19.09**	**10.21**	**5.31**	**0.23**
ColBERT	**67.25**	38.74	17.41	9.16	4.76	0.21
RobeCzech	0.42	0.34	0.27	0.23	0.16	0.04
RobeCzech 50ep (SNLI)	26.67	17.33	8.90	5.10	2.84	0.16
RobeCzech 80ep (SNLI)	23.90	15.98	8.30	4.78	2.68	0.16
BigBird (SNLI)	5.45	3.97	2.38	1.54	0.98	0.08
Nyströmformer 2048 2k Steps	0.32	0.26	0.16	0.13	0.11	0.03
Nyströmformer 2048 3k Steps	0.21	0.15	0.11	0.08	0.07	0.02
	R@1	R@2	R@5	R@10	R@20	R@500
BM25	29.09	37.55	47.70	52.64	55.91	63.52
mBERT	59.60	**75.28**	**87.55**	**92.84**	**95.47**	**98.89**
ColBERT	**62.59**	71.81	80.41	84.02	86.95	94.21
RobeCzech	0.35	0.57	1.19	1.97	2.79	15.41
RobeCzech 50ep (SNLI)	24.47	31.85	40.73	46.80	51.98	71.29
RobeCzech 80ep (SNLI)	22.02	29.28	37.98	43.62	48.66	70.57
BigBird (SNLI)	5.01	7.26	10.73	13.80	17.58	33.98
Nyströmformer 2048 2k Steps	0.24	0.41	0.66	1.10	1.89	11.61
Nyströmformer 2048 3k Steps	0.18	0.27	0.44	0.68	1.13	7.53
	F@1	F@2	F@5	F@10	F@20	F@500
BM25	30.07	26.41	17.09	10.44	5.86	0.29
mBERT	61.66	**52.74**	**31.34**	**18.40**	**10.06**	**0.45**
ColBERT	**64.84**	50.33	28.62	16.52	9.03	0.42
RobeCzech	0.38	0.42	0.44	0.40	0.30	0.07
RobeCzech 50ep (SNLI)	25.52	22.45	14.61	9.20	5.39	0.31
RobeCzech 80ep (SNLI)	22.92	20.67	13.63	8.62	5.07	0.31
BigBird (SNLI)	5.22	5.13	3.89	2.77	1.86	0.15
Nyströmformer 2048 2k Steps	0.27	0.32	0.26	0.23	0.20	0.05
Nyströmformer 2048 3k Steps	0.19	0.19	0.17	0.14	0.13	0.03
	M@1	M@2	M@5	M@10	M@20	M@500
BM25	29.92	34.30	37.23	37.99	38.22	38.37
mBERT	56.18	63.87	67.68	68.48	68.75	68.84
ColBERT	**63.93**	**68.41**	**70.70**	**71.16**	**71.34**	**71.47**
RobeCzech	0.32	0.39	0.49	0.59	0.63	0.74
RobeCzech 50ep (SNLI)	26.59	30.17	32.67	33.52	33.80	34.08
RobeCzech 80ep (SNLI)	24.26	28.02	30.52	31.22	31.57	31.85
BigBird (SNLI)	4.71	5.84	6.78	7.18	7.46	7.68
Nyströmformer 2048 2k Steps	0.19	0.24	0.29	0.36	0.40	0.48
Nyströmformer 2048 3k Steps	0.18	0.20	0.25	0.26	0.29	0.34

Table 5.4: Models' performance on the FEVER CS dataset.

underwent 50 episodes of representation generation fine-tuning using the machine-translated SNLI dataset. The extended training presumably caused a forgetting of important pre-trained patterns, and therefore, the 80 episode model performed slightly worse.

We observe the potential of "entailment training" using the translated SNLI dataset.

The base version of RobeCzech also significantly improved its performance after a few epochs of the SNLI training. Even the originally English model BigBird was able to outperform the Czech monolingual model Nyströmformer and provide at least some relevant documents.

The Nyströmformer model failed, despite producing a high accuracy on the MLM pre-training task. All of this leads us to agree with the results of Reimers and Gurevych (2019), who state that the crucial part of generating representation is to fine-tune the model to do so on SNLI-like datasets.

Differences between datasets are also visible. On the FEVER CS dataset, the paragraph-level-trained neural approaches outperformed every other model, as well as the traditional baseline BM25. The best version of RobeCzech was able to perform on-par with BM25, while the other tested approaches, namely BigBird and Nyströmformer, failed.

On the other hand, the ČTK dataset posed a significant challenge to the neural models. The traditional baseline BM25 performs on the same level as the best-performing neural model ColBERT.

These observations suggest that neural models struggle with comprehending ČTK news articles, apart from effectively emulating simple word-count-based representations (like TFIDF). The semi-structured style of Wikipedia articles, and the consistent explanatory nature of the articles, possibly allows the neural models to "focus" on the word meaning without being "distracted" by the changing style. News articles contain considerably more noise, and the sentiment of different articles also varies.

Conclusion

We introduced the problem of fact-checking and described modern approaches to the problem. We explained the motivation for studying the topic and proposed using new neural models to help with automatic fact-checking. Our research team created the first fact-checking dataset in the Czech language (Ullrich, 2021) and explored different models' architectures capable of good performance. Inspired by the FEVER pipeline (Thorne et al., 2018a) (see Figure 1.1), we split the problem into two distinct tasks – document retrieval and natural language inference.

This thesis further dealt with document retrieval. We first defined the task formally and then introduced well-established traditional techniques for the task. Then followed a brief description of the progress from RNN-based language models to transformers (Vaswani et al., 2017) and BERT (Devlin et al., 2018). Since the BERT model as described in (Devlin et al., 2018) and as adopted across the research field performs best for inputs up to 512 tokens, we were forced to work over paragraphs instead of the whole documents. Our colleagues (Rýpar, 2021) and (Dědková, 2021) have studied this type of document retrieval.

In this thesis, we explored whether we could improve retrieval performance by utilizing whole articles. We provided a summary of currently available papers regarding transformer language models supporting long inputs, namely Longformer, BigBird, Reformer, Linformer, Performer, and Nyströmformer. Since no pre-trained models for the Czech language were available, we either trained them from scratch or utilized the student-teacher method (Reimers and Gurevych, 2020) described in Section 2.4.6. Lastly, we compared the traditional, short-input, and long-input approaches in the document retrieval task and analyzed the results.

The explored models turned out not to outperform the traditional and the paragraph-level retrieval baselines. However, we highlighted the importance of SBERT-like fine-tuning and displayed its usefulness even for originally monolingual English models.

The main dataset of this work, ČTK dataset, and the ČTK infobank still pose a significant challenge to the document retrieval task, with the best models performing on-par with a simple, accurate-based baseline.

Future Work

As time progresses, automatic fact-checking will be needed more and more. With the joint effort across the machine learning research field, we hope to train better NLP neural models focusing on unstructured and news-like data. We wish to continue improving the created ČTK dataset and to now focus on the natural language inference task of the fact-checking pipeline.

Regarding long-input models, we would like to focus on applying representation fine-tuning to the models, as well as exploring the ICT pre-training task as suggested by Chang et al. (2020). Neural language models supporting long inputs are an exciting developing area of machine learning, and in the future, they might provide a significant benefit in fast searching through large databases of documents.

Bibliography

J Alammar. The illustrated transformer, 2018. URL https://jalammar.github.io/illustrated-transformer/. [Blog post; accessed 07-28-2021].

Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya P. Razenshteyn, and Ludwig Schmidt. Practical and optimal LSH for angular distance. *CoRR*, abs/1509.02897, 2015. URL http://arxiv.org/abs/1509.02897.

Mikhail Arkhipov, Maria Trofimova, Yuri Kuratov, and Alexey Sorokin. Tuning multilingual transformers for language-specific named entity recognition. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 89–93, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-3712. URL https://aclanthology.org/W19-3712.

Rosa I. Arriaga and Santosh Vempala. An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning*, 63(2):161–182, May 2006. ISSN 1573-0565. doi: 10.1007/s10994-006-6265-7. URL https://doi.org/10.1007/s10994-006-6265-7.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries*, 17(4):305–338, November 2016. ISSN 1432-1300. doi: 10.1007/s00799-015-0156-0. URL https://doi.org/10.1007/s00799-015-0156-0.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020. URL https://arxiv.org/abs/2004.05150.

Julie Binau and Henri Schulte. Danish fact verification: An end-to-end machine learning system for automatic fact-checking of danish textual claims. 2020.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326, 2015a. URL http://arxiv.org/abs/1508.05326.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015b.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL https://arxiv.org/abs/2005.14165.

Cristian Bucila, R. Caruana, and Alexandru Niculescu-Mizil. Model compression. In *KDD '06*, 2006.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. Pre-training tasks for embedding-based large-scale retrieval. *CoRR*, abs/2002.03932, 2020. URL https://arxiv.org/abs/2002.03932.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051, 2017. URL http://arxiv.org/abs/1704.00051.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. *CoRR*, abs/2009.14794, 2020. URL https://arxiv.org/abs/2009.14794.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019. URL http://arxiv.org/abs/1911.02116.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. Overview of the TREC 2020 deep learning track. *CoRR*, abs/2102.07662, 2021. URL https://arxiv.org/abs/2102.07662.

czes. czes, 2011. URL http://hdl.handle.net/11858/00-097C-0000-0001-CCCF-C. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Leon Derczynski, Julie Binau, and Henri Schulte. Maintaining quality in FEVER annotation. In *Proceedings of the Third Workshop on Fact Extraction and VERification (FEVER)*, pages 42–46, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.fever-1.6. URL https://aclanthology.org/2020.fever-1.6.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Barbora Dědková. Multi-stage methods for document retrieval in the czech language, 2021.

Aidan N. Gomez, Mengye Ren, Raquel Urtasun, and Roger B. Grosse. The reversible residual network: Backpropagation without storing activations. *CoRR*, abs/1707.04585, 2017. URL http://arxiv.org/abs/1707.04585.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2021. doi: 10.1109/TBDATA.2019.2921572.

William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26, 1984.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. *CoRR*, abs/2004.12832, 2020. URL https://arxiv.org/abs/2004.12832.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *CoRR*, abs/2001.04451, 2020. URL https://arxiv.org/abs/2001.04451.

Tom Kocmi, Martin Popel, and Ondrej Bojar. Announcing czeng 2.0 parallel corpus with over 2 gigawords. *arXiv preprint arXiv:2007.03006*, 2020.

Michal Křen, Václav Cvrček, Tomáš Čapka, Anna Čermáková, Milena Hnátková, Lucie Chlumská, Tomáš Jelínek, Dominika Kováříková, Vladimír Petkevič, Pavel Procházka, Hana Skoumalová, Michal Škrabal, Petr Truneček, Pavel Vondřička, and Adrian Zasina. SYN v4: large corpus of written czech, 2016. URL http://hdl.handle.net/11234/1-1846. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291, 2019. URL http://arxiv.org/abs/1901.07291.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. Pyserini: An easy-to-use python toolkit to support replicable IR research with sparse and dense representations. *CoRR*, abs/2102.10073, 2021. URL https://arxiv.org/abs/2102.10073.

Kateřina Macková and Milan Straka. Reading comprehension in czech via machine translation and cross-lingual transfer. In Petr Sojka, Ivan Kopeček, Karel Pala, and Aleš Horák, editors, *Text, Speech, and Dialogue*, pages 171–179, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58323-1.

Timothy Niven and Hung-Yu Kao. Probing neural network comprehension of natural language arguments. *CoRR*, abs/1907.07355, 2019. URL http://arxiv.org/abs/1907.07355.

Dino Oglic and Thomas Gärtner. Nyström method with kernel k-means++ samples as landmarks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2652–2660. PMLR, 2017. URL http://proceedings.mlr.press/v70/oglic17a.html.

Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. Understanding the exploding gradient problem. *CoRR*, abs/1211.5063, 2012. URL http://arxiv.org/abs/1211.5063.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Pavel Přibáň, Tomáš Hercig, and Josef Steinberger. Machine learning approach to fact-checking in West Slavic languages. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 973–979,

Varna, Bulgaria, September 2019. INCOMA Ltd. doi: 10.26615/978-954-452-056-4_113. URL https://aclanthology.org/R19-1113.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016. URL http://arxiv.org/abs/1606.05250.

M. Kafaei Razavi, A. Kerayechian, M. Gachpazan, and S. Shateyi. A new iterative method for finding approximate inverses of complex matrices. *Abstract and Applied Analysis*, 2014:563787, Sep 2014. ISSN 1085-3375. doi: 10.1155/2014/563787. URL https://doi.org/10.1155/2014/563787.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084, 2019. URL http://arxiv.org/abs/1908.10084.

Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. *CoRR*, abs/2004.09813, 2020. URL https://arxiv.org/abs/2004.09813.

Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126. Gaithersburg, MD: NIST, January 1995. URL https://www.microsoft.com/en-us/research/publication/okapi-at-trec-3/.

Martin Rýpar. Methods of document retrieval for fact-checking, 2021.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050, 2003. URL http://arxiv.org/abs/cs/0306050.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

Jakub Sido, Ondrej Prazák, Pavel Pribán, Jan Pasek, Michal Seják, and Miloslav Konopík. Czert - czech bert-like model for language representation. *CoRR*, abs/2103.13031, 2021. URL https://arxiv.org/abs/2103.13031.

Milan Straka, Jakub Náplava, Jana Straková, and David Samuel. Robeczech: Czech roberta, a monolingual contextualized language representation model. *CoRR*, abs/2105.11314, 2021. URL https://arxiv.org/abs/2105.11314.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL http://arxiv.org/abs/1409.3215.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and verification. *CoRR*, abs/1803.05355, 2018a. URL http://arxiv.org/abs/1803.05355.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. The fact extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages

1–9, Brussels, Belgium, November 2018b. Association for Computational Linguistics. doi: 10.18653/v1/W18-5501. URL https://aclanthology.org/W18-5501.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. The FEVER2.0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, 2018c.

Herbert Ullrich. Dataset for automated fact checking in czech language, 2021. URL http://bertik.net/dp_final.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Ellen Voorhees. The eighth text retrieval conference (trec-8), November 2000. URL https://doi.org/10.6028/NIST.SP.500-246.

Shusen Wang and Zhihua Zhang. On the lower bounds of the nyström method. *CoRR*, abs/1303.4207, 2013. URL http://arxiv.org/abs/1303.4207.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020. URL https://arxiv.org/abs/2006.04768.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*, abs/1704.05426, 2017. URL http://arxiv.org/abs/1704.05426.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. *CoRR*, abs/2102.03902, 2021. URL https://arxiv.org/abs/2102.03902.

Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 1253–1256, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350228. doi: 10.1145/3077136.3080721. URL https://doi.org/10.1145/3077136.3080721.

Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Reproducible ranking baselines using lucene. *J. Data and Information Quality*, 10(4), October 2018. ISSN 1936-1955. doi: 10.1145/3239571. URL https://doi.org/10.1145/3239571.

Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. Critically examining the "neural hype": Weak baselines and the additivity of effectiveness gains from neural ranking models. *CoRR*, abs/1904.09171, 2019. URL http://arxiv.org/abs/1904.09171.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062, 2020. URL `https://arxiv.org/abs/2007.14062`.

Appendix A - Acronyms

BERT Bidirectional Encoder Representations from Transformers

FEVER Fact Extraction and Verification

GPT-3 Generative Pre-trained Transformer 3

ČVUT České Vysoké Učení Technické (Czech Technical University in Prague)

ČTK Česká Tisková Kancelář (Czech News Agency)

NEI Not Enough Info

TREC Text Retrieval Conference

NNLM Neural Networks

TFIDF Term Frequency - Inverse Document Frequency

BM25 Best Match 25

TF Term Frequency

IDF Inverse Document Frequency

BOW Bag of Words

RNN Recurent Neural Networks

Seq2Seq Sequence to Sequence

SOTA State of The Art

NER Named Entity Recognition

NLI Natural Language Inference

QA Question Answering

ICT Inverse Cloze Task

BFS Body First Selection

WLP Wiki Link Prediction

FAISS Facebook AI Similarity Search

GloVe Global vectors for word representation

SBERT Sentence-BERT

RoBERTa Robustly optimized BERT

XLM-R XL Multilingual RoBERTa

SGD Stochastic Gradient Descend

LSH Locality-sensitive Hashing

CNN Convolutional Neural Networks

RevNet Reversible Residual Networks

iid Independent and Identically Distributed

DCI Dataset-weighted Cue Information

mBERT multilingual BERT

SNLI The Stanford Natural Language Inference (SNLI) Corpus

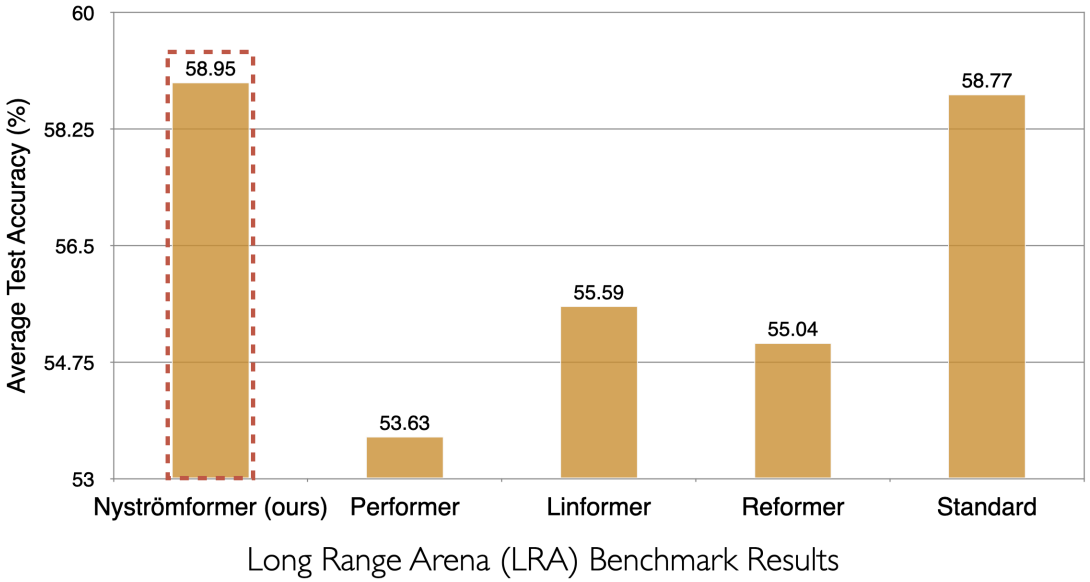
MRR Mean Reciprocal Rank

Appendix B - LRA Comparison

The authors of the Nyströmformer model (Xiong et al., 2021) published the following comparision between the Nyströmformer, Performer (Choromanski et al., 2020), Linformer (Wang et al., 2020), and Reformer (Kitaev et al., 2020) models:

Model	ListOps (2K)	Text (4K)	Retrieval (4K)	Image (1K)	Pathfinder (1K)	Avg
Standard	37.10	65.02	79.35	38.20	74.16	58.77
Reformer	19.05	64.88	78.64	43.29	69.36	55.04
Linformer	37.25	55.91	79.37	37.84	67.60	55.59
Performer	18.80	63.81	78.62	37.07	69.87	53.63
Nyströmformer (ours)	37.15	65.52	79.56	41.58	70.94	**58.95**

Table 1: Experimental results on Long Range Arena (LRA) benchmark. We report accuracy for each individual task and average accuracy across all tasks. Our Nyströmformer performs competitively with standard self-attention, outperforming Reformer, Linformer, and Performer. While we achieved consistent results reported in (Tay et al. 2020) for most tasks in our PyTorch reimplementation, the performance on Retrieval task is higher for all models even we tried different hyperparameters.



The upper figure reprinted from (Xiong et al., 2021), the bottom figure reprinted from the attached repository[3].

Appendix C - Code Repository

The code used in this thesis is avalilable at `https://gitlab.fel.cvut.cz/factchecking/master-thesis-code`. It contains multiple ad hoc jupyter notebooks, which were comfortable to work with using remote access to the cluster. Distinct subtasks reside in their own respective directories.