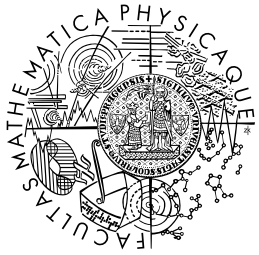


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

**MASTER THESIS**

Bc. Kateřina Macková

# Question Answering

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: RNDr. Milan Straka, PhD.

Study programme: Computer Science

Study branch: Artificial Intelligence

Prague 2019

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date .....  .....

Author's signature

I would like to thank my supervisor Milan Straka for his oversight and guidance for all the time.

Title: Question Answering

Author: Bc. Kateřina Macková

Department: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Milan Straka, PhD., Institute of Formal and Applied Linguistics

Abstract: Question answering is a computer science discipline in the field of natural language processing and information retrieval. The goal is to build a system that can automatically find an answer to a certain question in the text. Nowadays, there exist a lot of models trained on huge training data sets in English. This work focuses on building similar models in Czech without having any Czech training datasets. In this work, we have used SQuAD 1.1 and translated it to Czech to create training and development datasets. Then, we have trained and tested BiDirectional Attention Flow and BERT models. The best obtained result on the Czech dataset is from BERT model trained on Czech with exact match 60.48% and F1 score 73.46%. In addition, we have also trained BERT model on English dataset and we have evaluated it on Czech testing dataset without translation. We have reached exact match 63.71% and F1 score 74.78%, which is extremely good in spite of the fact that the model has not seen any Czech question answering data before. Such a model is very flexible and provide a question answering system in any language for which we have enough monolingual raw texts.

Keywords: Question answering Natural language processing SQuAD Bidirectional Attention Flow Transformer BERT

# Contents

**1 Introduction** **3**

**Introduction** **3**

1.1 Our Contributions . . . 3

**2 Question Answering** **5**

2.1 Natural Language Processing . . . 5

2.1.1 History . . . 5

2.1.2 Reading Comprehension and QA . . . 6

2.1.3 Applications . . . 6

2.2 Question Answering . . . 7

2.2.1 History . . . 7

2.2.2 Domain Types . . . 8

2.2.3 Basic Approach . . . 8

2.2.4 Application . . . 9

**3 QA Datasets** **10**

3.1 Existing Datasets . . . 10

3.1.1 MCTest . . . 10

3.1.2 Wiki-QA . . . 11

3.1.3 TREC-QA . . . 11

3.1.4 News-QA . . . 11

3.1.5 CNN/Daily . . . 11

3.1.6 Children's Book Test . . . 12

3.1.7 Summary . . . 12

3.2 SQuAD Dataset . . . 12

3.2.1 Data Collection . . . 13

3.2.2 Dataset Analysis . . . 13

3.2.3 Evaluation . . . 15

3.2.4 Conclusion . . . 16

**4 BiDirectional Attention Flow** **17**

4.1 BiDAF Model . . . 17

**5 Bidirectional Encoder Representations from Transformers** **20**

5.1 Training Procedure . . . 21

5.1.1 Pretraining . . . 21
5.1.2 Finetunning . . . 22
5.2 Transformation for Question Answering . . . 22
5.3 Results . . . 23

**6 Constructing Czech Question Answering Dataset 24**
6.1 Introduction . . . 24
6.1.1 Dataset . . . 24
6.1.2 Translation of the Data . . . 25
6.1.3 Evaluation Metrics . . . 32

**7 Model Training and Evaluation 33**
7.1 BiDAF Model . . . 33
7.1.1 Translation of English Data to Czech . . . 33
7.1.2 Translation of Czech Input into English . . . 35
7.1.3 Summary of Results . . . 36
7.2 BERT Model . . . 36
7.2.1 Main Findings . . . 40
7.2.2 Summary . . . 41

**8 Conclusion 42**

**Conclusion 42**

**Bibliography 45**

**List of Figures 47**

**List of Tables 49**

**A Overview of Electronic Attachments 50**

# 1. Introduction

Question answering is a computer science discipline in the field of natural language processing and information retrieval. The goal is to build a system that can automatically find the answer to a certain question in the text, where the computer needs to understand the question and the text to be able to work with it furthermore. Computer understanding of text is achieved by such numerical internal representation of sentences which represents language semantics and word relations thoroughly and the system is able to return a relevant answer to the posed question. Syntactical and semantical analysis of question and corresponding text is also necessary to answer the question correctly.

In English, there exist huge datasets made for this task. In this work we are using only the SQuAD 1.1 dataset, which is English dataset for question answering task with around 100,000 question-answer pairs, which is widely used to train a lot of different models with relatively good accuracy.

Unfortunately, the datasets like this exist only for English and we would like to be able to produce such a models also for other languages. This work covers building similar models in Czech language without having any Czech training datasets by reusing English models and English datasets.

## 1.1 Our Contributions

At first, we translated SQuAD dataset into Czech to create Czech training and development datasets. Unfortunately, every translation of data brings noise into it. We have measured how much the data was damaged after the translation by comparing translated answer with the answer in the text using longest common substring method to find match value between these two answer sentences. Ideally, they should be exactly the same as in English dataset, but in the Czech one they are not. We have split the translated dataset according to the computed match value of the answer and of the part of the text containing the answer and we have observed which amount of data is unusable and should be thrown away. We have found out, that the data with match value less than 80% are damaged too much and should be thrown away. However, only around 20% of all the data had to be discarded this way. For the development dataset, we have chosen only data with match value 100% as for the other datasets, the inaccuracies after the translation were devaluing the evaluation.

After preparing Czech training and development datasets, we have approached

the problem in two ways – employing a high-quality Czech-English translation system, and using cross-lingual transfer from English to Czech directly without any access to translation system or even parallel data. This was done by reusing two models – BiDirectional Attention Flow model and BERT model.

Firstly, we have trained BiDirectional Attention Flow model on translated Czech SQuAD dataset and we have trained all the possible combinations of train and development datasets according to the rounded match value to the nearest value from following – 100%, 95%, 90%, 85%, 80%. The best combination was with training set with match 80% and development set with match 100%.

Secondly, we have trained BiDirectional Attention Flow on English SQuAD and we have translated Czech development dataset to English, we have passed it to the model for evaluation and we have translated the resulting answers back to Czech and evaluated them, obtaining better results than the best model trained on Czech.

Thirdly, we have trained BERT model on Czech dataset with match values 80% and 100% and evaluated it on Czech dataset with match value only 100%, obtaining only considerably better results.

Fourthly, we have trained BERT on English SQuAD and we have again translated Czech inputs into English and then English answers back to Czech and evaluated them, arriving at the best performance for Czech question answering in this thesis.

Lastly, we have evaluated BERT directly on Czech development dataset while training it on English to avoid necessity of translating and damaging data. Surprisingly, the BERT model surpasses both BiDirectional Attention Flow models and BERT models trained on the translated Czech data. Such a system can be reused for any language for which only a raw monolingual data are available while still reaching very good performance.

# 2. Question Answering

In this chapter, we describe the basics of natural language processing that occupies with transforming of humans language into machine language. It is an important area, that allows machines to work with natural language and solve lots of different tasks as machine translation or question answering. After this brief introduction into natural language processing, we describe the basics of question answering, which is the problem we are trying to solve.

## 2.1 Natural Language Processing

Natural language processing (NLP) is a sub area of linguistic, artificial intelligence and computer science, which analyses natural language tasks that require natural language comprehension. Basically, it occupies with the interaction between human and machine language with goal to represent and work with data from natural language. This chapter was inspired by Wikipedia [a].

NLP has many application, for example machine translation, speech recognition, information retrieval, natural language generation and recognition or text to speech processing. Question answering task, which is described in this work, is also one of the fields of application of NLP.

### 2.1.1 History

The history of NLP generally started in 1950s, when Alan Turing published so-called Turing test, which was able to test intelligence of machines.

Later, first experiments with automatic translation started. The biggest boom started in 1960s with development of first machine learning algorithms and statistical systems. Increase of computational power, amount of data and human necessity of machine translation of increasing number of documents were also very important reasons of this big boom in this branch in this decade.

In 1980's, the experiments based od statistical machine translation and maximum entropy models were restored and became the most popular for the creation of more effective models.

In the beginning of second decade of the $20^{\text{th}}$ century natural language processing started to tend to use huge neural network systems, that showed their huge computational power and gave good results.

Recent researchers aim to train and develop unsupervised and semi supervised

algorithms, which are able to learn from unlabeled data, which we have almost unlimited amount. They supplement supervised methods and allow further exploration as they do not suffer from lack of training data, which is caused by the fact that to label a dataset it takes a really long time.

### 2.1.2 Reading Comprehension and QA

Reading comprehension (RC) is one of the NLP areas. Understanding of the text is considered to be a crucial for question answering task. This section was inspired by Wikipedia [c].

Reading comprehension is an ability of the machine to process text and to understand it. Fundamental skills for reading comprehension are

- understanding of meaning of single words
- understanding of meaning of whole sentences,
- relations among words and sentences,
- ability to make links to previous words or sentences,
- understanding of the meaning of whole text,
- extract the main idea of text,
- deduction capabilities,
- determine the other situation influences (mood, intonation, environment, context).

Computer understanding of text is such preprocessing that can reuse context of words and sentences to create an internal representation of input text that represents language semantics and word relations thoroughly and the system is able to return a relevant response to a posed query.

### 2.1.3 Applications

NLP is a very important discipline that allows to process and to represent natural language in computers. Among the most important problems, which it helps to solve, belong machine translation, grammatical error correction, text-to-speech synthesis, automatics speech recognition, summarization of the text, chatbots and dialog systems, question answering and many others.

## 2.2 Question Answering

Question answering (QA) is a computer science discipline in the field of natural language processing and information retrieval. Its goal is to build a system that can automatically find the answer to certain question in the text. For that, computer needs to understand the meaning of the question and of the text. Then, it will be able to work with it furthermore.

Question Answering Systems are computer programs used for obtaining the correct answers to the questions. The question is posed by human in natural language and computer needs to process it, process the text and reply relevantly. Fundamental thought of these systems is to assist man-machine interaction. Question answering implementation is usually a computer program that deducts answer to posed question by querying a structured database of information which it creates from text and training data. This section was inspired by Wikipedia [b].

### 2.2.1 History

The history of QA starts in 1950s a little while after the beginning of NLP processes. First systems which were invented to answer the question were BASEBALL and LUNAR. BASEBALL was domain-based QA system made in America in 1960 which was able to answer question about baseball league in US. LUNAR was another domain-based system which was able to answer the questions about geological analysis of rocks from mission Apollo on the Moon. Both of the systems were really efficient. They were not like today's QA systems as they did not understand the text, but they were only decomposing the question, finding the keywords and then using deduction if the database of knowledge to find correct answers.

During next years, another systems were developed for QA tasks in domain-based QA. Especially, in 1970s a lot of knowledge-based domains for plenty of different areas with relatively good accuracy were created. All of them had one common feature - they were similar to todays ones but they were based on deduction more than on understanding of the text. They had a huge database of knowledge about the domain from which they deduced answers as mentioned above. For example ELIZA used similar principles for information retrieval.

In 1970s-1980s complex development of statistical linguistic theories started. It motivated human to teach machines how to understand the meaning of the texts and not only to deduce some information without better understanding of

it. One of these first systems was Unix Consultant developed by the end od 1980s. This system was able to answer specific questions about operating system Unix according to the type of answer. LILOG was similar system for understanding of the text concering about tourist industry in one German city. They both really helped in development of computer deduction and text comprehension.

Nowadays, specialized systems to answer questions in natural language were developed. They are even able to understand the meaning of the text and the question. The most common ones are Wolfram Alpha, which is an online computation machine and EAGLi that can answer questions dealing with life health. With boom of deep neural networks, they started to be widely trained to solve this kind of tasks.

### 2.2.2 Domain Types

There are two types of question answering domains. Closed-domain QA and open-domain QA.

Closed-domain QA occupies with questions in specific domain, for example medicine or literature and they can simply reuse knowledge specific for particular domain, which is often formalized in ontologies.

Open-domain QA occupies with question about literally everything and it can rely only on general ontologies and world knowledge. Naturally, these systems need many more datasets which they can use for deduction of answer. For that, it is more complicated to train them.

### 2.2.3 Basic Approach

Firstly, we need to define a model, which we can now imagine as black box, that will be able to answer the questions. A huge training set will be necessary to enable us to teach model how to answer the questions with respect to certain context paragraph. Data set has to consist of paragraphs with relied questions and their answers. Deep learning neural network has been shown to be a good model for this task.

In the basic training process, it is necessary to preprocess the dataset by tokenizing each paragraph with content, question and answer into single words. Then, relations among the words have to be found. After having a suitable input data representation, training of the model can start. Training is learning of mappings from the questions to the text and also mappings between answers and question and answers and text. When model is trained, it can be subsequently

used to predict answers of previously unseen questions.

To answer given questions using a trained model, we process as follows. After posing a question as input, keyword extraction is a first step. It can be question words as 'who' or 'where' or all other important keywords that can decipher the type of the question and then the correct answer in the text. Word tagging is widely used to identify the words and finding the correct type of the question. After finding the right keywords, information retrieval from the text can be used to obtain the answer.

### 2.2.4 Application

QA systems are really important and useful systems with application in a wide variety of tasks. Nowadays, there are huge unlabeled data in the world accessible to anybody. Information retrieval from this huge datasets is a complicated task. Earlier, it was done by humans. Nowadays, QA systems allow us to facilitate such a tasks and they can do them automatically by computers.

Another wide area of application is in the development of dialog systems and chatbots, that are designed to simulate human conversation and even help people to solve their problems and receiving information. Some of them can be seen for example in e-shops, where they are trying to help customers to find what they need.

# 3. QA Datasets

Question Answering is a very popular topic. There exist a lot of datasets in English for this task. In this chapter, we will explore these most common datasets and describe the SQuAD dataset, which we have chosen to work with. This chapter was inspired by Pranav Rajpurkar and Liang [2016].

## 3.1 Existing Datasets

Reading comprehension is an ability to read and understand a text and then eventually answer the questions that was posed to the content of context of the text. It is a big challenge for the machines as it requires natural language understanding and knowledge of the world also. There exist many English datasets for this tasks that vary in size, difficulty, and collection methodology. Unfortunately, the high-quality ones usually made by human are too small and the bigger ones that are generated automatically by machines are not so good for training. The main reasons why they are not so good are that they are not specific enough for this task and questions are not posed in natural language. They are generated automatically and for that they may not test comprehension directly. We begin with a brief survey and comparison of available datasets. We describe SQuAD, MCTest, TREC-QA, Wiki-QA, News-QA and CBT.

### 3.1.1 MCTest

Machine Comprehension Test (MCTest) is a freely available dataset that consists of 660 elementary-level children's stories with associated questions for the machine comprehension of the text. It was designed in 2013 by Richardson et al. This dataset was created by crowd-workers with 4 questions per paragraph and 4 different choices of answers for each question. The stories and questions were carefully limited by reducing the world knowledge that is required to be known for the task. The questions are designed to require basic level of reasoning and this is making the dataset quite challenging. Moreover, the data set is not big enough to produce a good results if we use it as training data for our models. More about this dataset can be found here Richardson et al. [2013].

### 3.1.2 Wiki-QA

Wiki-QA if a freely available dataset for open domain question answering created by crowd-workers. This dataset contains 3047 questions which were originally sampled from real Bing queries based on Wikipedia articles. The creation of this dataset is quiet similar to SQuAD dataset. The only difference is that its answer selection is a whole sentence of the text selection but SQuAD only requires selecting a specific span in the sentence in the text. More about Wiki-QA can be found here Yang et al. [2015].

### 3.1.3 TREC-QA

Text REtreival Conference (TREC) focuses on creating different question answering datasets since 1999 and since that time, several different QA datasets were released. The last dataset contains 1479 question-answer pairs which is not enough and for that, we cannot use it to train our models as we would not get sufficient results. More about TREC-QA can be found here TREC [2019].

### 3.1.4 News-QA

News-QA is a freely available challenging machine comprehension dataset with almost 120,000 human-generated question-answer pairs based on more than 10,000 news articles from CNN, where answers are spans of text in corresponding articles. The most closely related comprehension dataset to this one is SQuAD. On the other hand, SQuAD is more realistic then News-QA. Moreover, some of the questions have no answer in corresponding article, what makes this dataset more challenging. More about this datasets can be found here Trischler et al. [2016].

### 3.1.5 CNN/Daily

The CNN/Daily Mail corpus consists of 1.4 million of question - answer pairs from the news articles from CNN newspapers. Dataset was created automatically by taking articles as a source text for posing questions. For each article, questions were generated synthetically by deleting a single entity from abstract summary texts, which follows each article. For that, finding the correct answer is mostly achieved by recognizing contextual link between the article and the question. The process is automatic so it is not so difficult to create big amount of data. Unfortunately, it was proved that this task needs only limited amount of reasoning

steps and the accuracy of the best models is almost same as human's accuracy. It means that this dataset is big enough, but it does not have enough quality to be used. More about this datasets can be found here Chen et al. [2016].

### 3.1.6 Children's Book Test

The Children's Book Test (CBT) was created by similar process as CNN/Daily Mail. Instead of news articles, 20-sentence excerpts from children's books were taken. Questions are generated by deleting a single word in each $21^{st}$ sentence of the text. It was generated automatically. There are 4 splits of the dataset and each split contains over 100,000 stories. This is good size to train deep learning models for our QA task, but it is not real QA, because there is no questioning, but only filling a missing word in each $21^{st}$ sentence. More about this datasets can be found here Hill et al. [2016].

### 3.1.7 Summary

We have gone through several datasets and described and compared them. However, reading comprehension is really difficult task and if we want machines to make a progress on these challenges we need to have huge but also a high-quality datasets to train the models properly. These two parameters of the datasets are both of the same importance to achieve good results while selecting appropriate dataset for training. For that, we have chosen SQuAD dataset in this work as we found it as the best one for this task.

## 3.2 SQuAD Dataset

The Stanford Question Answering Dataset (SQuAD) version 1.1 Pranav Rajpurkar and Liang [2016] is a freely available reading comprehension dataset consisting of 107,785 question-answer pairs based on 536 articles. It was created by crowd-workers on a rich set of Wikipedia articles. Unlike the other datasets, every answer to a question is a segment of text from the corresponding reading paragraph. It was not the first dataset that was ever created, but it was the first huge high-quality one for reading comprehension tasks.

Dataset answer selection is as follows. For selecting the correct one, system must get through all possible spans in the text and find the one that is matching the most to posed question. It generates a high number of possible candidates which must be compared and evaluated. Then, the best one must be chosen as

the required answer. For that, the special techniques based on distances and dependency trees are used.

Let us describe how the dataset was created to understand better our model results after training on it. This section was inspired by Pranav Rajpurkar and Liang [2016].

### 3.2.1 Data Collection

This dataset was collected in three stages:

1. Article selection

   To obtain high quality data, the top 10,000 articles from English Wikipedia website was taken. From this amount of data, 536 articles were randomly sampled. Then, these articles were divided into individual paragraphs and all non-textual data were erased. Also, the articles and paragraphs shorter than 500 characters were removed as they do not contain enough information to pose questions to it. Then, resultant 23215 paragraphs were split into training set and test set, while training set is 80% size of original dataset and test set is remaining 20%.

2. Question-answer collection

   On each paragraph, crowd-workers made manually up to 5 questions asking about its content. Each answer to each question was required to be a part of the text.

3. Additional answers collection

   To make evaluation more robust, at least 2 additional answers were created for each question in development and test set. If some questions were unanswerable in the text, crow-workers created an answers without marking them in the text.

### 3.2.2 Dataset Analysis

It is necessary to analyze the questions and the answers to understand properties of whole dataset. Three main aspects were analyzed for measuring, how difficult the answer is for the system:

1. Diversity of answer types

Analysis of diversity in answers means to automatically categorize the answers to numerical or non-numerical. Non-numerical are subsequently separated according to the word class. It is called POS tag. One of the POS tag categories are nouns. Nouns are then tagged accordingly to the place, time, person, etc. This tags are called NER tags, see Figure 3.1.

2. Reasoning required to answer questions

   It measures difficulty of answering of questions based on reasoning that is necessary for selecting the correct answer. For that, they had sampled several questions from each article and then manually labeled the examples with categories mentioned above. The results showed, that each of the answers has some syntactical or lexical deviation between the question and answer in the text. They have described these deviations in their article.

3. Degree of syntactic divergence between the question and answer

   Stratification by syntactic divergence is an automatic method for quantification of syntactical divergence between answer and question which measures difficulty of the answer. It was established as minimal distance between all possible words which belongs to the answer, see Figure 3.2.

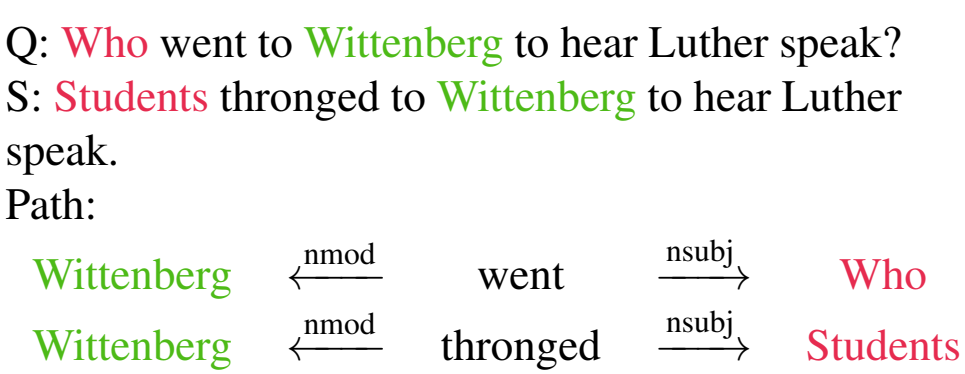


Figure 3.1: An example showing the keyword selection and dependencies modeling between answer and question. Source Pranav Rajpurkar and Liang [2016].

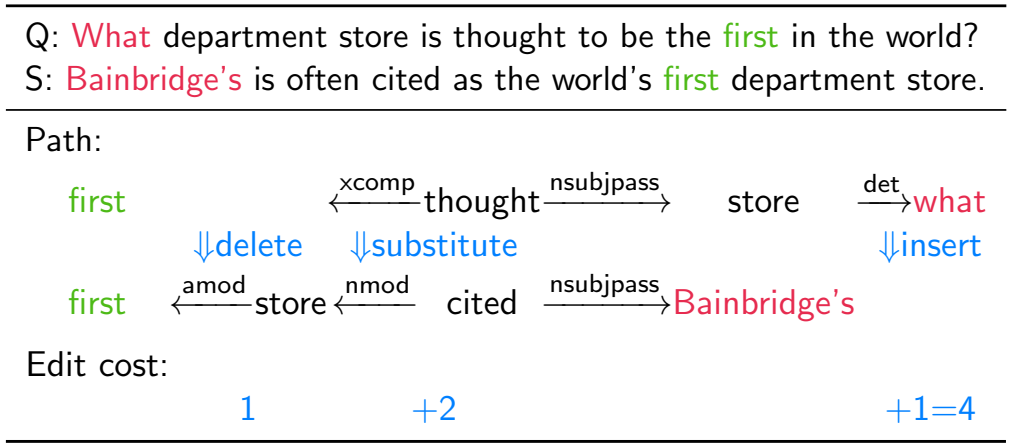


Figure 3.2: An example showing the computation of syntactic divergence between answer and question. Source Pranav Rajpurkar and Liang [2016].

#### Methods for Analysis

A logistic regression model was created and compared to candidate answer generation method and sliding window method. Generating of candidate answer means passing character by character and generating all possible answers and finding the best one. Sliding window method means computing unigram/bigram overlap between sentence containing answer and question and by using sliding-window select the best answer.

In logistic regression model, several types of features for each candidate question were selected. They were devised according to the linguistic analysis specially for this task and they are matching of word frequency, match of bigram frequency, match of root, span of word frequency, lexicalization, parsing and path in dependency tree. Loss is computed by AdaGrad with initial learning rate 0.1.

### 3.2.3 Evaluation

There were used several metrics.

1. Exact match

   Exact percent match between whole real answer and whole predicted answer is computed.

2. F1 score

   Matching real answer and predicted answer word by word and computing accuracy word by word. Then, mean value is taken.

### 3.2.4 Conclusion

This dataset is the best for our QA task as it is high-quality and big enough for training deep learning neural network. Unfortunately, it does not solve problem with unanswerable questions. For that, SQuAD 2.0 was created 2 years later. It combines SQuAD 1.1 with 50,000 unanswerable questions that are linked to already existing paragraphs. It has lower accuracy than SQuAD 1.1 because unanswerable questions are more challenging. However, in our models, we use only SQuAD 1.1, as it is enough for us to have only answerable questions.

# 4. BiDirectional Attention Flow

Reading comprehension and question answering tasks gained significant popularity in past several years and the necessity of effective and accurate models increased with it. Nowadays, end-to-end trained systems reach promising results in domain-based QA tasks. One of the key factors in such a progress was the usage of an attention mechanism, that allowed systems to concentrate on target area in the context of a paragraph, which is the most relevant to answer the question.

Bidirectional Attention Flow (BiDAF) Seo et al. [2016] is a hierarchical multi-stage architecture for QA tasks. BiDAF is modeling representation of context in several layers as character layer, word layer and contextual layer with usage of attention mechanism. Attention is computed in each step in both directions context-query and query-context and it allows both sides to communicate and share information. For that, the loss of the information was reduced and more accurate models were created.

By the time BiDAF was released, it was trained on Stanford Question Answering Dataset (SQuAD) and it has outperformed all already know models. This chapter was inspired by Seo et al. [2016].

## 4.1 BiDAF Model

BiDAF model has multi-level architecture which consists of 6 layers, see Figure 4.1.

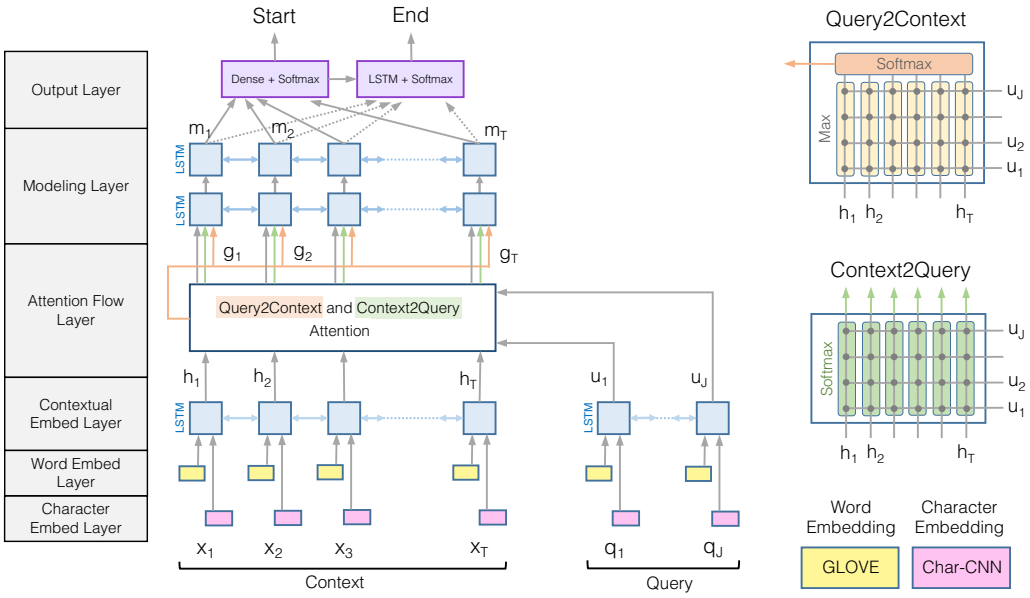


Figure 4.1: BiDirectional Attention Flow model. Source Seo et al. [2016]

1. Character Embedding Layer

2. Word Embedding Layer

   In these two layers, the text is split word by word and numerical representation of each word is obtained. It means that each word is mapped into vector space according to the pretrained model. It is mostly performed by complex convolution networks. Character embeddings represent how the word looks like in its written form and word embeddings represent the meaning of the words.

3. Contextual Embedding Layer

   This layer takes as input embeddings from previous layer and apply Long Short Term Memory Network (LSTM) to model the interaction between words. Its output are tuned embeddings of each word which should represent word meaning in given context.

4. Attention Flow Layer

   This layer is responsible for linking information from the context to the words in the question. In the contrary to the previous popular attention mechanisms, this one is not used for summary of vector and context into the one new vector, but it allows them both separately to float into modeling layer. Moreover, the attention vector is added. This process can decrease

information loss and increase model accuracy. Input of this layer are vector representations of content and query. Outputs are vector representation of context words with embeddings from previous layer.

Bidirectional attention means that it has two directions of processing vectors. First direction is text to attention query, which marks words in the question which are most relevant for each word in the content text. Second direction is query to attention that marks, which words from the text are the most relevant for each word in the question and which of them are also important for the answer. Basically, we have matrix that says how much the words from the query fit the words from the text. From that, we obtain vector which says, what from the question matches the most words in the text and for each word in the text we got vector that says what from the question matches this word.

5. Modeling Layer

   For each word we now have context embedding vector from layer 3 that says what from the question matches this word and the vector that describes the question from layer 4. Subsequently, all these inputs are concatenated together. From that, we have encoded representation of perceiving words from query in context. This layer uses bidirectional LSTM network that produces vector of relations among the words in the context with respect to question.

6. Output Layer

   Output Layer takes the outputs from previous layer made by LSTM network and process it furthermore. To be more detailed, its inputs are representation of perception word from the context in query and contextual information about words in context with respect to whole paragraph and query. Then, this layer classifies where the start and end indices of the answer in the context are. For each possible index the probability is computed from pretrained weights. Finally, the most corresponding answer is chosen as the part of the text bounded with these indices with the highest probability.

Loss function for validation this model is sum of negative squares. According to the article, they have trained model on SQuAD dataset with F1 score 81.1% and exact match (EM) score 73.3%.

# 5. Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers (BERT) Devlin et al. [2018] Wolf et al. [2019] Radford et al. [2018] is a new universal language representation model designed to pretrain deep bidirectional representations from unlabeled text. Subsequently, it can be finetuned with only one additional output layer to create specific model for a certain task.

Previous general language processing models only used unidirectional language models to learn general language representations from left to right. It has been shown in Wolf et al. [2019] that this restricts the power of the pretrained representations because it limits the choice of architectures as each token can only attend to previous one in self-attention layers. This restriction can be very harmful when applying finetunning for certain tasks. For example, in question answering, it is very important to consider context in both directions among question, answer and text.

BERT solves this deficiency by using a language model, which randomly masks some of the tokens from the input and tries to predict the original words based only on context. It allows to link left and right context and to pretrain bidirectional Transformer, which subsequently creates contextual representation. It also uses next sentence prediction to achieve better results. Next sentence prediction means that model receives an input document and the pair of sentences and tries to predict whether the second given sentence is following the first given sentence in the documents. Finally, whole sentence context is represented in embedding of each word, see Figure 5.1. This chapter was inspired by Devlin et al. [2018].

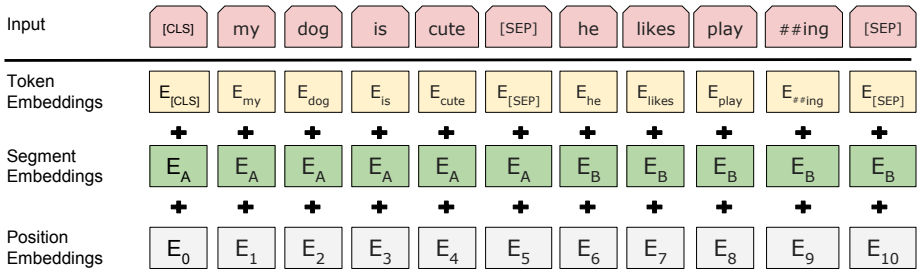


Figure 5.1: BERT input representation. Source Devlin et al. [2018]

## 5.1 Training Procedure

Training procedure has 2 steps, see Figure 5.2.

1. Pretraining

   This step has the goal to train and create high capacity language model on unlabeled corpus.

2. Finetunning

   Subsequently, the language model can be finetuned and adapted on the labeled data specific for target task.

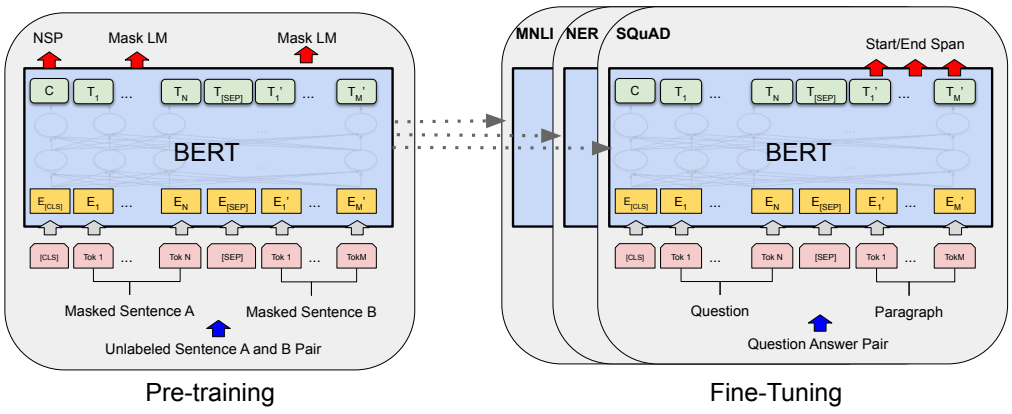


Figure 5.2: Overall pretraining and finetunning process for BERT. Source Devlin et al. [2018]

### 5.1.1 Pretraining

Neural network is used for unsupervised training on unlabeled corpus of tokens. Particularly, multilayer Transformer decoder for creation of language model was

developed, see Figure 5.3. It applies multihead self-attention over the input context token. Then, feedforward layer based on positions is used for creating output distribution over target tookens. More about Transformer model can be found at Wolf et al. [2019].

### 5.1.2 Finetunning

After training a model we want to adapt parameters pro given target. We have supervised dataset, where each input sequence of tokens has its label. Inputs are passed on pretrained model to reach final activation that is added to output linear layer to predict label. Moreover, we have found that finetunning has improved model generalization and speed up convergence.

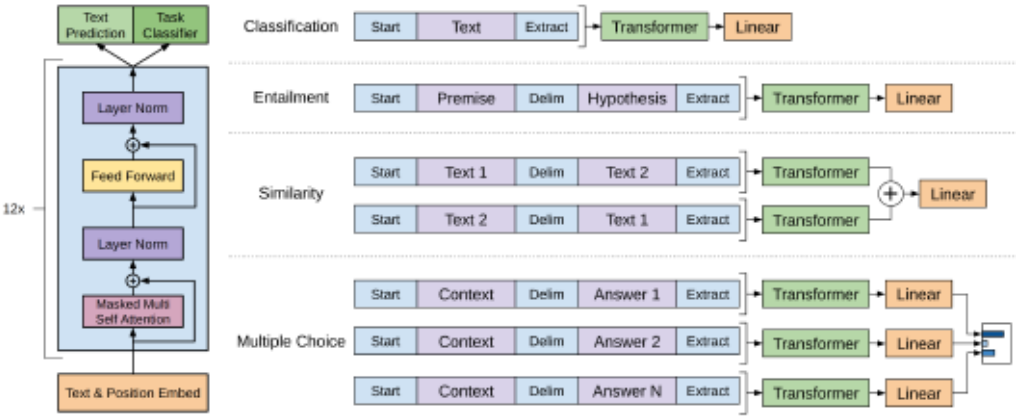


Figure 5.3: Overall Transformer architecture. Source Radford et al. [2018]

## 5.2 Transformation for Question Answering

QA task has structured inputs, so the modification of input is necessary before passing it into the model. The advantage is that instead of suggesting a specific architecture for a given model, we will only convert structured inputs into ordered sequences which can be processed by our model.

Each transformation requires start and stop tokens (`<s>`, `<e>`) randomly initialized. As input, we obtain context document $c$ and question $q$ and set of possible answers $\{a\}$. For that, we need to join document, question and each possible answer into one sequence with adding separating token as follows $[c; q; \$; a]$. Each of these sequences is preprocessed by the model independently and then normalized by softmax layer to receive output distributions over possible answers. More precisely, whole question and answer is represented together in one sentence.

## 5.3 Results

According to the article, they have trained model on unlabeled dataset and then it was finetuned for QA task on SQuAD 1.1 dataset. They have used two versions of models – $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ which differs in number of layers, number of self-attention heads and hidden layer size. The $\text{BERT}_{\text{LARGE}}$ has outperformed the $\text{BERT}_{\text{BASE}}$ model and has reached F1 score 93.2% on testing dataset and 92.2% on development dataset. We can notice, that it is much better then BiDirectional Attention Flow discussed in previous chapter, see Table 5.1.

| **System** | **Dev(EM)** | **Dev(F1)** | **Test(EM)** | **Test(F1)** |
|---|---|---|---|---|
| $\textbf{BERT}_{\textbf{based}}$ | 84.2% | 91.1% | 85.1% | 91.8% |
| $\textbf{BERT}_{\textbf{large}}$ | 86.1 % | 92.2% | 87.4% | 93.2% |

Table 5.1: Comparison of results of BERTs trained on English from Devlin et al. [2018].

# 6. Constructing Czech Question Answering Dataset

In previous chapters, we have covered the basics of reading comprehension and question answering. Also, we have described two main models designed for this task – BiDAF Seo et al. [2016] and BERT Devlin et al. [2018] and SQuAD dataset Pranav Rajpurkar and Liang [2016] which was used for training. Especially the second model has achieved very good results. Unfortunately, this dataset is in English language. We would like to be able to train such models also for QA in Czech language. Ideally, without a necessity of translating any data. In this chapter, we describe, how previously described dataset and models can be reused for reaching this goal.

## 6.1 Introduction

At first, we have to describe basic tools and technologies, that we have used for reaching the goal of our work and describe dataset structure in detail.

### 6.1.1 Dataset

We have downloaded the dataset SQuAD 1.1 from `https://rajpurkar.github.io/SQuAD-explorer/`. We have used it for training ad evaluation and also for creating of Czech dataset.

The structure of the dataset is as follows. There are two JSON files. First one is *train-v1.1.json* and it contains all data for training – there are context paragraphs with several questions and each question has one answer. Second one is *dev-v1.1.json*, which is used for evaluation. The structure of this file is almost the same with one difference. It was annotated manually by several crowdworkers, so there can be several answers for one question. The most matching answer was always chosen to be compared with the predicted one to reach the highest accuracy. It also allows a little deviations in answering, which can be useful as the predicted answer is not always identical with the original one and still can be correct. The size of training dataset is 87,599 questions and development set is 10,570 questions.

The structure of both data files looks as follows. There is a tag *data* containing list of all articles. Inside this tag, there is always a title of the article in *title* tag

having a list of single paragraphs containing the context related to this title. They are called *paragraphs* tags. Each paragraph has its own list with answers and questions in *qas* tag, which furthermore consists of three tags. First one is *question* tag, which contains the text of the question. Second one is *id* tag, as each question has its own id for easier identification. Last one is *answers* tag containing the text of the answer in *text* tag, and also, starting index of the answer in the text represented in the *answer_start* tag.

Basically, the structure looks like this:

```
{ Data[{
        title
        paragraphs {[
                context
                qas [{
                        answers[
                                text
                                answer_start
                        ]
                }]
                question
                id
        ]}
}]
version }
```

### 6.1.2 Translation of the Data

We have used several data translation of the SQuAD dataset to Czech and eventually back to English language. For that, we have used LINDAT Translator, which is the best translator between Czech and English, that is developed at Faculty of Mathematics and Physics at Charles University by the Institute of Formal and Applied Linguistics. More about this translator can be found here Popel.

In English dataset, the answer in the *text* tag in the *answers* tag is exactly the same as a part of the text in the *context* tag. Unfortunately, the translation of the dataset brings a noise into in. Therefore, the part of the text containing the correct answer and the answer in *text* in *answers* tag may differ after translation of the dataset. It is caused because of different grammatical rules between these languages. Czech language has much richer inflectional morphology which can

cause problems during translation. Moreover, the sentence in the context of the paragraph can be translated in different way than the answer itself. Moreover, the *answer_start* tag value must be recomputed as the order and the length of the words may have changed after translation.

As mention above, when we have translated all of the paragraphs, answers and questions, the start index of each answer in the text had have to be recomputed as we need it during training. The problem is that we cannot use exact match because of several reasons.

First one is that the answer may not fit the text exactly. For that, we cannot use exact match and we need to go word by word and find longest common substring by following algorithm, see Algorithm 1. We start with the whole text and compute match between it and the translated answer. Meanwhile, we systematically delete first word from the remaining until we have empty string. We measure which of the resultant common substrings is the longest one. We only compute longest common substrings that ends with the end of the word.

**Algorithm 1** Recomputation of the start index of the answer in the text.

**Input:**

$text$ = translated text

$answer$ = translated answer

$idx$ = original index of the middle of the answer

**Output:**

$bestMatch$ = start index of the bes answer

**Algorithm:**

```
for i = 0 to len(text) do
  if i is beginning of the word then
    lcs = longestCommonSubstring(text[i:len(text)], answer)
  end if
end for
maxLcs = find maximum of lcs
if maxLcs is only one item then
  return maxLcs
else
  for i = 0 to len(maxLcs) do
    maxPos=maxLcs.match*(1-abs(idx-maxLcs[i].idx)/len(text))
  end for
end if
return max(maxPos)
```

The other problem is that there can be more occurrences of the answer in the text and only one is the correct one. For that, we consider that the text sentences are approximately in the same order in English and in Czech and that the original answer position can be used to find the correct position of translated answer. Therefore, for each possible answer we compute its final score according to its value of longest common substring match multiplied by the distance from the position of original answer in original text. To facilitate the work, middle position of the answer is taken. The nearer is the actual position to the original one and the more similar texts are, the higher score this candidate answer obtains. Finally, the answer with the highest score is chosen and its starting index is taken as the correct one. If the starting index points to the middle of the word, it is moved that it points to the beginning of it.

To facilitate our work with translated data, we have modified the final JSON file. Two new tags into the *answers* tag were added. First one is *answer_end*, which is computed during recomputation of the starting index. It is pointing to the end of the last word of the answer in the text and it was added because of the easier visualization of the answer in the context paragraph. This tag is also useful while selecting the answer from the text as in *text* tag, we have translated answer and not exactly the text of the answer from the text. The other one is *answer_match* and it is value of the score of the match, see the new structure below.

```
{ Data[{
        title
        paragraphs [{
                context
                qas [{
                        answers[
                                text
                                answer_start
                                answer_end
                                answer_match
                        ]
                }]
                question
                id
        }]
}]
version }
```

Unfortunately, every machine makes some mistakes during translation. We now describe the most common ones. One of them is word order, that is confusing the system while recomputing start index of the answer in the text. One of the examples can be seen in Figure 6.1. The other common mistake is caused by synonyms. The translator has chosen two different Czech words in question and answer for the same one in English, see Figure 6.2. Another problem is caused by different language properties - Czech words are declined, see Figure 6.3. The last common mistake I mention is translating of numbers. They can be once written as words and translated and secondly written as numbers. Then, the algorithm of recomputing index is confused, see Figure 6.4. The same problem is with the names, see Figure 6.5.

**Pes**

| | |
|---|---|
| **CONTEXT** | Pes domácí (Canis lupus familiaris neboli Canis familiaris) je domestikovaný kanid, který byl po tisíciletí selektivně chován pro různé způsoby chování, smyslové schopnosti a fyzické vlastnosti. |
| **QUESTION** | Co je Canis familiaris? |
| **ANSWER** | domácí pes |

Figure 6.1: Example of the selected answer by the algorithm with changed word order between text and answer.

**Frédéric_Chopin**

| | |
|---|---|
| **CONTEXT** | Všechny Chopinovy skladby zahrnují klavír. Většina je určena pro sólový klavír, i když napsal také dva klavírní koncerty, několik komorních skladeb a některé písně k polským textům. Jeho klávesový styl je vysoce individuální a často technicky náročný; jeho vlastní výkony byly proslulé svými nuancemi a citlivostí. Chopin vymyslel koncept instrumentální balady. K jeho významným klavírním dílům patří také mazurky, valčíky, nokturnovky, polonézy, études, impromptus, scherzos, předehry a sonáty, z nichž některé vyšly až po jeho smrti. K vlivům na jeho kompoziční styl patří polská lidová hudba, klasická tradice J. S. Bacha, Mozarta a Schuberta, hudba všech, které obdivoval, a také pařížské salony, kde byl častým hostem. Jeho inovace ve stylu, hudební formě a harmonii a spojení hudby s nacionalismem měly vliv po celé pozdní romantické období i po něm. |
| **QUESTION** | Jaký nástroj obsahovaly všechny Frédéricovy skladby? |
| **ANSWER** | piano |

Figure 6.2: Example of the selected answer by the algorithm with synonyms in text and answer.

**To_Kill_a_Mockingbird**

| | |
|---|---|
| **CONTEXT** | Jako jižanský gotický román a Bildungsroman obsahují hlavní témata filmu Zabít ptáčka rasovou nespravedlnost a zničení nevinnosti. Učenci zaznamenali, že Lee se v americkém hlubokém jihu zabývá také otázkami třídních, odvahy, soucitu a genderových rolí. Kniha je široce vyučována ve školách ve Spojených státech s lekcemi, které zdůrazňují toleranci a dehonestující předsudky. Navzdory svým tématům je "Zabít ptáčka" předmětem kampaní za odstranění z veřejných tříd, které jsou často napadány za používání rasových nadávek. |
| **QUESTION** | O tom, jak zabít ptáčka, se hodně čte ve školách ve kterých zemích? |
| **ANSWER** | Spojené státy |

Figure 6.3: Example of the selected answer by the algorithm with different declination in text and answer.

**Saint_Barths**

| | |
|---|---|
| **CONTEXT** | Jeden senátor zastupuje ostrov ve francouzském Senátu. První volby se konaly 21. září 2008 a poslední v září 2014. Svatý Bartoloměj se dne 1. ledna 2012 stal zámořským územím Evropské unie, ale obyvatelé ostrova zůstávají francouzskými občany se statusem EU, kteří jsou držiteli pasů EU. Francie je odpovědná za obranu ostrova a jako taková na ostrově rozmístila bezpečnostní síly, které tvoří šest policistů a třináct četníků (vysláni na dvouleté období). |
| **QUESTION** | Kolik senátorů zastupuje St. Barts ve Francii? |
| **ANSWER** | Jedna |

Figure 6.4: Example of the selected answer by the algorithm with non-translated numbers in text and answer.

**Atlantic_City,_New_Jersey**

| | |
|---|---|
| **CONTEXT** | Díky své poloze v Jižním Jersey, obklopujícím Atlantský oceán mezi bažinami a ostrovy, bylo Atlantic City vnímáno developery jako prvotřídní nemovitost a potenciální rekreační město. V roce 1853 byl postaven první komerční hotel The Belloe House, který se nacházel u Massachusetts a Atlantic Avenue. |
| **QUESTION** | Jak se jmenuje první komerční hotel postavený v Atlantic City? |
| **ANSWER** | Dům Belloe |

Figure 6.5: Example of the selected answer by the algorithm with partially translated names in text and answer.

After having all the data translated and successfully recomputed start and end indices, we analyzed them to evaluate, how good or bad the translation are. Figure 6.7 and Figure 6.6 demonstrate how successful the translation was. We can also observe, that we have obtained a bit different results for both sets. The answers in train dataset are with less match than in the development dataset, what is probably caused by the character of answers as the development dataset contains more answers for one question – the question is preserved when at least one of the answers is with required match.

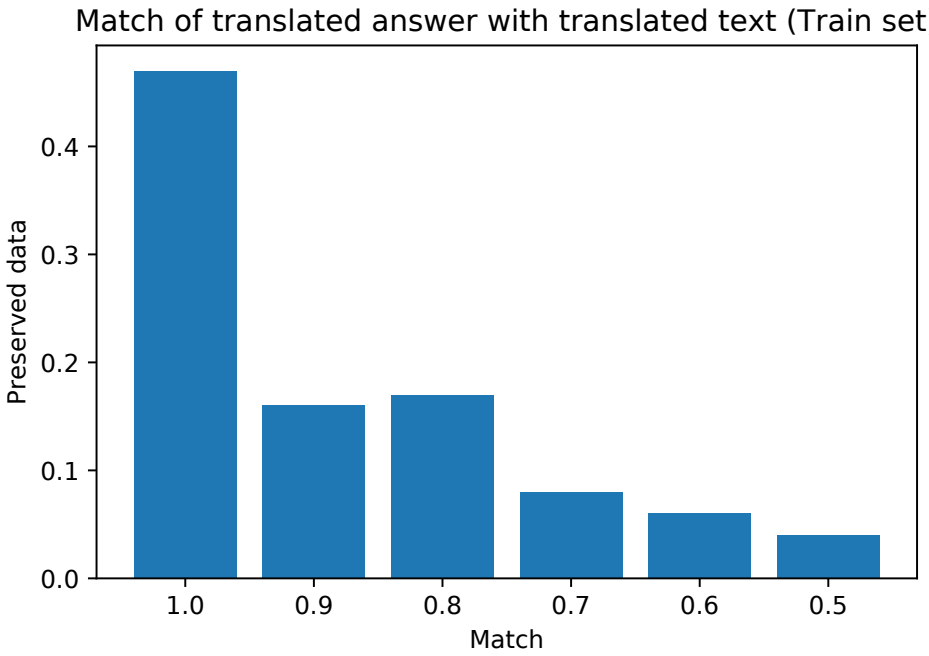


Figure 6.6: Plot of how much the answers match the answers in the text for the training set.

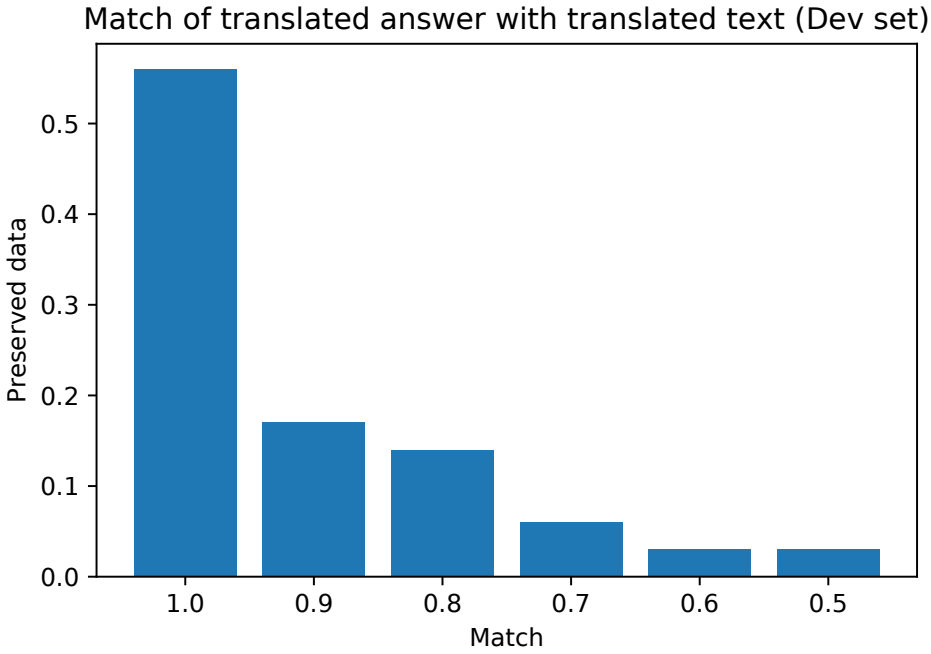


Figure 6.7: Plot of how much the answers match the answers in the text for the development set.

According to the observation of the Figure 6.6, we can see that there are only few translation that have match less than 80%. If we discard them, we still would preserve almost 80% of the data which would hopefully be enough to make good predictions. We can observe exact values of how many percent of data is preserved for different values of exact match in the Table 6.1.

| **Match** | **Train set size** | **Dev set size** |
|---|---|---|
| **100%** | 46.80% | 55.89% |
| $\geq$ **90%** | 63.26% | 73.17% |
| $\geq$ **80%** | 80.18% | 86.93% |
| $\geq$ **70%** | 87.97% | 92.80% |
| $\geq$ **60%** | 93.62% | 96.12% |
| $\geq$ **50%** | 97.69% | 98.75% |

Table 6.1: Preservation of original datasets, where the match is higher than the defined value.

For better observation of the influence of the translation mistakes to the training model process, we have create several versions of the datasets. They contain only answers with match greater or equal to a certain value. We have used 5% increments and created datasets, which have match 100%, more than 95%, more

than 90%, more than 85% and more than 80% for both training and development set. For the information how much percent of original data was preserved in newly created files see Table 6.2.

| **Match** | **Train set size** | **Dev set size** |
|---|---|---|
| **100%** | 46.80% | 55.89% |
| $\geq$ **95%** | 51.97% | 61.88% |
| $\geq$ **90%** | 63.26% | 73.17% |
| $\geq$ **85%** | 72.85% | 81.57% |
| $\geq$ **80%** | 80.18% | 86.93% |

Table 6.2: Preservation of original datasets, where the match is higher than defined value.

### 6.1.3 Evaluation Metrics

For evaluation, we have used two metrics, as described in Chapter 3. First one is *exact match* which compares translated answer with answer in the text and returns a percent, how much the answers fit each other. Second metrics is *F1 score* where the percent of mutual fitting is computed for each word separately and then, the mean value is taken, so it is more accurate.

However, evaluation on translated Czech dataset is not accurate because Czech language is declined an there can be differences in word morphology in predicted and original answer. Therefore, we have lemmatized by MorphoDiTa Straková et al. [2014] all answers in the file with predicted answers and in the original development dataset. We have evaluated these lemmatized answers to achieve more accurate results. Therefore, all the results on Czech datasets are measured after lemmatization of the files.

# 7. Model Training and Evaluation

## 7.1 BiDAF Model

Biderectional Attention Flow (BiDAF) model was trained on SQuAD dataset. More about this model can be found at Chapter 4.

As mentioned above, the goal of this work is to reuse this model to solve QA task also in Czech language. We have used model from `https://github.com/allenai/bi-att-flow/tree/dev`. There are two main attitudes, both linked with machine translation of data between English and Czech. First one is to take whole SQuAD dataset and translate it into Czech. Then, train model in Czech. The second is to train model in English and translate Czech input to English, then let the model to produce the answer and finally, translate it back to Czech.

### 7.1.1 Translation of English Data to Czech

We have taken whole SQuAD data set and translated from English to Czech by LINDAT translator. Then, we have trained model in Czech and tested its accuracy. The training took around 20 hours on GPU dependently on the combination of train and development datasets.

For the training process, we had to change English embeddings used for the English dataset to Czech embeddings. We have used Czech embeddings created on 4 billions of Czech words using word2vec model, keeping embeddings for most frequent 15 million words.

After translation, we have obtained 5 data files for train and 5 data files for testing with different matches between the answer in the text and translated answer as described earlier. To find the best combination of train and development datasets we have tried to train model on all the possible combinations. For that, we have trained 25 models having answers with match from $\geq$80% to 100%. The resultant accuracies and F1 scores of these training processes can be observed in Table 7.1 and Table 7.2.

If we compare all these results in the graph with exact match and F1 score, we can see that the best results we have with training data having answers with match greater or equal to 80% and development set with 100% match between translated answer and answer in the text. The results are not surprising because

| **Dev/Train** | **100%** | **≥95%** | **≥90%** | **≥85%** | **≥80%** |
|---|---|---|---|---|---|
| **100%** | 53.15% | 53.18% | 53.03% | 53.37% | 53.74% |
| **≥95%** | 49.52% | 50.92% | 51.38% | 51.55% | 52.16% |
| **≥90%** | 44.31% | 46.87% | 49.86% | 50.85% | 51.60% |
| **≥85%** | 41.46% | 44.44% | 48.64% | 50.66% | 51.70% |
| **≥80%** | 39.95% | 43.21% | 47.85% | 50.05% | 51.44% |

Table 7.1: Exact match after translation SQuAD to Czech and then training and testing models on data files with corresponding matching values.

| **Dev/Train** | **100%** | **95%** | **90%** | **85%** | **80%** |
|---|---|---|---|---|---|
| **100%** | 62.70% | 64.02% | 64.06% | 64.96% | 65.49% |
| **≥95%** | 60.17% | 62.68% | 63.34% | 64.11% | 64.74% |
| **≥90%** | 55.99% | 59.79% | 62.32% | 63.58% | 64.28% |
| **≥85%** | 53.40% | 57.79% | 61.50% | 63.48% | 64.42% |
| **≥80%** | 52.30% | 57.01% | 60.93% | 63.18% | 64.26% |

Table 7.2: F1 after translation SQuAD to Czech and then training and testing models on data files with corresponding matching values.

match above 80% does not bring such a noise into the dataset as the lower values do, and we also have much more data for training than with match 100%. Therefore, the model can then answer more questions. The reason why the best development set is with answers with match 100% is because the other development sets are noisy and it is harder for the model to predict them correctly, see Graph 7.1.

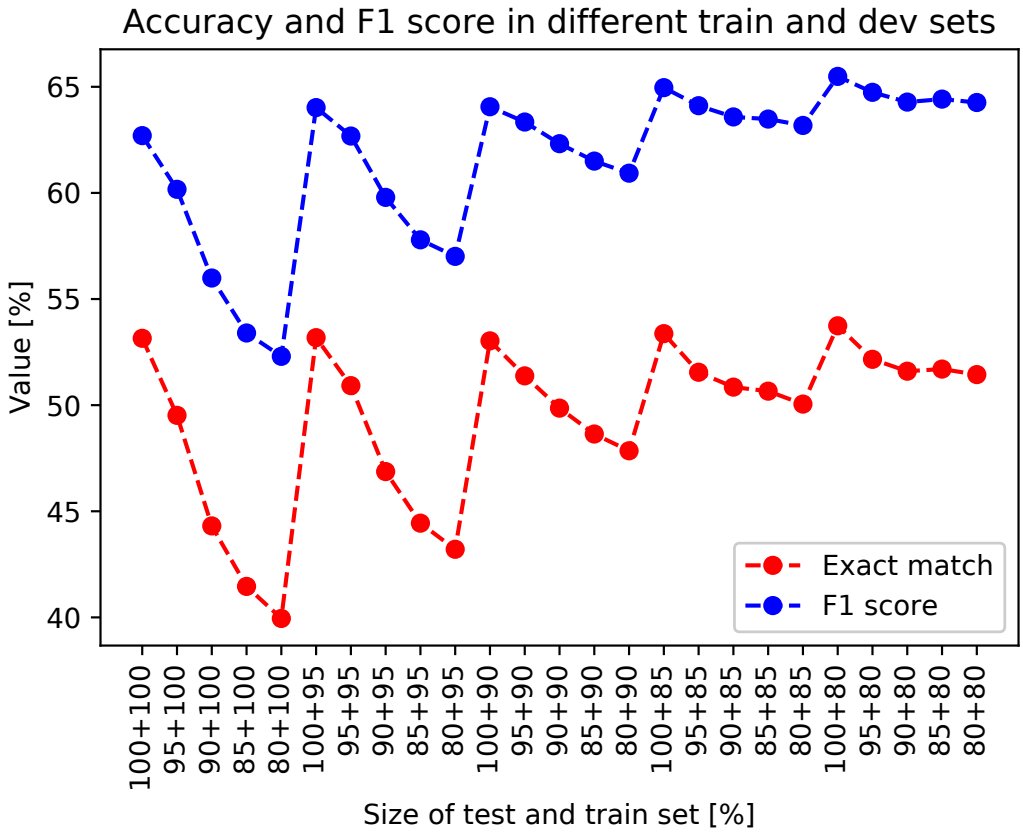


Figure 7.1: Plot of exact match how much the answers match the answers in the text for testing.

### 7.1.2 Translation of Czech Input into English

We have trained the BiDAF model in English. Then, we have taken Czech development data set, we have translated it into English and finally, we have run in on a pretrained English model.

To make the evaluation completed, we have taken English answers and translated them back to Czech language. Then we have measured similarity between original and newly obtained Czech answers. We got these results, see Table 7.3.

| **Model** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| BiDAF | EN | CZ-EN-CZ | 53.57% | 65.84% |

Table 7.3: Results after translation of Czech input to English and then translation predicted English outputs back to Czech for evaluation.

As we do not have any Czech development dataset, we have translated whole SQuAD development dataset from English to Czech. From that, we have chosen only questions and answers having the match between translated answer and

answer in the text 100% or almost 100%. Match was computed by the same way like it was mentioned in the previous chapter and by the same algorithm which is described in Algorithm 1.

### 7.1.3 Summary of Results

If we compare the best result of the model trained in Czech and the best result of the model trained in English, we can see that the total best results we obtained from model trained on English dataset and evaluated with translating Czech input into English and then translating English output back into Czech, see Table 7.4

We think that English model is better because we have more data and there is no noise caused by translation. Moreover, LINDAT translator used for translation was constructed in the way that if we translate sentence from English to Czech and then back to English, it is trying to return original sentence and for that, the loss is minimal.

| **Model** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| BiDAF | EN | EN | 64.22% | 75.29% |
| BiDAF | CZ100 | CZ100 | 53.15% | 62.7% |
| BiDAF | CZ80 | CZ100 | 53.74% | 65.49% |
| BiDAF | EN | CZ-EN-CZ | 53.57% | 65.84% |

Table 7.4: Comparison of results from all models.

To sum it up, with the best model for Czech we have exact match 53.57% and F1 score 65.84% and for English model we have exact match 64.22% and F1 score 75.29%. We cannot compare these results directly as the development datasets are not exactly the same but we can use it just for the orientation how good the Czech models are.

Nevetherless, original English BiDAF model is generally not so good. As mentioned above, it has exact match only 64.22% and F1 score 75.29%. For that, we have tried to reuse another model that gives exact match 80.81% and F1 score 88.27% for English. It is called the BERT model.

## 7.2 BERT Model

We have downloaded the BERT from `https://github.com/google-research/bert`. We have finetuned it on SQuAD dataset to create models for our Czech

QA task. Finetunning took from 3 to 6 six hours on GPU, depending on training and development datasets.

Firstly, we have trained BERT on the original English SQuAD dataset. We have tried different number of epochs. The best results we obtained with 2 epochs so we have trained the other models with 2 epochs as well, see Table 7.5.

| **BERT** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| **1 epoch** | EN | EN | 79.2% | 87.35% |
| **2 epochs** | EN | EN | 80.81% | 88.27% |
| **3 epochs** | EN | EN | 80.03% | 87.8% |

Table 7.5: Comparison of results of BERT using different number of epochs.

After having ideal number of epochs, we have trained BERT also on SQuAD translated to Czech. We have used Multilingual BERT model which was designed for more languages than only English. It was trained on datasets from 100 different languages to create also multilingual version of the BERT that would be able to train models for QA also in other languages.

There are two versions of this model. First one is Multilingual BERT cased which is using cased dataset. Second one is multilingual BERT uncased which is converting dataset to its uncased version by lowercasing and stripping diacritics. We have trained both of this versions on two different combinations of Czech datasets.

At first, we have chosen Czech training dataset witch match 80% and development dataset with match 100%. The reason was that this configuration has given the best results in the BiDAF model. Moreover, we have tested Czech training dataset and testing dataset both witch match 100% because we just wanted to be sure that the previous combination is really the best one for BERT too, see Table 7.6.

| **Multilingual BERT** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| Cased | CZ 80% | CZ 100% | 60.34% | 72.77% |
| Uncased | CZ 80% | CZ 100% | 60.78% | 73.46% |
| Cased | CZ 100% | CZ 100% | 62.12% | 73.0% |
| Uncased | CZ 100% | CZ 100% | 62.14% | 73.46% |

Table 7.6: Comparison of results of Multilingual BERT trained and evaluated on Czech using different training and testing sets.

Surprisingly, we have found that the training on dataset with match 100%

gives us bit better results than the training on dataset with match 80%. We think that as BERT is already pretrained on huge unlabeled corpus, the training dataset with matching value 80% is bigger but bringing more noise into the results.

In addition, we have tried different number of epochs while training these models to avoid under or over fitting. We have compared F1 and exact match score for 1, 2, 3 and 4 number of epochs after training on Czech dataset with match 100% - see Figure 7.2 and also after training on Czech dataset with match 80% - see Figure 7.3. We have found that for both models the best results are achieved after 2 epochs.

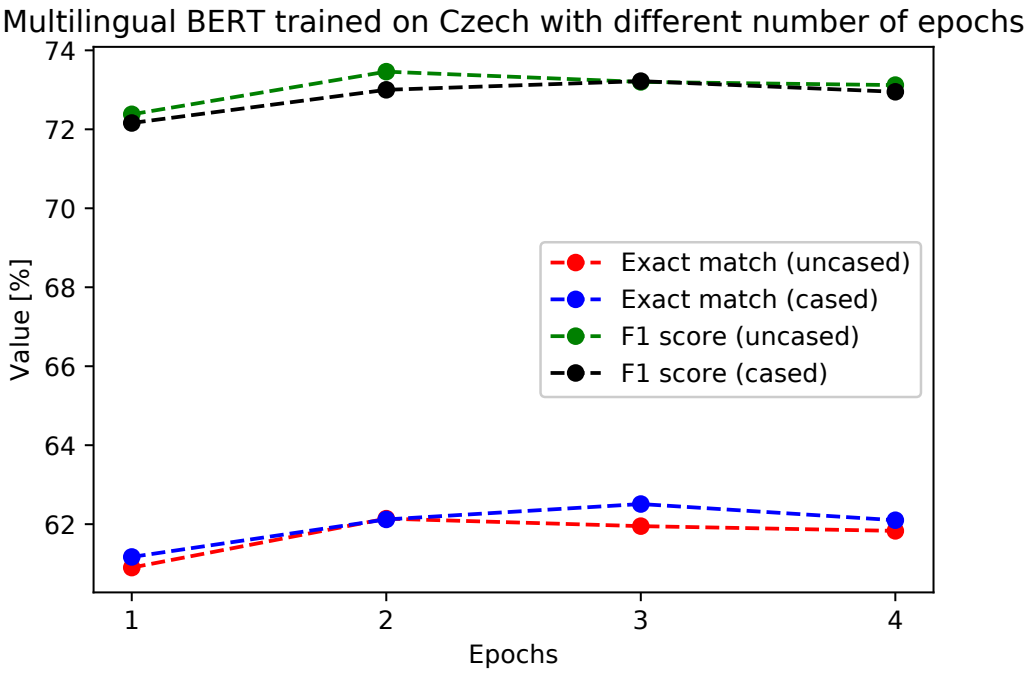


Figure 7.2: How much the number of epochs influence the exact match and F1 score during training of Multilingual BERT on Czech training dataset with match 100%.

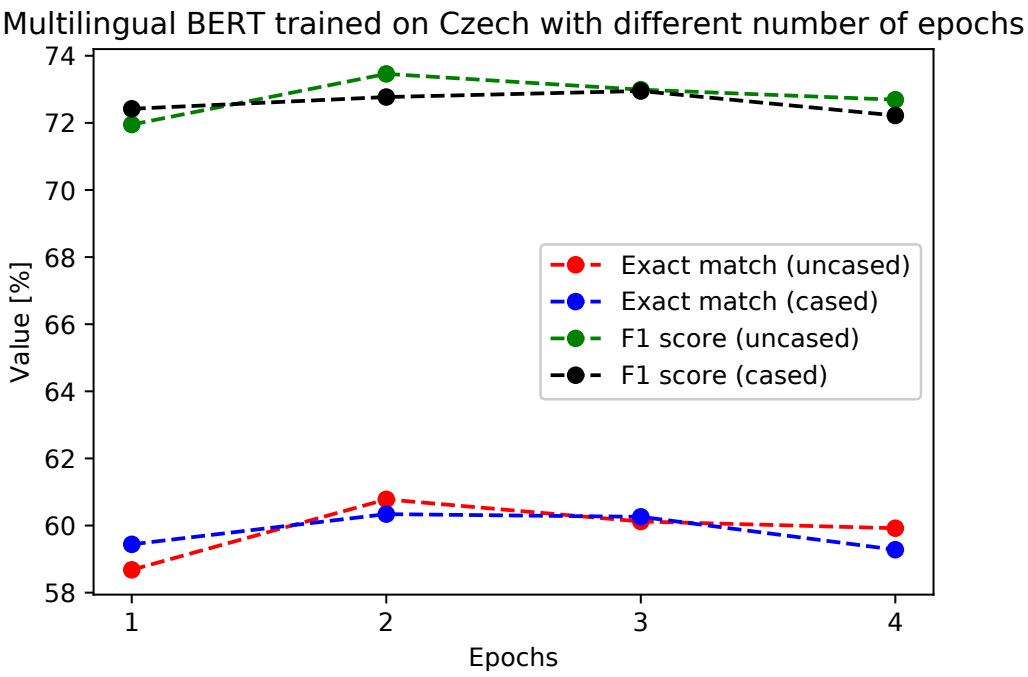


Figure 7.3: How much the number of epochs influence the exact match and F1 score during training of Multilingual BERT on Czech training dataset with match 80%.

We have also tried to train Multilingual BERT for English. Firstly, we have evaluated it on English to see how much different the multilingual model is from the original BERT. Then, we have evaluated the model on Czech by the same way as in BiDAF model. We will now revise the ways of the evaluation for better understanding.

- Dev EN We have evaluated the model on English test dataset.
- Dev CZ We have evaluated the model directly on Czech test dataset without passing any Czech data into the model before.
- Dev CZ-EN-CZ We have translated the Czech development dataset to English and then we have translated the English answers back to Czech and we have made evaluation on Czech dataset with comparison to the original Czech answers.

All of these evaluations were made on both cased and uncased models. We have described how we have evaluated the Multilingual BERT model trained on English and we can compare the results in the table, see Table 7.7.

Finally, we have done the same evaluation method also for original BERT trained on English just to compare the results, see Table 7.8.

| Multilingual BERT | Train | Dev | Exact match | F1 |
|---|---|---|---|---|
| Cased | EN | EN | 82.26% | 89.26% |
| Uncased | EN | EN | 81.86% | 89.12% |
| Cased | EN | CZ | 58.78% | 70.16% |
| Uncased | EN | CZ | 63.71% | 74.78% |
| Cased | EN | CZ-EN-CZ | 65.49% | 77.0% |
| Uncased | EN | CZ-EN-CZ | 65.17% | 77.0% |

Table 7.7: Comparison of results of Multilingual BERTs trained on English and evaluated on English and Czech.

| BERT | Train | Dev | Exact match | F1 |
|---|---|---|---|---|
| Cased | EN | EN | 80.81% | 88.27% |
| Uncased | EN | EN | 80.26% | 87.75% |
| Cased | EN | CZ | 12.95% | 24.93% |
| Uncased | EN | CZ | 6.74% | 21.20% |
| Cased | EN | CZ-EN-CZ | 64.93% | 76.63% |
| Uncased | EN | CZ-EN-CZ | 63.12% | 75.33% |

Table 7.8: Comparison of results of BERTs trained on English and evaluated on English and Czech.

### 7.2.1 Main Findings

We cannot exactly say whether a cased or an uncased model is better. The cased model is more accurate because it can distinguish words with diacritics and words starting with capital letters and it can preserve more information in data. On the other hand, the uncased model use smaller vocabulary and less data is necessary to train the model properly. It cannot be said which model should give us better results. Sometimes we have better cased model and sometimes the uncased model is better. For that, we always train and evaluate both versions.

Multilingual BERT for Czech is a bit worse than Multilingual BERT for English. It makes perfect sense as the Czech is much harder for BERT than English and there is a noise caused by translation in the Czech dataset.

Evaluation on Czech the dataset on BERT trained in English gives quite bad results. It is logical as BERT in only trained in English and has never seen any other language before. On the other hand, it is remarkable, that Czech evaluation on Multilingual BERT trained in English gives such a good results, in spite of the fact it has never seen any Czech question answering data before. From our

point of view, it is thanks to the fact that Multilingual BERT was trained on 100 different languages and that languages like Czech and English has several common features. When model sees some similarity between languages, it is trying to solve it by sharing one piece of whole network because it has limited capacity and it fits 100 languages into one network and it needs to do some optimization. To increase the efficiency, model is trying to merge similar features into one common module. Moreover, the embeddings for different languages after a simple linear transformation demonstrate a remarkable amount of similarity, as demonstrated by Artetxe et al. [2018], Mikel Artetxe and Agirre [2018], Sakuma and Yoshinaga [2019]. BERT is able to find this relations and use this space also for Czech language although, it was trained on English.

This paper Lewis et al. [2019] published in November 2019 has occupied with similar problem. The authors have also trained BERT model for question answering in 6 different languages but they have not tried Czech. They had confirmed our hypothesis and results that Multilingual BERT is such good even for other languages than English.

It is remarkable that CZ-EN-CZ models are noticeably better than CZ models. It is probably caused by te fact, that Czech morphology is richer than English one and therefore, training of CZ models need more data. Moreover, English models have more data for training as we have discarded some of the Czech data after translation because of the translation errors. To achieve the same accuracy, CZ models need bigger datasets. On the other hand, the biggest disadvantage of CZ-EN-CZ model is the necessity of trained translation system that needs huge dataset of parallel data and it is complicated to train such a model. In our opinion, the best model is trained on English and then evaluated on Czech as it does not need any translation in spite of the fact that it does not have the highest accuracy and F1 score.

### 7.2.2 Summary

We have observed that the best BERT model on Czech dataset is Multilingual BERT trained on English and evaluated by translating the Czech input into English and then translating the answer back to Czech. It has exact match 65.49% and F1 score 77.0%. If we want to avoid any translating, we have the best model Multilingual BERT trained on English and evaluated directly on Czech with 63.71% exact match and 74.78% F1 score. It does not have the highest score but it has one huge advantage – this model does not have any necessity of any data translation.

# 8. Conclusion

In this thesis, we have elaborated question answering in Czech without having any Czech data for training. We have translated English dataset SQuAD 1.1 to Czech to create Czech training and development datasets. We have tried several options for training the BiDirectional Attention Flow and the BERT model.

At first, we have trained and evaluated both of these models on Czech dataset. Then, we have trained them on English dataset and we have evaluated them on Czech dataset translated to English and we have translated the English answers from the model back to Czech. Finally, we have trained Multilingual BERT on English and we have evaluated on Czech dataset without any translation of the data.

To the conclusion, we have to compare BiDAF and BERT model results. In both of these we have observed that training in English gives better results than training in Czech.

Firstly, it is important to mention that BERT has reached much better results in English QA task than BiDAF, see Table 8.1.

| **Model** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| **BiDAF** | EN | EN | 64.22% | 75.19% |
| **BERT** | EN | EN | 80.81% | 88.27% |

Table 8.1: Comparison of results of BiDAF and BERT trained and evaluated on English SQuAD

Secondly, we can observe results of training of both models in Czech. We can see that BiDAF is better in training dataset of matching value 80% and BERT is better on training dataset of matching value 100%. If we compare the best results for both model for Czech, we can see that the best model Multilingual BERT is much better in exact match and in F1 score than the best model BiDAF.

Thirdly, we have compared models trained in English and evaluated on Czech dataset where we have translated the inputs into English, we have let the model to predict answer and then, we have translated the answer back to Czech and compared it with original answer. We can see the results in Table 8.2 and Figure 8.1.

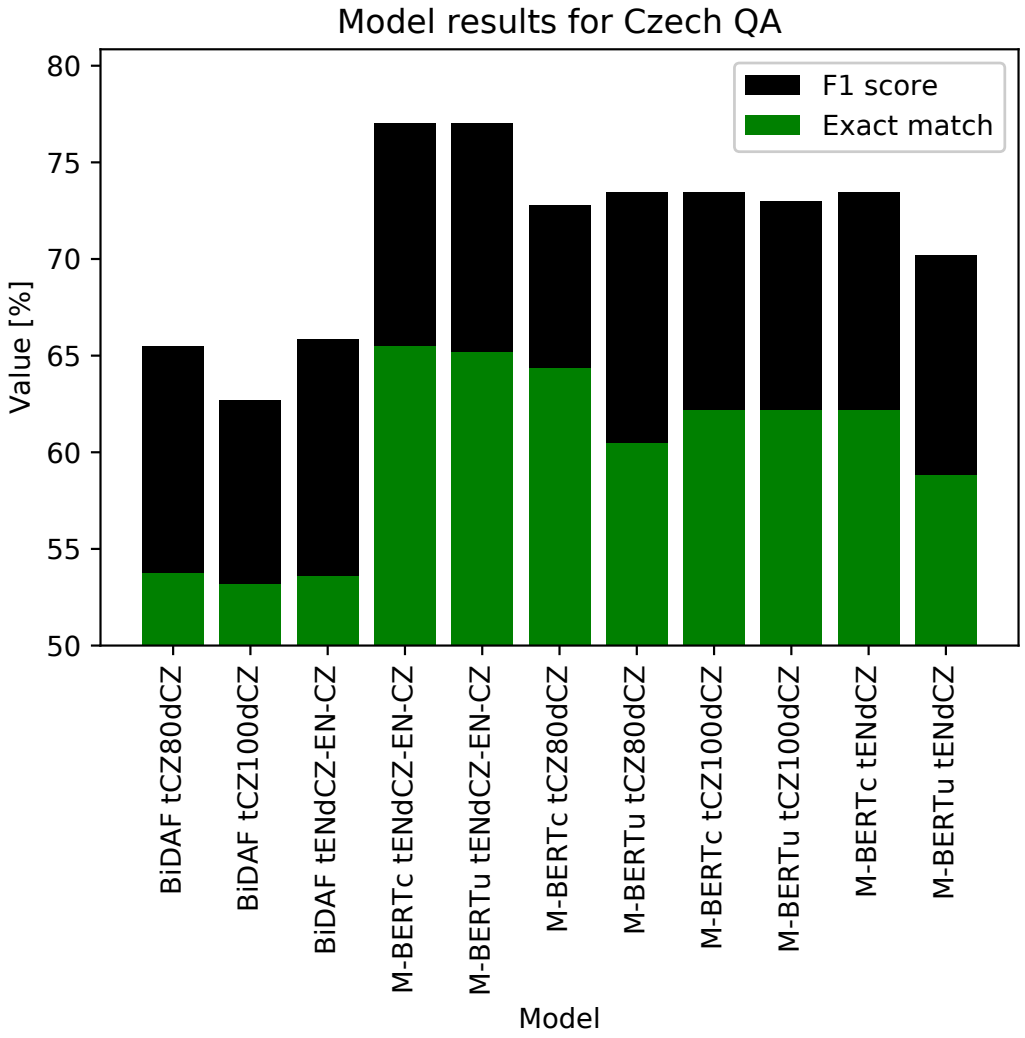


Figure 8.1: Comparison of exact match and F1 score for all models for Czech QA.

Finally, the most suitable model for Czech QA is Multilingual BERT uncased trained on English and evaluated on Czech that has reached 63.71% exact match 74.78% F1 score. The reason is that it not dependent on any translator as it does need to translate any data. It cannot be exactly compared with the best model for English QA as we have a bit different development dataset but we have included the results for the best model for QA in English just for orientation, see Table 8.3.

| **Model** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| BiDAF | CZ 80% | CZ 100% | 53.74% | 65.49% |
| BiDAF | CZ 100% | CZ 100% | 53.15% | 62.7% |
| BiDAF | EN | CZ-EN-CZ | 53.57% | 65.84% |
| Multi BERT cased | EN | CZ-EN-CZ | 65.49% | 77.0% |
| Multi BERT uncased | EN | CZ-EN-CZ | 65.17% | 77.0% |
| Multi BERT cased | CZ 80% | CZ 100% | 64.34% | 72.77% |
| Multi BERT uncased | CZ 80% | CZ 100% | 60.48% | 73.46% |
| Multi BERT cased | CZ 100% | CZ 100% | 62.16% | 73.0% |
| Multi BERT uncased | CZ 100% | CZ 100% | 62.14% | 73.46% |
| Multi BERT cased | EN | CZ | 58.78% | 70.16% |
| Multi BERT uncased | EN | CZ | 63.71% | 74.78% |
| BERT cased | EN | CZ | 12.95% | 24.93% |
| BERT uncased | EN | CZ | 6.74% | 21.20% |

Table 8.2: Comparison of all models for Czech QA evaluated after lemmatization.

| **Model** | **Train** | **Dev** | **Exact match** | **F1** |
|---|---|---|---|---|
| **BERT** | EN | EN | 80.81% | 88.27% |
| **Multi BERT** | EN | CZ-EN-CZ | 65.49% | 77.0% |
| **Multi BERT** | EN | CZ | 63.71% | 74.78% |
| **Multi BERT** | CZ | CZ | 62.14% | 73.46% |

Table 8.3: Comparison of best uncased models for English QA and Czech QA.

# Bibliography

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/viewPaper/16935.

Danqi Chen, Jason Bolton, and Christopher D. Manning. Thorough examination of the cnn/daily mail reading comprehension task. 2016. URL https://www.aclweb.org/anthology/P16-1223.

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

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. He goldilocks principle: Reading children's books with explicit memory representations. 2016. URL https://arxiv.org/pdf/1511.02301.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. Mlqa: Evaluating cross-lingual extractive question answering. 2019. URL https://arxiv.org/pdf/1910.07475.pdf.

Gorka Labaka Mikel Artetxe and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. *CoRR*, 2018. URL http://arxiv.org/abs/1805.06297.

Martin Popel. Cuni transformer neural mt system for wmt18. URL https://www.aclweb.org/anthology/W18-6424.pdf. [Accessed 2-November-2019].

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

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

M. Richardson, C.J.C.Burges, and Erin Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text.

*EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2013. URL https://www.researchgate.net/publication/286965813_MCTest_A_challenge_dataset_for_the_open-domain_machine_comprehension_of_text.

Jin Sakuma and Naoki Yoshinaga. Multilingual model using cross-task embedding projection. 2019. URL https://www.aclweb.org/anthology/K19-1003.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, 2016. URL http://arxiv.org/abs/1611.01603.

Jana Straková, Milan Straka, and Jan Hajič. Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. pages 13–18, 2014. URL http://www.aclweb.org/anthology/P/P14/P14-5003.pdf.

TREC. Question answering collections. 2019. URL https://trec.nist.gov/data/qa.html.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *CoRR*, 2016. URL https://arxiv.org/abs/1611.09830.

Wikipedia. Natural language processing — Wikipedia, the free encyclopedia. Online: https://en.wikipedia.org/wiki/Natural_language_processing, a. [Accessed 23-November-2019].

Wikipedia. Question answering — Wikipedia, the free encyclopedia. Online: https://en.wikipedia.org/wiki/Question_answering, b. [Accessed 23-November-2019].

Wikipedia. Reading comprehension — Wikipedia, the free encyclopedia. Online: https://en.wikipedia.org/wiki/Reading_comprehension, c. [Accessed 23-November-2019].

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, and etc. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, 2019. URL https://arxiv.org/abs/1910.03771.

Yi Yang, Wen tau Yih, and Christopher Meek. Wiki-qa: A change dataset for open-domain question answering. 2015. URL https://www.aclweb.org/anthology/D15-1237.

# List of Figures


3.1 An example showing the keyword selection and dependencies modeling between answer and question. Source Pranav Rajpurkar and Liang [2016]. . . . . 14
3.2 An example showing the computation of syntactic divergence between answer and question. Source Pranav Rajpurkar and Liang [2016]. . . . . 15

4.1 BiDirectional Attention Flow model. Source Seo et al. [2016] . . . 18

5.1 BERT input representation. Source Devlin et al. [2018] . . . . . 21
5.2 Overall pretraining and finetunning process for BERT. Source Devlin et al. [2018] . . . . . 21
5.3 Overall Transformer architecture. Source Radford et al. [2018] . . 22

6.1 Example of the selected answer by the algorithm with changed word order between text and answer. . . . . 29
6.2 Example of the selected answer by the algorithm with synonyms in text and answer. . . . . 29
6.3 Example of the selected answer by the algorithm with different declination in text and answer. . . . . 29
6.4 Example of the selected answer by the algorithm with non-translated numbers in text and answer. . . . . 29
6.5 Example of the selected answer by the algorithm with partially translated names in text and answer. . . . . 29
6.6 Plot of how much the answers match the answers in the text for the training set. . . . . 30
6.7 Plot of how much the answers match the answers in the text for the development set. . . . . 31

7.1 Plot of exact match how much the answers match the answers in the text for testing. . . . . 35
7.2 How much the number of epochs influence the exact match and F1 score during training of Multilingual BERT on Czech training dataset with match 100%. . . . . 38

7.3 How much the number of epochs influence the exact match and F1 score during training of Multilingual BERT on Czech training dataset with match 80%. . . . . 39

8.1 Comparison of exact match and F1 score for all models for Czech QA. . . . . 43

# List of Tables


5.1 Comparison of results of BERTs trained on English from Devlin et al. [2018]. . . . . 23

6.1 Preservation of original datasets, where the match is higher than the defined value. . . . . 31
6.2 Preservation of original datasets, where the match is higher than defined value. . . . . 32

7.1 Exact match after translation SQuAD to Czech and then training and testing models on data files with corresponding matching values. 34
7.2 F1 after translation SQuAD to Czech and then training and testing models on data files with corresponding matching values. . . . . 34
7.3 Results after translation of Czech input to English and then translation predicted English outputs back to Czech for evaluation. . . 35
7.4 Comparison of results from all models. . . . . 36
7.5 Comparison of results of BERT using different number of epochs. 37
7.6 Comparison of results of Multilingual BERT trained and evaluated on Czech using different training and testing sets. . . . . 37
7.7 Comparison of results of Multilingual BERTs trained on English and evaluated on English and Czech. . . . . 40
7.8 Comparison of results of BERTs trained on English and evaluated on English and Czech. . . . . 40

8.1 Comparison of results of BiDAF and BERT trained and evaluated on English SQuAD . . . . . 42
8.2 Comparison of all models for Czech QA evaluated after lemmatization. . . . . 44
8.3 Comparison of best uncased models for English QA and Czech QA. 44

# A. Overview of Electronic Attachments

Directory **Attachments**

- **Czech SQuAD** The file containing the SQuAD training and development datasets to Czech.
  - *dev_cz_80.json*
  - *dev_cz_100.json*
  - *train_cz_80.json*
  - *train_cz_100.json*
- **English SQuAD** The file containing the source SQuAD training and development datasets in English.
  - *dev-v1.1.json*
  - *train-v1.1.json*
- **Predictions** The file containing predicted Czech answers for all BERT models.
  - *predictions_cz80_multi_cased.json*
  - *predictions_cz80_multi_cased.json*
  - *predictions_cz100_multi_uncased.json*
  - *predictions_cz100_multi_uncased.json*
  - *predictions_en_bert_cased_cz.json*
  - *predictions_en_bert_uncased_czencz.json*
  - *predictions_en_bert_cased_cz.json*
  - *predictions_en_bert_uncased_czdencz.json*
  - *predictions_en_multi_cased_cz.json*
  - *predictions_en_multi_uncased_czdencz.json*
  - *predictions_en_multi_cased_cz.json*
  - *predictions_en_multi_uncased_czencz.json*
- **Scripts** The file containing translation, lemmatizaton and other scripts.

  - *compare_lcs_and_accord.py*
  - *create_html_visualization.py*
  - *lemmatize_dev.py*
  - *lemmatize_pred.py*
  - *select_data_above_threshold.py*
  - *translate_answers_EN-CZ.py*
  - *translate_dev_CZ-EN.py*
  - *translate_dev_EN-CZ.py*

- *evaluate-v1.1.py* Script for evaluation of exact match and F1 score of predicted answers.