DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2021

What the BERT?

Fine-tuning KB-BERT for Question Classification

JONATAN CERWALL

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

What the BERT?

Fine-tuning KB-BERT for Question Classification

Jonatan CERWALL

Master's Programme, Machine Learning, 120 credits
Date: June 22, 2021

Supervisor: Johan Boye
Examiner: Olov Engwall
School of Electrical Engineering and Computer Science
Swedish title: Vad i BERT?
Swedish subtitle: Finjustering av KB-BERT för frågeklassificering

Abstract

This work explores the capabilities of KB-BERT on the downstream task of Question Classification. The TREC data set for Question Classification with the Li and Roth taxonomy was translated to Swedish, by manually correcting the output of Google's Neural Machine Translation. 500 new data points were added. The fine-tuned model was compared with a similarly trained model based on Multilingual BERT, a human evaluation, and a simple rule-based baseline.

Out of the four methods of this work, the Swedish BERT model (*SwEAT-BERT*) performed the best, achieving 91.2% accuracy on TREC-50 and 96.2% accuracy on TREC-6. The performance of the human evaluation was worse than both BERT models, but doubt is cast on how fair this comparison is.

SwEAT-BERTs results are competitive even when compared to similar models based on English BERT. This furthers the notion that the only roadblock in training language models for smaller languages is the amount of readily available training data.

Keywords

BERT, KB-BERT, Question Classification, TREC, Li and Roth Taxonomy

Sammanfattning

Detta arbete utforskar hur bra den svenska BERT-modellen, KB-BERT, är på frågeklassificering. BERT är en transformermodell som skapar kontextuella, bidirektionella ordinbäddningar.

Det engelska datasetet för frågeklassificering, TREC, översattes till svenska och utökades med 500 nya datapunkter. Två BERT-modeller finjusterades på detta nya TREC-dataset, en baserad på KB-BERT och en baserad på Multilingual BERT, en flerspråkig variant av BERT tränad på data från 104 språk (däribland svenska). En regel-baserad modell byggdes som en nedre gräns på problemet, och en mänsklig klassificeringsstudie utfördes som jämförelse.

BERT-modellen baserad på KB-BERT (*SwEAT-BERT*) uppnådde 96.2% korrekthet på TREC med 6 kategorier, och 91.2% korrekthet på TREC med 50 kategorier. Den mänskliga klassificeringen uppnådde sämre resultat än båda BERT-modellerna, men det är tvivelaktigt hur rättvis denna jämförelse är.

SwEAT-BERT presterade bäst av metoderna som testades i denna studie, och konkurrenskraftigt i jämförelse med engelska BERT-modeller finjusterade på det engelska TREC-datasetet. Detta resultat stärker uppfattningen att tillgänglighet till träningsdata är det enda som står i vägen för starkare språkmodeller för mindre språk.

Nyckelord

BERT, KB-BERT, Frågeklassificering, TREC, Li och Roth Taxonomi

Acknowledgments

I would like to thank Johan Boye for his valuable insights during the course of the project.

Stockholm, June 2021
Jonatan Cerwall

Contents

Chapter 1

Introduction

Machine Learning (ML) in Natural Language Processing (NLP) have, in both popularity and size, exploded over the latest years. One reason for this explosion in size has been the sheer availability of text as a source of data. There is, however, a sizable discrepancy of readily available data when comparing between different languages. Most research in NLP is focused on English, and as such, most data sets are English. This poses a problem when developing models for smaller languages.

In 2021, there were 1.3 billion English speakers, both first language speakers and second language speakers. Contrast this to Swedish, which has 13 million speakers, both first and second [1]. Therein lies the challenge in building language models for Swedish.

The trend for NLP in recent years is to *pre-train* a model on an enormous corpus of general text data, and subsequently *fine-tune* the model on smaller, task-specific data sets. These tasks, as in the case of Sentiment Analysis (classifying sentences as positve or negative) and Question Answering (finding the answer to questions posed in natural language), can be quite different, but intuitively they do share some fundamental structure. This pre-training + fine-tuning is an example of transfer learning - learning how to solve one problem by using trained layers and weights from a related problem. The intuition is that the pre-trained model learns these fundamental structures, while the fine-tuned model learns to perform on the specific task. This has proven to work well in practice, but it is unclear how deep this understanding of language is [2].

One of these pre-trained models is called BERT (*Bidirectional Encoder Representations from Transformers*). Originally published in 2018 by Devlin et. al, BERT quickly established itself as a landmark architecture in the field.

Boasting state-of-the-art performance on many different benchmark tasks [3], it showed the power of pre-training + fine-tuning.

Since then, BERT has been the subject of many different extensions and alterations, and has been a popular architecture for smaller languages, since training BERT similarly to the original paper but using a native corpus has proven to outperform multilingual models. The National Library of Sweden introduced KB-BERT in 2020, a Swedish BERT, and even though much work has been done on English versions of BERT, much less has been done on just how much knowledge smaller language versions of BERT retains.

In addition to this English BERT, Devlin et. al. released a multilingual BERT. Built in the same way as the English one but trained on a multitude of language corpora, M-BERT has become a popular benchmark for smaller language BERTs to overcome.

This works aims to investigate the performance of KB-BERT in relation to BERT and M-BERT by exploring its effectiveness on the downstream task of Question Classification (QC).

QC is a type of text classification problem, which aims to predict the Expected-Answer-Type (EAT) of the question. For instance, for the question:

Var ligger Ishotellet?

we would want our model to predict the EAT to be a location. Intuitively, figuring out what *type* of answer a question is expecting is often the first thing humans do in order to accurately answer the question itself, and this has traditionally been the case for Question Answering (QA) systems as well. Recently, QA models have been favouring an end-to-end approach, leaving EATs somewhat behind.

Since Swedish is a much smaller language, any comparison between English BERT and KB-BERT necessitates translation of an English data set. Question Classification was chosen as the comparison downstream task since the most popular data set for QC is both widely used enough in order to have enough comparison models, while still small enough in order for it to be feasibly translated by one person.

By fine-tuning KB-BERT on a *translated* QC data set and comparing the performance to similar fine-tuned models of English BERT on the original English data set we can start to compare KB-BERT and BERT.

Problem

- How does a Swedish BERT model perform at the task of Question Classification compared to the original English BERT?
- How does a Swedish BERT model perform at the task of Question Classification compared to Multilingual BERT?

Contributions

This work provides several contributions. The first is the investigation of the performance of BERT for smaller languages (in this case Swedish) compared to English BERT. The second is the trained model itself, which can be used for inference. The third is the data set used for training and testing, which is a translated (and corrected) version of TREC-6 (and TREC-50).

Delimitations

There are multiple hyperparameters involved with training a model. This work does not explore this space in any great detail, but largely takes hyperparameter settings from other works.

Neither does it seek to find the best ML model for Swedish Question Classification on the TREC dataset, which would involve implementing and testing a multitude of different models.

Chapter 2

Background

BERT stands atop a mountain of innovations in NLP. In order to properly understand BERT, one needs to start at the bottom and work ones way up.

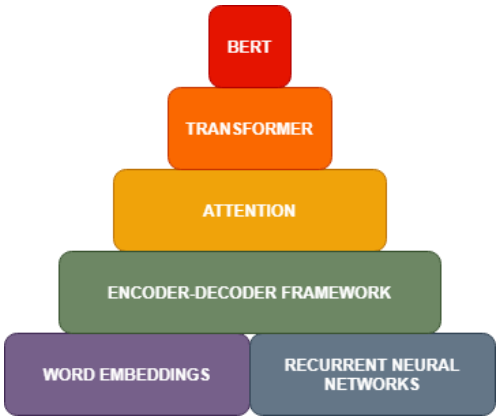

Figure 2.1: The mountain of BERT. Adapted from [4].

2.1 Word embeddings

Central to NLPs success in recent years is the word embedding. The idea of a word embedding is to represent words as vectors in multidimensional space, and in such a way that semantically similar words tend to group together. These embeddings can then be compared using a metric such as cosine similarity, which provides a way to measure semantic similarity between words. We will see that the trend of NLP research is to find ways to create better and better embeddings.

This concept has also found its way into multiple recommendation engines, as a way to measure and compute similarity between products or services [5].

2.1.1 Word2vec

Proposed in 2013 by Mikolov et. al., Word2vec is a word embedding built on the notion that "You shall know a word by the company it keeps", i.e. that words that tend to show up in similar contexts share semantic qualities [6]. For instance, consider the two sentences "the big blue bus", and "the big red bus". In this training corpus of only two sentences, the words "blue" and "red" share the same context, and should be considered semantically similar. Mikolov et. al. managed to create this embedding in two ways.

Continuous Bag of Words

The first way is by estimating the likelihood that a given word will appear in the context of the words surrounding it. This can be done in an efficient manner.

Consider once again the sentence "The big blue bus". If we set the size of the context to be three (one word before our target word and one word after) we would firstly increase the estimated likelihood of the word 'The' appearing in the context of the word 'big' (since 'The' is the first word in our corpus it lacks a previous context). We then slide our context window one step to the right, and increase the likelihood of the word 'big' appearing in the context of both the newly updated 'The' as well as the word 'blue'.

By sliding a context window like this over a corpus, the word embedding of each word gets continuously updated by the word embeddings surrounding it. The context window size n is a hyperparameter of the model ($n = 9$ in the original paper [6]).

Skip-gram

The second way is similar to the first way, but, in a sense, flipped. That is, estimating the likelihood that the surrounding words will appear in the context of a given word (see figure 2.2). The two methods produce slightly different results [6].

2.1.2 Results of Word2vec

The performance of the model on the training tasks is entirely disregarded. The result lies fully in the word embeddings, which exhibited some extraordinary

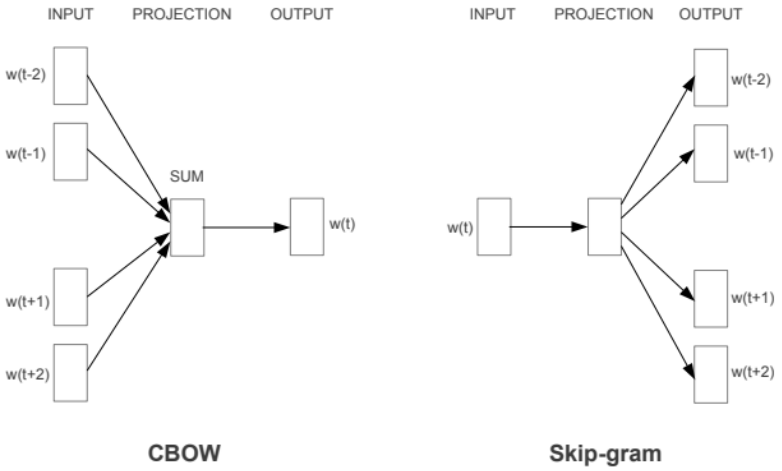

Figure 2.2: The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word. Derived from [6].

qualities. Most notably,

$$\text{king} - \text{man} + \text{woman} \approx \text{queen}$$

capturing the semantic quality that *king is to man what queen is to woman*. Capturing these sorts of semantic relationships (*a is to b what c is to d*) was consistent across multiple domains, and was a remarkable achievement. A PCA (Principal Component Analysis) projection of the vector representations of countries and their capital cities is presented in figure 2.3.

One major disadvantage of this type of embedding is that it is *context free*, meaning each word has one and only one embedding. This is a shortcoming, since words change based on context. For example, the word "bank" has widely different meanings in the context "river bank" as opposed to "bank account", but Word2vec has the same exact embedding for both.

2.2 Recurrent Neural Network

Another building block in the mountain of BERT is the Recurrent Neural Network (RNN). The term *recurrent* refers to the fact that these networks perform the same task over each instance of the sequence such that the output is dependent on the previous computations and results [8]. RNNs come in many forms, but consistent across them is the *hidden state*, meant to encode

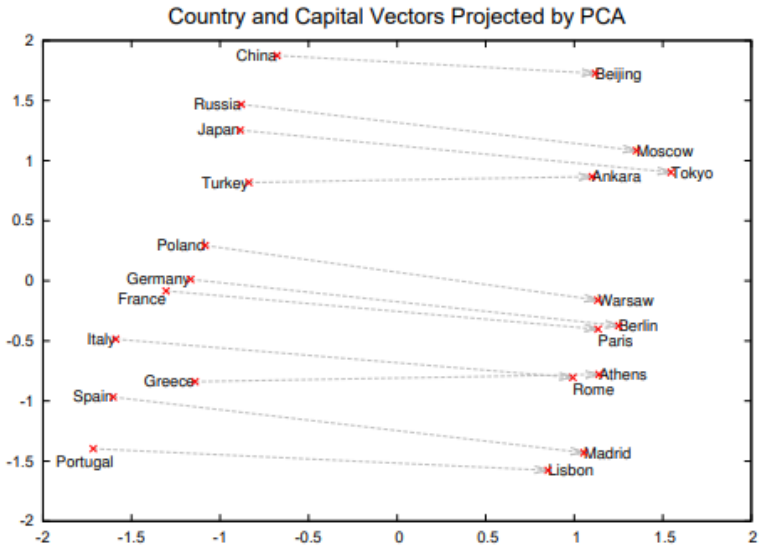

Figure 2.3: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. Derived from [7].

information about the timesteps the model has seen so far. This allows the model to keep a "memory" over the preceding sequence and use this information in future computations. The inherent sequential nature of the architecture made RNNs popular for NLP tasks [8], since natural language is inherently sequential.

Another factor in the popularity of RNNs in sequence modeling lies in its ability to model variable length of text, including very long sentences, paragraphs and even documents [9]. A shortcoming of this architecture, however, is picking up long-term dependencies [10]. For instance, consider the sentence "I grew up in France... I speak fluent *French*" in the context of a generative model. In order to properly predict and generate the last word as *French*, the model needs to keep track of information which could be arbitrarily far back in its memory. While in theory RNNs are capable of handling long-term dependencies, in practice they don't seem to [11].

Two note-worthy modifications to the original RNN architecture are Long Short Term Memory networks (LSTM) [11] and Gated Recurrent Units (GRU) [12]. They utilize special memory gates in order to solve the problem with long-term dependencies.

2.3 Encoder-Decoder frameworks

Originally proposed in 2014 by Cho et. al. (the same paper that introduced GRUs), the Encoder-Decoder framework consisted of two RNNs connected to each other [12]. The first one, dubbed the encoder, encodes a sequence of symbols to into a fixed-length vector representation, while the second one, the decoder, decodes this vector representation into another sequence of symbols. This strategy has been successfully employed in various sequence-to-sequence (seq2seq) tasks, most notably in Neural Machine Translation (NMT) where it was deployed to Google translate in 2016 [13]. The intuition for this Encoder-Decoder model builds upon the intuition for word embeddings, where the fixed length vector representation acts as an *contextual* embedding of the entire sequence.

This fixed length vector representation, or context vector, is typically the last hidden state of the first RNN. The output of the first RNN is typically discarded. The second RNN, the decoder, takes this hidden state as input in order to produce a new sequence.

A disadvantage of the embeddings created by this encoder-decoder framework is that they are *unidirectional*, meaning that the embedding of a certain word only depend on the words preceding it. This is a shortcoming, since words change meaning based on the succeeding words. For example, in the sentence "I accessed the bank account", the word "bank" would in a unidirectional model only be dependant on the words "I accessed the", potentially creating an wrongful embedding of "bank".

2.4 Attention

The fixed length vector representation of the encoder-decoder framework tended to be a bottleneck in terms of dealing with long sentences [14]. A solution was proposed in 2014 by Bahdanau et. al. [14], and refined in 2015 by Loung et. al. [15] called "Attention". Attention extends the encoder-decoder framework in two main ways.

First, instead of only sending one context vector from the encoder to the decoder, the last hidden state, the encoder saves a copy of all its hidden states, and sends all of them to the decoder.

Second, for each word, the decoder gives a scoring for each of the hidden states provided, indicating how much "attention" should be given to that particular hidden state for this particular word. The scoring is softmaxed,

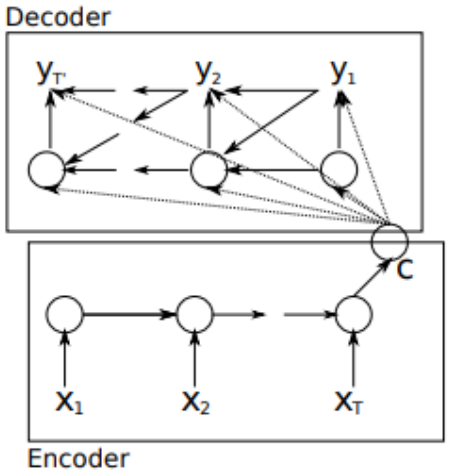

Figure 2.4: Illustration of the Encoder-Decoder framework. The dashed lines represent the dependencies on the context vector, on which each decoder hidden state and each generated symbol depends. Derived from [12].

amplifying hidden states with high scores and drowning out hidden states with low scores. The scoring then acts as the weights in what becomes a weighted average of each hidden state. This results in a new, weighted, hidden state, that allows the decoder to more accurately distinguish between features of the encoded sentence. The scoring is done by a feedforward neural network trained jointly with the model.

The idea of attention proved to be very effective, with Loung et. al. gaining new State-of-the-art performance on neural translation benchmarks [15]. The power of this technique shall not be understated. Consider the English sentence

> *"The agreement on the European Economic Area was signed in August 1992."*

and its French counterpart

> *"L'accord sur la zone économique européenne a été signé en août 1992."*

taken from the aforementioned papers. Notice that the ordering of words in the English phrase "European Economic Area" is flipped when translated to French, becoming "zone économique européenne". This is a common

occurrence in translation, and with attention this is beautifully handled. In figure 2.5 the attention in this example is visualized. In this figure it is clear that attention provides a way of dealing with different word orderings.

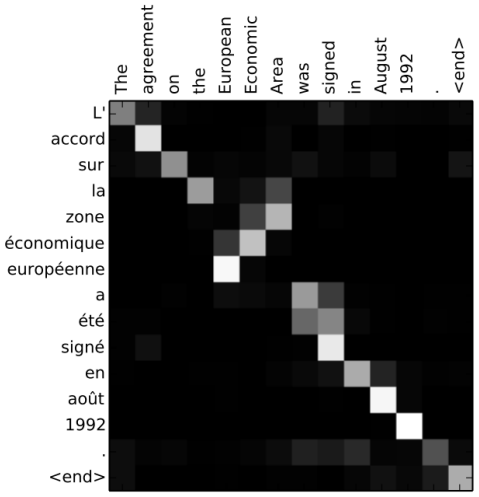

Figure 2.5: Visualisation of attention in English-French translation. Derived from [14].

2.5 Transformers

A major breakthrough in NLP comes in the form of Transformers. Introduced in the paper "Attention is all you need" by Vaswani et. al., 2017 [16], Transformers builds upon both the Attention mechanism introduced in the previous section as well as the encoder-decoder framework, but, crucially, drops the recurrent neural network, relying entirely on the attention mechanism to draw dependencies between input and output.

Vaswanis transformer consisted of six encoder blocks and six decoder blocks. Each block is identical but does not share weights. The number six seems to be somewhat arbitrary (later architectures have pushed the limits on this number, and it seems that, so far, bigger is better [18]). Each encoder block consists of a self-attention layer and a feed forward layer. As we will see, self-attention can be calculated for all words in the input sequence in parallel, breaking from the sequential nature of computations of the RNN, which allowed the transformer state-of-the-art performance while requiring

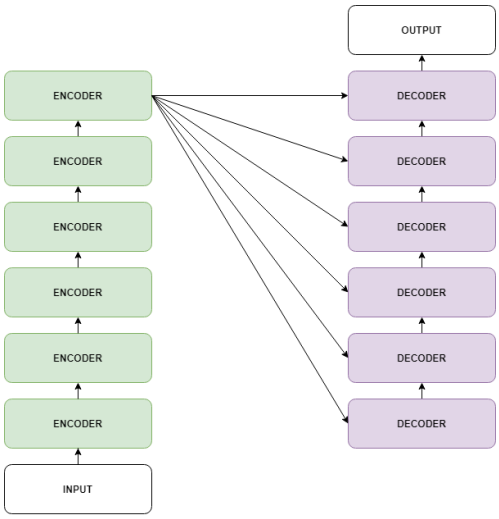

Figure 2.6: High-level view of the Transformer. Adapted from [17].

significantly less time to train [16]. A high-level depiction of the transformer is presented in figure 2.6.

2.5.1 Self-attention and bidirectionality

The attention mechanism in the transformer is slightly different than the one employed by Bahdanau et. al., dubbed "Self-attention". The idea is the same, but self-attention captures how an input sequence is attending to itself. For instance, take the sentence

The animal didn't cross the street because it was too tired.

What does *it* refer to? For a human, this is obvious, but for a machine it is not. The question of how different words in a sequence relate to one another is what self-attention seeks to answer. The claim by Vaswani et. al. is that the way words relate to one another is the only quality of natural language that needs capturing in order to facilitate good performance on NLP tasks (in this case machine translation).

A major benefit of self-attention is that it is inherently bidirectional. This allows the transformer to represent words using both its previous and next context. Contrast the previous example sentence "The animal didn't cross the street because it was too tired" with the sentence

The animal didn't cross the street because it was too wide.

With a change of just one word, the word "it" now refers to the street as opposed to the animal. Since self-attention is bidirectional, this change of meaning of the word "it" can be captured. A visualisation of self-attention is presented in figure 2.7.

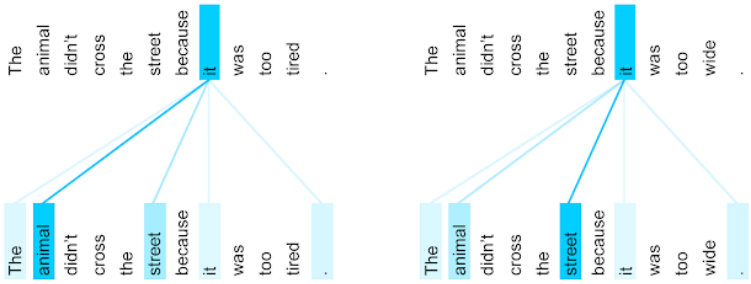

Figure 2.7: Illustration of self-attention. Note the difference in the self-attention of the word "it" between the two sentences. Derived from [19].

2.5.2 Self-attention in detail

In self-attention, similarly to Bahdanau's attention, a scoring is calculated between each word in the input sentence (including between a word and itself!). Since the transformer does away with hidden states, which in Bahdanau's attention are used in calculating scores, self-attention accomplishes the scoring by creating three vectors for each word, dubbed the query vector, the key vector, and the value vector. To illustrate self-attention, consider calculating the self-attention of the word 'I' in the sentence "I am beautiful".

The first step in calculating the self-attention is to create the three vectors for each word. This is done by multiplying the encoding of each word (in subsequent blocks this is the output of the previous block) with three weight matrices, W_q, W_k, and W_v, whose weights are learned during training.

The second step is to take the dot product of the query vector of 'I' with the key vector of all the words in the sentence, i.e. 'I', 'am', and 'beautiful'. This produces a number, a scoring of each word.

The third step is to softmax this scoring, amplifying high scores and drowning out low scores.

The fourth step is to multiply the value vector of all three words with this scoring, and then summing them up. This step creates an embedding of the

word 'I' using the softmaxed score of each word as weights in a weighted average.

To reiterate, the query and key vectors are used to calculate the scores, which are used as weights in a weighted average of the value vector of all words. This means that the embedding of 'I' is influenced by all words in the sentence, but is influenced by some words more than other. It is *attending* to some words more than others.

Dot-Product Attention

Self-attention is computed separately for each word, but can in computation be packed together to form efficient matrix multiplications [16]. This creates the attention function used by Vaswani et. al., Dot-Product Attention*:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T)V \tag{2.1}$$

where Q, K, and V are query-, key-, and value matrices respectively. The output is a contextual, bidirectional, word embedding of each word.

2.5.3 Multi-head attention

Self-attention is done 8 times in parallel (attention with 8 *heads*). This is done by linearly projecting (multiplying with weight matrices) the value-, key- and query matrices by 8 different learned linear projections. The output of these 8 distinct self-attention functions is then concatenated, and fed through yet another linear projection back to the original dimensions. Multi-head attention is a way of combining different attention functions, and since everything is trained together, the different attention heads end up complementing each other [16]. A visualisation of Multi-head attention is presented in figure 2.8.

2.6 BERT

BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers) is a Transformer model based on Vaswani's original transformer. There are two major differences.

The first major difference is that BERT only consists of the encoder part of the encoder-decoder framework. The second major difference is in the way BERT was trained. Vaswani's transformer was trained on a translation

* In practice, the scoring was scaled by a factor $\frac{1}{\sqrt{d_k}}$, (Scaled Dot-Product Attention) since the authors found that this led to more stable gradients [16]. It is a technical detail that matters quite little to the overall mechanism of self-attention, so this was omitted.

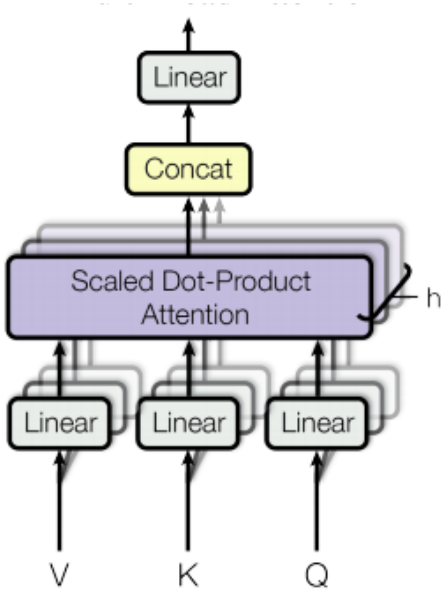

Figure 2.8: Multi-Head Attention. The value-, key- and query matrices are linearly projected (multiplied with trained weight matrices) in order to facilitate multiple distinct attention heads. $h = 8$ in the original paper. Derived from [16].

task (English-German and English-French), while BERT was trained on the two tasks Masked Language Modeling (MLM) and Next Sentence Prediction (NSP).

With these changes, BERT makes a split between creating embeddings of natural language and executing on natural language tasks. This split is put concretely in BERT's two-step framework *pre-training*, and *fine-tuning*.

BERT was initially released in two sizes, BERT-Base and BERT-Large. BERT-Base had 12 Attention layers and 110M parameters, while BERT-Large had 16 Attention layers and 340M parameters [3].

2.6.1 Pre-training and fine-tuning

During pre-training, the model is trained on unlabeled data over the different pre-training tasks. For fine-tuning, the BERT model is first initialized with its pre-training parameters and then its architecture is modified for the downstream task. This modification usually only involves adding a single layer in order to plug in task-specific inputs and outputs into BERT. All parameters of BERT are

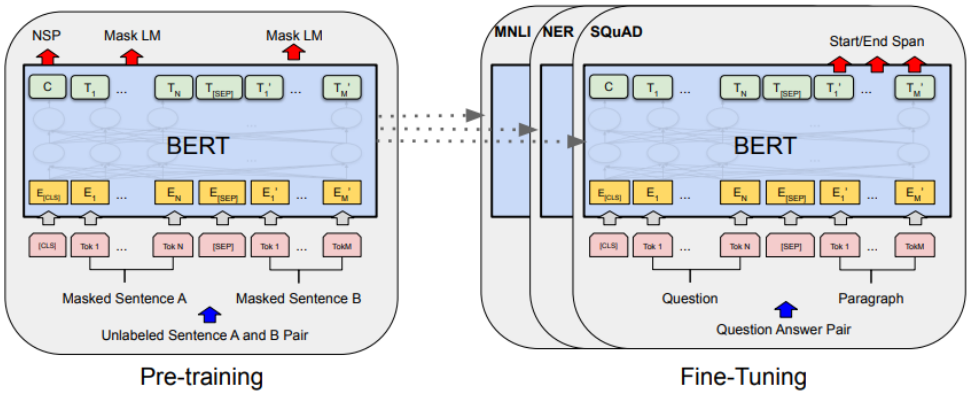

Figure 2.9: Pre-training and fine-tuning procedures for BERT. Derived from [3].

then fine-tuned (trained some more) end-to-end [3]. The big advantage of this approach lies in the training time of the fine-tuning procedure, which at most takes a few hours on a GPU. Contrast this with the training time of pre-training, which took 4 days on multiple TPUs (64 TPUs for BERT-Large) [3]. This split of pre-training vs fine-tuning is sometimes referred to as *deep transfer learning*. An illustration of BERT's pre-training + fine-tuning approach is presented in figure 2.9.

The result of the pre-training is the embeddings, and the performance of the model on the two tasks, MLM and NSP, is entirely disregarded. Notice the similarity between BERT and Word2vec in this aspect. While the embeddings of both BERT and Word2vec are bidirectional, BERT's embeddings are also contextual.

BERT was pre-trained on two corpora, totaling about 3,300M words (800M from BooksCorpus, 2,500M from English Wikipedia) [3].

2.6.2 Pre-training

BERT was pre-trained using two different pre-training tasks.

Masked Language modeling

In Masked Language Modeling (MLM), one word in each sentence is masked by replacing it with a [MASK] token, and the task is predicting it. It is similar to regular language modeling (predicting the next word based on the previous words) but crucially this allows the model to fuse left and right context,

Input	The man went to the $[\text{MASK}]_1$. He bought a $[\text{MASK}]_2$ of milk
Labels	$[\text{MASK}]_1$ = store; $[\text{MASK}]_2$ = gallon

Table 2.1: Example of Masked Language Modeling

Sentence A	The man went to the store.
Sentence B	He bought a gallon of milk.
Label	IsNextSentence
Sentence A	The man went to the store.
Sentence B	Penguins are flightless.
Label	NotNextSentence

Table 2.2: Example of Next Sentence Prediction

allowing for bidirectionality [3]. An example of MLM is presented in table 2.1.

Masking words like this comes with a downside, however, creating a mismatch between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning. To mitigate this, BERT does not always replace the "masked" word with the actual [MASK] token. BERT replaces the "masked" word with (1) the [MASK] token 80% of the time (2) a random word 10% of the time and (3) does not change the word at all 10% of the time [3]. For instance, the sentence "my dog is hairy" would 80% of the time appear as "my dog is [MASK]", 10% of the time as "my dog is apple" (where the word 'apple' is randomly sampled) and 10% of the time as "my dog is hairy".

Another downside of MLM lies in its poor data efficiency. Of all input tokens in the MLM pre-training task, only 15% of them were masked, and subsequently predicted. Contrast this to regular language modeling where 100% of the input tokens are predicted. This leads to an increase in the time it takes the model to converge, but only slightly [3].

Next Sentence Prediction

The second task that BERT was pre-trained on is Next Sentence Prediction (NSP). Two sentences were fed to BERT and the task is to determine if the second sentence follows the first sentence in the training corpus. This is important in tasks such as Question Answering (QA) and Natural Language Inference (NLI) whose success depends on understanding the *relationship* between two sentences [3]. An example of NSP is presented in table 2.2.

2.6.3 Input/Output representations

In order to make BERT handle a variety of downstream tasks, the input representation is handled in a very specific manner.

BERT needs to be able to handle multiple sentences, and differentiate between them (as in NSP, QA, and NLI). This is done in two ways. Firstly, BERT separates the two sentences with a special separation token ([SEP]). Secondly, BERT adds a segment encoding to every token indicating whether it belongs to sentence A or sentence B.

The first token of every input sequence to BERT must be the special classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks [3].

Self-attention in a transformer model is calculated for each word in a sequence simultaneously, but natural language is highly sequential. This means that BERT needs some way of encoding the sequential nature of a sentence. This is done by a positional encoding (Vaswani et. al. handled this problem similarly).

The input embeddings are the sum of the token encoding, the segmentation encodings and the positional encodings as shown in figure 2.10. Observe that the input embeddings is a separate embedding from the contextual bidirectional embeddings that BERT creates.

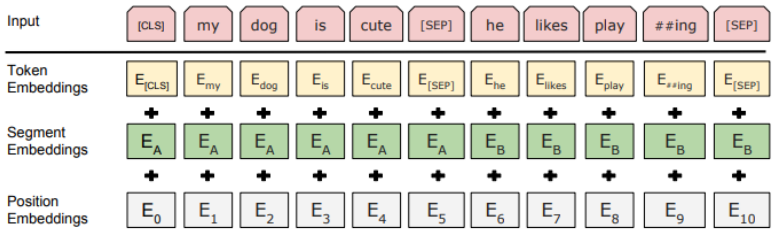

Figure 2.10: BERT input representation. Be wary of the term 'embedding' as it appears describing input representations, as this embedding is separate from the embeddings BERT creates. Derived from [3].

Since BERT processes each word in an input sequence in parallel, it requires each sentence to be of the same length. This is achieved by setting a max length, and either padding shorter sentences with the special pad token ([PAD]) or truncating longer sentences.

Corpus	Million words
Digitized newspapers	2 997M
Government reports	117M
Legal e-deposits	62M
Social media	31M
Swedish Wikipedia	29M
Total	3 497M

Table 2.3: Data sources for KB-BERT. Adapted from [25].

2.7 BERT for other languages

In addition to BERT model trained on English text, Devlin et. al. also released a multilingual BERT (M-BERT). M-BERT is built in the same way as BERT, but simply trained on a multitude of language corpora (104 languages, including Swedish). M-BERT demonstrated strong, sometimes state-of-the-art, performance on cross-lingual tasks [20, 21].

Because of its success, BERT has been adapted to multiple languages. These include (among others) French (CamemBERT [22]), Italian (AlBERTo [23]), Dutch (BERTje [24]), and Swedish (KB-BERT [25]). In general, it seems that native language BERTs outperform multilingual ones on native language tasks [22–29]. Which is hardly surprising, but still provides clear evidence and incentive to build more native language BERTs.

2.7.1 KB-BERT

The main challenge of building large language models for smaller languages is the lack of readily available training data. In building KB-BERT, the Royal Library of Sweden attacked this problem by gathering data from a variety of sources, and cleaning the data from these sources with various custom scripts and solutions. A breakdown of the sources used is presented in table 2.3. The total words in the pre-training corpus is actually greater than that for English BERT, which is impressive.

KB-BERT was built with the same architecture as BERT-Base and trained with the same hyper-parameter settings [25].

Coarse	Fine
ABBR	abbreviation, expansion
DESC	definition, description, manner, reason
ENTY	animal, body, color, creation, currency, disease, event, food, instrument, language, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word
HUM	description, group, individual, title
LOC	city, country, mountain, other, state
NUM	code, count, date, distance, money, order, other, percent, period, speed, temperature, size, weight

Table 2.4: The Li and Roth taxonomy

2.8 Question Classification

Question Classification (QC) is a subcategory of text classification. The task is to assign one or more class labels to questions written in natural language by the *type* of answer that the questions expect. For instance, "Where is the Eiffel Tower?" would ideally be assigned the label of "Location", since the question expects a location. The task of Question Classification is also referred to as predicting the *Expected-Answer-Type* (EAT). The set of question categories (classes) is usually referred to as *question taxonomy* or *question ontology* [30].

The data set commonly used in QC is the TREC data set*, introduced by Li and Roth (2002) [31], combined with the Li and Roth taxonomy. It is also commonly referred to as both the UIUC data set and the QAQC data set. The data set is fairly small, consisting of 5,500 training questions and 500 test questions. The taxonomy is hierarchical with 6 *coarse grained* classes (TREC-6) and 50 *fine grained* classes (TREC-50), giving each data point two classes (one fine grained and one coarse grained). The hierarchical nature of the taxonomy creates two distinct classification tasks, fine grained classification and coarse grained classification. The taxonomy is presented in table 2.4.

The distribution of classes in the TREC data set is skewed. The raw occurrences of coarse grained classes is presented in figure 2.11. The distribution of fine grained classes is presented in table A.1.

* https://cogcomp.seas.upenn.edu/Data/QA/QC/

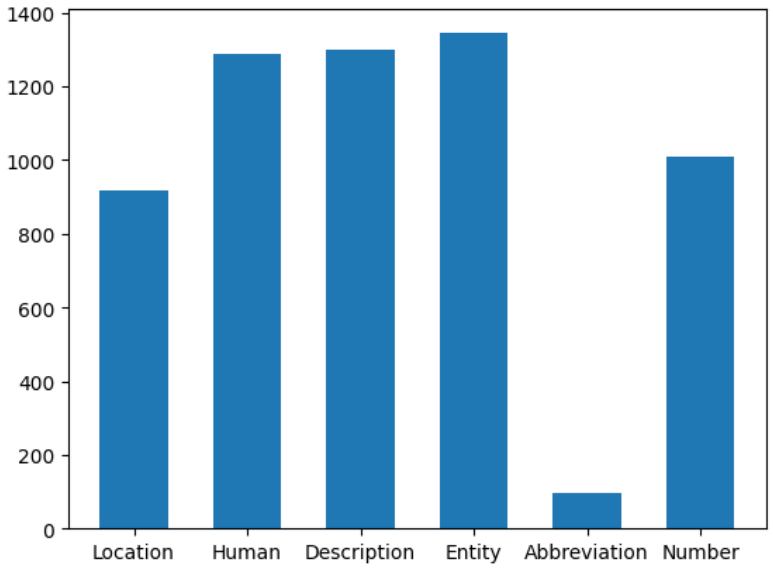

Figure 2.11: Occurrences of coarse grained classes in the TREC data set.

2.8.1 The ambiguity problem

One problem when creating a labeled data set for classifying questions by their EATs is the inherent ambiguity that a lot of questions exhibit. For instance, the question

What do bats eat?

could belong to **plant**, **animal**, or **food** [31]. Similarly, the question

What is bipolar disorder?

could belong to either **definition** or **disease**.

To combat this problem, Li and Roth allowed their classifiers to label questions with multiple labels [31], but each question in the data set was still given only one label. This approach might introduce problems with training, since some questions are not treated as positive examples for possible classes as they should be [31].

2.8.2 Models for Question Classification

There are three types of models which do well in question classification. Those that rely purely on rules, those that make use of machine learning and those that are a hybrid of the two.

Rule-based methods are concerned with creating hand-crafted classification rules. For example, it feels quite intuitive that a question which starts with the word 'Who' is very likely to be about a human individual. It is easy to see that this can be extended. It is not clear, however, how far it can be extended, or how easy it is to craft more 'general' rules which apply to all kinds of questions [30, 32]. It is very possible to achieve high accuracy with pure rule-based methods on TREC-50, with Madabushi et. al. achieving 97.2% [33].

Machine learning based methods vary. Success has been found with SVMs [34, 35], Advanced Kernel Methods [36, 37], and Maximum Entropy methods [34, 38]. as well as Attention-based-LSTMs [39] and BERT-based models [40, 41].

Hybrid methods usually involve building features from pre-defined rules and subsequently training a model with these features [34, 35].

2.9 Related work

A couple of English BERTs for QC have been built and tested. Most notably Xu et. al. [40], which experimented with BERT for QC on a number of different data sets, concluding that BERT facilitated good performance with a single model over multiple domains. Sun et. al. [41] experimented with different fine-tuning procedures for text classification, concluding that further in-domain-pre-training (IDPT) was beneficial to performance, culminating in BERT-IDPT-FiT.

Experiments on the capabilities of KB-BERT is sparse. Appelstål, 2020 [42] fine-tuned KB-BERT for text classification in a multimodal model, and Tengvall, 2020 [43] fine-tuned KB-BERT for Question Answering.

Chapter 3

Method

3.1 The data

The data set used in this study was the TREC data set for question classification. It was chosen for two reasons. The first reason is that TREC is a fairly widely used data set, allowing for meaningful comparisons. The second reason is that TREC is quite small, with about 6k data points (5500 for training and 500 for testing). The small size of the data set meant that it was feasible to translate it.

3.1.1 Translating the data set

The biggest hurdle in training language models for smaller languages is the lack of readily available data. Translating is a necessary step, and will be for the near future. Manually translating is an enormously time-consuming task, however, so some form of automation had to be employed. The data set was fed through Google's neural machine translation model [13] (Google Translate) via the Google translate API. This is not (yet) good enough on its own [44], however, so after this step each entry in the data set was manually corrected. The guideline for this correcting was to try to change as little as possible, while making certain that the translated sentences referred to the same thing that the original sentences did. For instance, the question,

What should the oven be set at for baking Peachy Oat Muffins?

was originally translated to by the NMT system as

Vad ska man ställa in ugnen för att baka Peachy Oat Muffins?

which without context seems to ask for a **location** (LOC) to put the oven in, or perhaps an **entity** (ENTY) to put in the oven. Since the original question wants a **number** (NUM) to set the oven at, this was corrected to

Vad ska man ställa in ugnen på för att baka Peachy Oat Muffins?

These types of mistakes were not the most common (the most common errors were encoding-related, double quotes '"' became '''), but they were the most important ones to get right since if these went unchanged the data set would suddenly have a lot more mislabeled data. The labels were not translated.

The manual correcting of the data set was done by one person over the course of two weeks. Some amount of translation errors is therefore to be expected.

3.1.2 Adding to the data set

In addition to translating TREC, 500 new data points were added. They were provided by KTH from a data set of Swedish reading comprehension questions. These were manually labeled with the Li and Roth taxonomy, and added to the training set of the translated TREC. The distribution of labels of these added data points were heavily skewed, however, with a majority of data points belonging to **description**. The raw occurrences of the added data points are presented in figure 3.1 and figure 3.2.

3.2 Models and methods

Two BERT-based models were built and fine-tuned, one based on KB-BERT and one based on M-BERT. In addition, a baseline method was constructed and a human evaluation study was performed, as comparisons.

3.2.1 Baseline method

In order to understand how hard the problem of question classification on TREC is, a baseline method was implemented. It was designed to be as simple as possible. By seeing how well a very simple model does, one can more accurately gauge how difficult the problem is. It does not tell the whole story however, as even if a very simple model can get reasonable high accuracy it is still very difficult to gauge how difficult the remaining percentage points are to get. It does however give a glimpse.

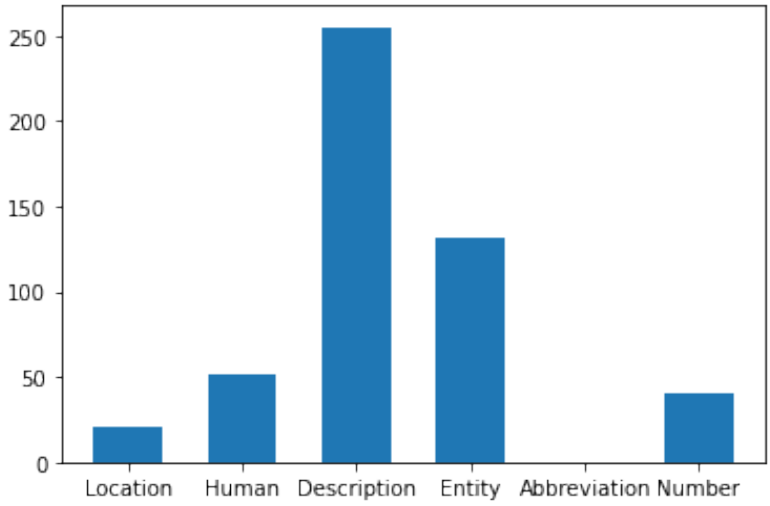

Figure 3.1: Occurrences of coarse grained labels for the added data points.

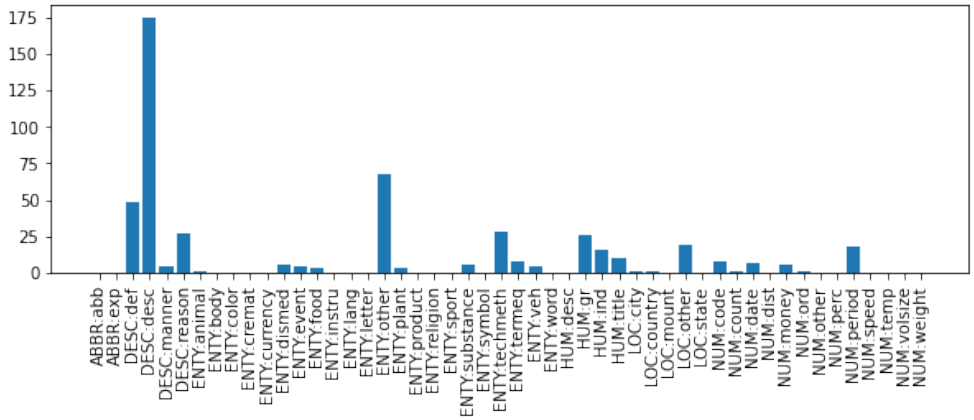

Figure 3.2: Occurrences of fine grained labels for the added data points. Notice that the majority of added data points were labeled as **description**.

The baseline method is rule-based, since the best performing non-ML solutions on TREC are rule-based. It simply treats the question as a bag-of-words (disregarding the order in which they appear), and looks for *wh*-words (*why, which, who, how*). When one such word is found, it is given a pre-defined label. If more than one *wh*-word is found, it is labeled at random between the pre-defined labels of the *wh*-words in the question. Nine *wh*-words are supported. If none of these are found in the bag-of-words, the question is

Wh-word	Coarse label	Fine Label
Hur	DESC	manner
Vad	DESC	definition
Vem	HUM	individual
Varför	DESC	reason
När	NUM	date
Vilka	HUM	group
Vilket	ENTY	ENTY:other
Vilken	ENTY	ENTY:other
Var	LOC	LOC:other
Default	DESC	description

Table 3.1: Baseline method

given a default label.

By building a simple baseline method we set a lower bound on the problem. The questions this baseline method tries to answer are *How hard is question classification in general? How much space for improvement is left?*. The rules of the baseline method is presented in table 3.1.

3.2.2 Human evaluation

Since this data set is small (only 500 data points in the test set) it was feasible to perform a human evaluation study. This was only done on TREC-6, because presenting 50 different choices per question quickly becomes tedious, and difficult to find evaluators for.

The test set was divided into ten chunks of 50 questions each, and each chunk was sent to a different human participant as a Google form. The participants was told to classify questions as to what feels most correct, relying on human intuition. The full questionnaire is included in appendix C.

3.2.3 SwEAT-BERT

SwEAT-BERT (Swedish-EAT-BERT) was built and trained in accordance with the original BERT paper.

Feeding BERT input

BERT needs the input in a special way. The first token of every sequence is always the special classification token ([CLS]), and each pair of sentences fed to BERT needs to be separated by the special separation token ([SEP]). Each

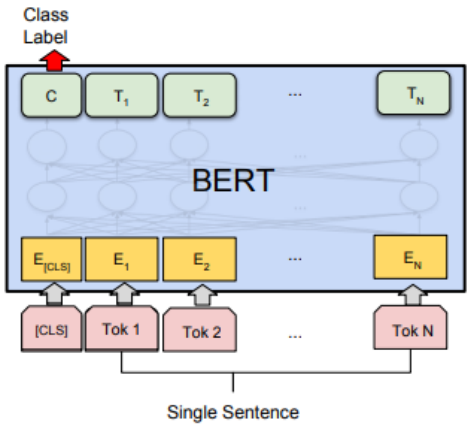

Figure 3.3: Architecture for fine-tuning BERT for sequence classification. Derived from [3].

Sentence	Vem anses vara den största tillverkaren av fioler?
Tokens	'[CLS]', 'Vem', 'anses', 'vara', 'den', 'största', 'tillverkaren', 'av', 'fiol', '##er', '?', [SEP], [PAD]
Token Indexing	[2], [3755], [3618], [358], [97], [1424], [37769], [65], [23521], [6], [302], [3], [0]

Table 3.2: Example of BERTs tokenization.

input sequence needs to be the same length, so each sentence was padded to the length of the longest sentence.

For this task, the input to BERT is only one sentence, which put the [SEP] token at the end right before the padding. An example of how the input was tokenized is presented in table 3.2. The tokenization was performed with the help of the tokenizer accompanying KB-BERT [25], hosted by Huggingface.

Adding classification layer

The building of *SwEAT-BERT* was heavily inspired by BertForMultipleChoice, hosted by Huggingface [45]. It uses the final hidden vector corresponding to the special classification token ([CLS]) as the aggregate representation in accordance with the original BERT paper [3]. The only new parameters added are classification layer weights $W \in \mathbb{R}^K$, where K is the number of labels. This meant that since the two tasks in the TREC data set have different number of labels, two different *SwEAT-BERT*s were built, *SwEAT-BERT-50* and *SwEAT-BERT-6*. It is possible to avoid this, by only building *SwEAT-*

BERT-50 and setting its coarse-level prediction as the category in which its fine-level prediction is contained, but fine-tuning is so quick that building two models did not present any additional challenge. Both models are identical apart from K, and will for the remainder of this thesis be treated as the same model. A standard cross-entropy loss was computed.

Fine-tuning and hyperparameters

SwEAT-BERT was fine-tuned for 4 epochs with a batch size of 16. It was optimized with Adam [46] and a linear scheduler, with an initial learning rate of 5e-5 and zero warmup steps. No real hyperparameter search was done, but a couple of training runs was done on a validation set, and the hyperparameters of the best performing one (so-called babysitting) was used when trained on the full training set and tested on the test set. No changes were made to the model after testing on the test set. No additional models were trained after testing on the test set. Training took roughly 10 minutes on a cloud GPU.

3.2.4 M-BERT for Question Classification

For comparison, a QC-model based on M-BERT was built and trained in the same way as *SwEAT-BERT*. This model will be referred to as *mEAT-BERT*.

Chapter 4

Results

The results of this research is presented in table 4.1. *SwEAT-BERT* outperforms *mEAT-BERT* on Swedish TREC-50 and outperforms both *mEAT-BERT* and the human evaluation on Swedish TREC-6. The performance gained from a fine-tuning KB-BERT versus fine-tuning M-BERT is in line with previous studies comparing the two [25], and closely resembles the gap native language models seem to gain over multilingual ones.

A comparison between *SwEAT-BERT* and current state-of-the-art performance by non-rule-based methods on English TREC-6 is presented in table 4.2. While this work and the models presented in table 4.2 are not tested on the exact same data set, the comparison still provides valuable insights. *SwEAT-BERT* is 1.6 percentage points worse than the best performing BERT model on TREC-6 (BERT-IDPT-FiT [41]), and 0.8 percentage points worse than the best performing BERT model on TREC-50 (BERT-QC [40]). It is very close in performance to BERT-QC, which is built and trained similarly.

Method	TREC-50	TREC-6
Rule-based baseline	45.8	53.8
mEAT-BERT	84.2	93.8
SwEAT-BERT	**91.2**	**96.2**
Human evaluation	/	84.6

Table 4.1: Accuracy score on TREC-50 and TREC-6

Model	TREC-50	TREC-6
SNoW (Li and Roth 2002)	84.2	91.0
Att-LSTM (Xia et al. 2018)	/	**98.0**
BERT-QC (Xu et al. 2019)	**92.0**	96.2
BERT-IDPT-FiT (Sun et al. 2019)	/	97.8
SwEAT-BERT (This work)	91.2	96.2

Table 4.2: Accuracy score on TREC-50 and TREC-6

Label	Precision	Recall	Support
Location	94%	95%	81
Human	98%	98%	65
Description	98%	97%	138
Entity	94%	93%	94
Abbreviation	100%	100%	9
Number	96%	97%	113

Table 4.3: Detailed results of *SwEAT-BERT* on TREC-6.

4.1 SwEAT-BERT

Looking more closely at the results of *SwEAT-BERT*, it seems that the model is quite robust. On TREC-6 the model seems to perform equally well on all labels, regardless of the amount of support. The more detailed results of *SwEAT-BERT* on TREC-6 is presented in table 4.3. The full confusion matrix is presented in figure 4.1. Of the 500 questions in the test set, only 19 questions were wrongly classified. These questions, along with the predicted label and the correct label, are presented in appendix B.

4.2 Qualitative analysis

Some of *SwEAT-BERTs* misclassifications can be explained by translation errors. For instance, the question "What city's newspaper is called 'the Star'?" is quite obviously referring to the city, while the translation "Vilken stadstidning heter 'The Star'?" is referring to the newspaper itself. This led both BERT models, as well as the human participant, to classify it erroneously. The same could be said for a number of questions ("fault line" became "fellinjen" etc).

Other misclassifications can be seen as ambiguous, where the BERT models predictions doesn't seem too off. One example of this is "Vad är

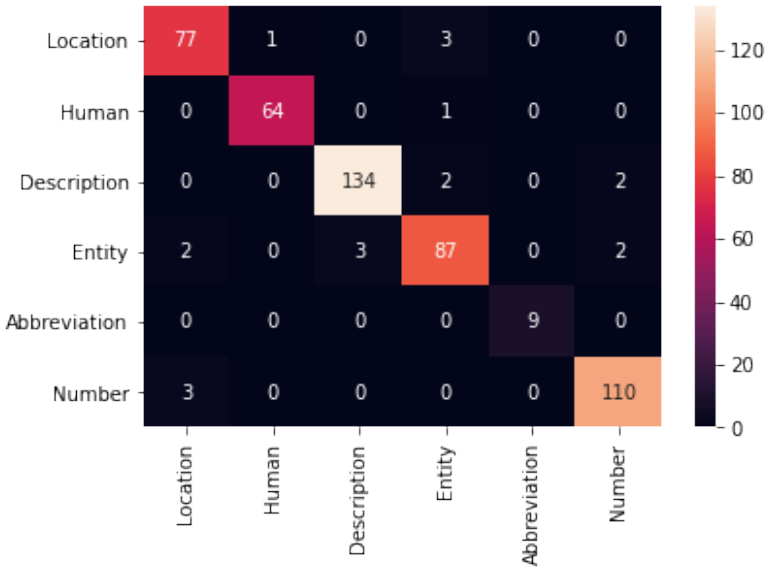

Figure 4.1: Confusion matrix of *SwEAT-BERT*s predictions on TREC-6. The predicted labels are on the vertical axis and the true labels are on the horizontal axis

momsen i Minnesota?" (eng. "What is the sales tax in Minnesota?") where the true label is **entity**, but both *SwEAT-BERT* and the human participant labeled it as a **number**, and *mEAT-BERT* labeled it as a **description**.

A few misclassifications do stand out however, where *SwEAT-BERT* seem to "misunderstand" [47] the question. These are the two questions "Vad är koppars smältpunkt?" and "Vad är guldets smältpunkt?" (eng. "What is the melting point of copper?" and "What is the melting point of gold?"). The true label for both of these are **numeric**, but *SwEAT-BERT* labeles these both as a **location**. Since "punkt" translates to "point", it seems as though *SwEAT-BERT* places too much emphasis on one type of meaning of "punkt". This seems rather strange, since the contextual nature of BERTs embeddings should be able to handle this change of meaning. Nothing in the tokenization of these sentences illustrates why this mistake occurs, and *SwEAT-BERT* is very confident in its predictions in both cases. While it is possible to dig deeper into the attention heads in order to gain some insight, there are in total 124 attention functions, combined in non-linear fashion, and as such this was beyond the scope of this study. Problems with attention as explanation is discussed in section 5.4.

Since rule-based methods on TREC have achieved high accuracy scores, it

is natural to wonder how much of this type of approach *SwEAT-BERT* utilizes. While making definitive statements of a transformer models approach is hard, this doesn't seem to be the case. Consider the two sentences "Vad är alla hjärtans dag?" and "Vad är ryggradslösa djur?" (eng. "What is Valentine's Day"? and "What are invertebrates?"). Both of these are correctly classified as **description** by the simple rule-based baseline, but *SwEAT-BERT* classifies the first as a **number**, probably recognizing that Valentine's Day is a date, and the second one as **entity**, probably recognizing that invertebrates are animals. A lot of questions in the training data that start with "Vad är..." have the label **description**, which makes this divergence from the statistics of the training set notable. *mEAT-BERT* classifies these two examples exactly as *SwEAT-BERT*.

That said, there are instances where *SwEAT-BERT* seems to fall back on this somewhat 'safe' approach. In the question "Vad är Ohio State Bird?" (eng. "What is the Ohio state bird?") the capitalization of "Ohio State Bird" resulting from a translation error seems to trip both BERT models up. In this instance, both *SwEAT-BERT* and *mEAT-BERT* classifies this as **description**, which might suggest that the BERT models have a sort of 'backup plan' if they don't recognize some of the words. And indeed, *SwEAT-BERT* labels the question "Vad är dfkjghkdjfghk?" as **description**.

Chapter 5

Discussion

5.1 Performance

The results of *SwEAT-BERT* are encouraging. They further strengthen the notion that native language BERTs outperforms multilingual ones on native language tasks. Furthermore, since *SwEAT-BERTs* performance in comparison to English state-of-the-art models is competitive, it suggests that architectures that perform well on English benchmarks is applicable to Swedish, given Swedish training data.

5.1.1 Comparing performance across different data sets

The QAQC data set and the Swedish QAQC data set are two different data sets. Almost no data point looks the same, and it could be argued that the two are vastly dissimilar. They overlap only in structure and labeling scheme. Still, given perfect translation, it does not feel unreasonable to argue that they are the same. Given perfect translation it is not unreasonable to argue that the comparison across different languages is perfectly valid.

That said, the translation was not perfect. I am not a linguist or a translator, I am merely a speaker of both languages. The inevitable errors in translation should negatively affect the comparability between models trained on the two data sets.

5.2 Human evaluation

While there is a sizable performance discrepancy between the BERT models and the human evaluation, this does not necessarily mean that the BERT models are better. This is due to two factors, the ambiguity of the task as well as the non-intuitiveness of the taxonomy.

5.2.1 Ambiguity

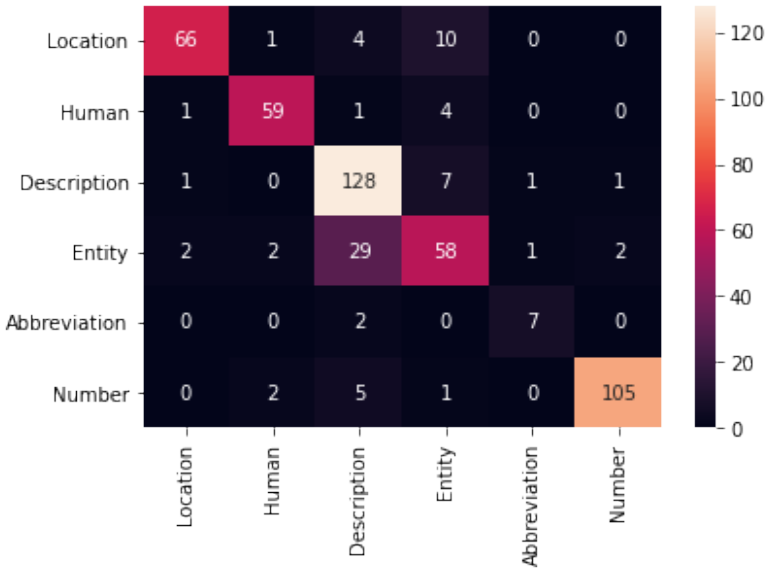

Figure 5.1: Confusion matrix of the human participants. The predicted labels are on the vertical axis and the true labels are on the horizontal axis

The problem is ambiguous, but the labels are set. This leads to an interesting dilemma, since an answer is not necessarily wrong just because is it wrong. Lets look at a key example. Consider the following question,

Vad är elproduktionen i Madrid, Spanien?

which translates to "What is the electrical output in Madrid, Spain?". This is a very ambiguous question (even *SwEAT-BERT* got it wrong). It could be asking for the amount of output, in which case the label should be **number**. On the other hand, it could be asking for a description of the electrical output, or a definition of what the electrical output in Madrid is, in which case the label

should be **description**. It could also be asking for the type of electrical output Madrid produces, in which case the label could be considered as being **entity**. No one answer is objectively correct, but they vary in terms of probability. For this question, the human participant labeled it as a **description**, but the 'correct' label was **entity**. For reference, I would have labeled this question as **number**.

This example clearly indicates the problem of ambiguity in question classification. It also indicates the problem of manual labeling, at times inaccurate, at times ambiguous.

In figure 5.1 the confusion matrix of the human evaluation is presented. Most errors occur when the human participant would label something as **description**, when the 'correct' label would be **entity**, as in the example above.

5.2.2 Non-intuitiveness

The majority of error's made by the human participants seem to stem from not knowing the taxonomy. Consider table 5.1

Question	*Hur snabbt är ljusets hastighet?*
Translation	*How fast is the speed of light?*
Predicted label	**description**
True label	**number**

Table 5.1: Human classification error

which at first glance seem to be a question of ambiguity. The answer to this question could be a **number**, but could also want a description of the speed, i.e. *It's the speed limit of the universe*. However, this interpretation becomes increasingly unlikely when one considers that **speed** is a subcategory of **number** in the Li and Roth taxonomy. Since the participants were not given the full 50 classes, this information could not have been known. On the other hand, a trained model would easily be able to catch the fact that every example in the training set involving **speed** would have **number** as the label.

The issue at play here is the fact that humans are asked to do a task that is not intuitive, with little explanation and too few examples. Classifying questions in categories based on the type of answer that question expects is not something that people usually do. Even typing it out is awkward and clumsy. However, giving human participants the full 50 classes and longer explanation would create a higher threshold for participating, which for this thesis was not possible.

5.2.3 Purpose of human evaluation

With all this in mind, the fact that the trained models outperform humans should not come as a surprise. The task is not a particularly human task, and thus outperforming the human evaluation becomes less impressive. Indeed, Li and Roth's model from 2002 performs better [31]. Still, it presents a valuable comparison, and clearly shows the usefulness machine learning provides in the QC space.

The real human benchmark would be trained participants, knowing the full taxonomy. This was however both out of scope and out of budget for this thesis, and is left for future work.

5.3 Complexity of the problem

While performing significantly worse than any other method, the simple rule-based baseline still performs significantly better than random (completely random is 2% on TREC-50 and 16.7% TREC-6). This suggests that much can be gained in terms of raw accuracy by applying simple classification rules. It also gives a hint as to why rule-based methods perform so well on QC problems. It remains unclear how well rule-based methods generalize to bigger and more varied data, however.

The small difference between the simple rule-based methods performance on TREC-50 and TREC-6 suggests that the complexity of the problem is not linear with the amount of classes. It also suggests that it is possible to achieve high accuracy scores by *gaming* the data, as it were. Note that the simple method can only choose 9 of the 50 classes as its classification, but still boasts close to 50% accuracy on TREC-50. The choice of which 9 classes to be available as options was made with the prevalence of the classes in the data set in mind.

It is very possible that a deep neural network would pick up on this skewed class distribution and use that knowledge for inference.

5.4 Attention analysis

It is possible, with the help of BertViz [48], to visualise the attention of a BERT model. While this is interesting, and clearly a major improvement in transparency over previous architectures, it is difficult to find intuitive meaning in the visualisation. Compare figure 5.2 with figure 5.3, both

visualizing *SwEAT-BERTs* attention for the sentence "*Vem anses vara den största tillverkaren av fioler?*" which translates to "*Who is considered the largest manufacturer of violins?*".

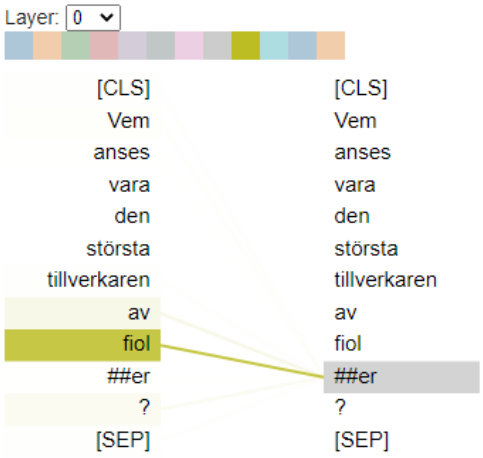

Figure 5.2: Cherrypicked example of *SwEAT-BERTs* attention

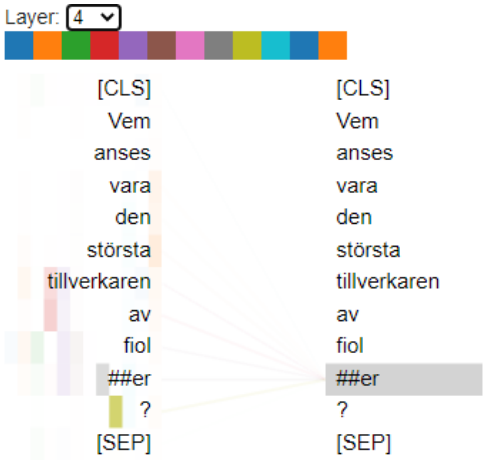

Figure 5.3: Random example of *SwEAT-BERTs* attention

Figure 5.2 was created with the intent on finding an attention head that made sense. It seems intuitive that the token '##er' (*fioler* is the multiple form of *fiol*) should attend to the word which it modifies, but looking at what the token '##er' attends to at a different layer, with multiple attention heads, as in figure 5.3, things quickly become less clear.

The question of whether attention can be seen as an explanation for a model's prediction is a debated one, with compelling arguments from both sides [49–51].

5.5 Ethical analysis

As machine learning systems grow ever more present in daily life and begins to influence society, ethical examination of these systems are in order [52]. Two types of ethical considerations are discussed, social bias within the model as well as ethical use of the model.

5.5.1 Social bias within the model

Social bias in the data sets which models are trained on will inevitably seep into the model itself. This thesis only touches the subject lightly by examining a sample of the Swedish TREC data set on one axis only, representation of women versus representation of men.

By extracting questions containing forms of "kvinna" or "tjej" ("woman" or "girl") and forms of "man" or "kille" ("man" or "boy") the differences between these types of sentences could be examined. Only a sample of all gender-specific questions was examined, and further research on the gender bias (as well as other biases) is needed. The regular expression used to extract these samples are included in D.

Among the gender specific questions sampled, it seems that the questions concerning women are more focused on sex and on the body than the questions concerning men. The questions concerning men, on the other hand, seem to be more focused on specific job titles. While this should not be taken as objective analysis, it is hardly surprising that this would be the case since the data set was constructed by two men in 2002.

Despite this, *SwEAT-BERT* seems quite robust against such bias. It assigned the same label for both "*Vad gör kvinnor om dagarna*" (eng. "*What does women do during the day?*") and "*Vad gör män om dagarna*" (eng. "*What does men do during the day*"). This seems to be due to the fact that *SwEAT-BERT* makes its decisions based very little on the specifics of the question, and very much on the structure. It assigned the same label to "*Vad gör dfkgjhd om dagarna?*" as it did the previous questions. It would be wrong to assume that SwEAT-BERT is impervious to bias, however, since almost no trained model is [52]. The nature of *SwEAT-BERTs* social bias is left for future work.

5.5.2 Ethical use of *SwEAT-BERT*

The method of using a question classification system built on KB-BERT will be used in a research project at KTH to generate reading comprehension questions. It might also be used in some QA systems. Since these types of systems is generative, it is very possible for them to exhibit unethical traits. One can imagine a QA system infused with political propaganda, or a reading comprehension generation system that is biased. This is inevitably always going to be a possibility in generative models.

The fear of misuse of generative models is growing with the effectiveness of these models. The full GPT-2 [53] model was held from the public for fear of misuse, but was later released in full. GPT-3 [18] has a wait list and a vetting process in order to get access to it. While these strategies make sense for big, general, powerful, generative models, it makes a lot less sense for *SwEAT-BERT*. These types of questions should nevertheless be raised.

5.6 Sustainability

This section covers three different aspects of sustainability with regards to this work.

5.6.1 Ecological

Training big language models uses a lot of power, and as such emit a sizeable amount of CO_2. Some estimates place the ecological impact of pre-training BERT at roughly the same as a trans-American flight [54]. Although this estimation is reliant on carbon emission data from GPUs (since the same data from TPUs were unknown), it highlights that there is a real environmental cost to training big language models.

While emissions of training BERT are high, the flexibility of the pre-training + fine-tuning framework does provide some advantages over similar, more narrow, models. Fine-tuning, while not entirely harmless, emits substantially less that pre-training. Being able to pre-train BERT once, and then use one pre-trained model across multiple tasks eliminates the need for training new big language models for each task separately, and should be seen as a net positive.

Research into the properties and performance of BERT models *could* be argued to be more sustainable than research into other similar models, but the fact remains that it will impact the environment either way.

5.6.2 Economical

In the cases where a question classification model is needed, *SwEAT-BERT* is superior to manual labour in both accuracy and speed. However, these cases are probably few and far between. The real economical advantage lies in more general language models.

Having access to a strong language model like BERT is beneficial for any language, and carries economical implications. Writing is an essential part of a lot of different jobs (writing e-mails is possibly the biggest contributor). Being able to automate some parts of the writing process leads to more time for other things, with increased productivity as the result.

5.6.3 Social

As the field of NLP progresses, it is possible that support for smaller languages become smaller and smaller. Since performance is so dependant on the amount of training data, and less data is naturally produced of smaller languages compared to bigger ones, this scenario seems increasingly likely.

With this backdrop, investigations into NLP tools for smaller languages is almost a must in order for smaller languages to keep up.

While training language models requires a large amount of training data, training any kind of model requires a large amount of compute. In a social aspect, this means that only the biggest companies can create the biggest ML models, which creates a quite worrisome disparity, especially since building bigger models seems to invariably yield better performance [18].

Chapter 6

Conclusions

I can conclude that given similar conditions, BERT seems to perform similarly on question classification for both English and Swedish. There doesn't seem to be any extra difficulty for transformer-style models to learn Swedish over English, which suggests that data availability is the only road-block in NLP research for smaller languages.

Native language BERTs still outperform multilingual BERT on native language tasks.

6.1 Future work

While it is possible to extend *SwEAT-BERT*, by either pre-training the model further or by applying one of many proposed alterations (e.g. mixout [55]), future work should probably take the form of comparing KB-BERT to BERT across a multitude of different downstream tasks. This study on its own, having compared the two on one task only, is too small of a sample size to draw any definitive conclusions.

References

[1] G. F. S. Eberhard, David M. and C. D. F. (eds), "Ethnologue: Languages of the world. twenty-fourth edition. dallas, texas: Sil international," 2021. [Online]. Available: http://www.ethnologue.com.

[2] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 2021.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] C. McCormick. (2019) Bert research - ep. 1 - key concepts & sources [blog post]. [Online]. Available: https://mccormickml.com/2019/11/11/bert-research-ep-1-key-concepts-and-sources/

[5] J. Alammar. (2019) The illustrated word2vec [blog post]. [Online]. Available: https://jalammar.github.io/illustrated-word2vec/

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[7] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *arXiv preprint arXiv:1310.4546*, 2013.

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Computational intelligenCe magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[9] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015*

conference on empirical methods in natural language processing, 2015, pp. 1422–1432.

[10] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.

[13] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2016.

[15] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[17] J. Alammar. (2018) The illustrated transformer [blog post]. [Online]. Available: https://jalammar.github.io/illustrated-transformer/

[18] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[19] J. Uszkoreit. (2017) Transformer: A novel neural network architecture for language understanding. [Online]. Available: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

[20] S. Wu and M. Dredze, "Beto, bentz, becas: The surprising cross-lingual effectiveness of bert," *arXiv preprint arXiv:1904.09077*, 2019.

[21] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual BERT?" in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019. doi: 10.18653/v1/P19-1493 pp. 4996–5001. [Online]. Available: https://www.aclweb.org/anthology/P19-1493

[22] L. Martin, B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, de la Clergerie, D. Seddah, and B. Sagot, "Camembert: a tasty french language model," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. doi: 10.18653/v1/2020.acl-main.645. [Online]. Available: http://dx.doi.org/10.18653/v1/2020.acl-main.645

[23] M. Polignano, P. Basile, M. de Gemmis, G. Semeraro, and V. Basile, "AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets," in *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, vol. 2481. CEUR, 2019. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85074851349&partnerID=40&md5=7abed946e06f76b3825ae5e294ffac14

[24] W. de Vries, A. van Cranenburgh, A. Bisazza, T. Caselli, G. van Noord, and M. Nissim, "Bertje: A dutch bert model," 2019.

[25] M. Malmsten, L. Börjeson, and C. Haffenden, "Playing with words at the national library of sweden–making a swedish bert," *arXiv preprint arXiv:2007.01658*, 2020.

[26] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang, and J. Pérez, "Spanish pre-trained bert model and evaluation data," in *PML4DC at ICLR 2020*, 2020.

[27] B. Chan, S. Schweter, and T. Möller, "German's next language model," 2020.

[28] Y. Kuratov and M. Arkhipov, "Adaptation of deep bidirectional multilingual transformers for russian language," 2019.

[29] A. Virtanen, J. Kanerva, R. Ilo, J. Luoma, J. Luotolahti, T. Salakoski, F. Ginter, and S. Pyysalo, "Multilingual is not enough: Bert for finnish," 2019.

[30] B. Loni, "A survey of state-of-the-art methods on question classification," 2011.

[31] X. Li and D. Roth, "Learning question classifiers," in *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

[32] X. Li, D. Roth, and K. Small, "The role of semantic information in learning question classifiers," in *Proceedings of the International Joint Conference on Natural Language Processing*. Citeseer, 2004.

[33] H. T. Madabushi and M. Lee, "High accuracy rule-based question classification using question syntax and semantics," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 1220–1230.

[34] Z. Huang, M. Thint, and Z. Qin, "Question classification using head words and their hypernyms," in *Proceedings of the 2008 Conference on empirical methods in natural language processing*, 2008, pp. 927–936.

[35] J. Silva, L. Coheur, A. C. Mendes, and A. Wichert, "From symbolic to sub-symbolic information in question classification," *Artificial Intelligence Review*, vol. 35, no. 2, pp. 137–154, 2011.

[36] Y. Pan, Y. Tang, L. Lin, and Y. Luo, "Question classification with semantic tree kernel," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 837–838.

[37] D. Tomas and C. Giuliano, "A semi-supervised approach to question classification." in *ESANN*. Citeseer, 2009.

[38] P. Blunsom, K. Kocik, and J. R. Curran, "Question classification with log-linear models," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006, pp. 615–616.

[39] W. Xia, W. Zhu, B. Liao, M. Chen, L. Cai, and L. Huang, "Novel architecture for long short-term memory used in question classification," *Neurocomputing*, vol. 299, pp. 20–31, 2018.

[40] D. Xu, P. Jansen, J. Martin, Z. Xie, V. Yadav, H. T. Madabushi, O. Tafjord, and P. Clark, "Multi-class hierarchical question classification

for multiple choice science exams," *arXiv preprint arXiv:1908.05441*, 2019.

[41] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?" in *China National Conference on Chinese Computational Linguistics*. Springer, 2019, pp. 194–206.

[42] M. Appelstål, "Multimodal model for construction site aversion classification," 2020.

[43] T. Tengvall, "A method for automatic question answering in swedish based on bert," 2020.

[44] T. Isbister and M. Sahlgren, "Why not simply translate? a first swedish evaluation benchmark for semantic similarity," 2020.

[45] I. Huggingface. (2021) Bert. [Online]. Available: https://huggingface.co/transformers/model_doc/bert.html#bertformultiplechoice

[46] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[47] M. Mitchell, "Why ai is harder than we think," 2021.

[48] J. Vig, "A multiscale visualization of attention in the transformer model," *arXiv preprint arXiv:1906.05714*, 2019.

[49] S. Vashishth, S. Upadhyay, G. S. Tomar, and M. Faruqui, "Attention interpretability across nlp tasks," *arXiv preprint arXiv:1909.11218*, 2019.

[50] S. Jain and B. C. Wallace, "Attention is not explanation," *arXiv preprint arXiv:1902.10186*, 2019.

[51] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," *arXiv preprint arXiv:1908.04626*, 2019.

[52] S. L. Piano, "Ethical principles in machine learning and artificial intelligence: cases from the field and possible ways forward," *Humanities and Social Sciences Communications*, vol. 7, no. 1, pp. 1–7, 2020.

[53] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[54] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[55] C. Lee, K. Cho, and W. Kang, "Mixout: Effective regularization to finetune large-scale pretrained language models," *arXiv preprint arXiv:1909.11299*, 2019.

Appendix A

Distribution of TREC

Label	Count	Label	Count
Abbreviation	17	Term	100
Expansion	78	Vehicle	31
Definition	544	Word	26
Description	281	Description (human)	50
Manner	278	Group	195
Reason	197	Individual	1017
Animal	128	Title	26
Body	18	City	147
Color	50	Country	158
Creation	207	Mountain	24
Currency	10	Other (location)	514
Disease	105	State	73
Event	58	Code	9
Food	107	Count	372
Instrument	11	Date	265
Language	18	Distance	50
Letter	9	Money	74
Other (entity)	229	Order	7
Plant	18	Other (number)	64
Product	46	Percent	30
Religion	4	Period	83
Sport	63	Speed	15
Substance	56	Temperature	13
Symbol	11	Size	13
Technique	39	Weight	15

Table A.1: Occurrences of fine grained labels in the TREC data set

Appendix B

TREC-6 classification errors

Question	Predicted label	Correct label
Vad är ryggradslösa djur?	Entity	Description
Vad görs med slitna eller föråldrade flaggor?	Entity	Description
Vad är Ohio State Bird?	Description	Entity
Vad är den största fellinjen nära Kentucky?	Location	Entity
Vilket amerikanskt stats motto är "Live free or Die"?	Human	Location
Vad är momsen i Minnesota?	Number	Entity
Vilken stadstidning heter "The Enquirer"?	Entity	Location
Vad är koppars smältpunkt?	Location	Number
Vilken New York City-struktur kallas också Twin Towers?	Entity	Location
Vad är elproduktionen i Madrid, Spanien?	Number	Entity
Vad är alla hjärtans dag?	Number	Description
Vad är bandbredd?	Number	Description
Vad är kriteriet för att vara juridiskt blind?	Description	Entity
Vad är mul- och klövsjuka?	Description	Entity
Vad är guldets smältpunkt?	Location	Number
Vad är höjden över havet i St. Louis, MO?	Location	Number
Vilken stadstidning heter "The Star"?	Entity	Location
Vilket kloster plundrades av vikingar i slutet av 700-talet?	Location	Entity
Vad organiserade Jesse Jackson?	Entity	Human

Table B.1: Misclassified samples from TREC-6.

Appendix C

Human evaluation questionnaire

Målet med detta frågeformulär är att kunna jämföra hur bra mina modeller är jämfört med människor. Det jag försöker göra är att klassificera frågor i olika kategorier beroende på vilket typ av svar som frågan frågar efter.

Till exempel, frågan “I vilken stad ligger Eiffeltornet?” skulle klassificeras som “Plats” då frågan är ute efter en stad. Om du är osäker på vilken kategori som en fråga hör hemma i, ta den som känns bäst och kör vidare, det är inte överdrivet viktigt. Vissa frågor kommer kännas som att de hör hemma i flera kategorier, ta då den som känns bäst.

Varje frågeformulär har 50 frågor som ska kategoriseras. Tack för hjälpen!

Kategorier:

- Förkortning - alla typer av förkortningar
- Beskrivning - definitioner, anledningar, beskrivningar, abstrakta koncept
- Plats - landmärken, länder, stater, etc.
- Människor - titlar, grupper av människor, beskrivningar av personer
- Nummer - koder, datum, längder, vikter
- Saker - alla typer av substantiv

Fler exempel:

- Vad är datumet för Boxing Day? - Nummer
- Vilka klädesplagg är polletter i Monopol? - Saker
- Namnge 11 berömda martyrer. - Människor
- Vad är det olympiska mottot? - Beskrivning

Appendix D

Regex used for ethical analysis

Two separate regular expressions were used in order to gather data points concerning gender discrepancies. They are presented in table D.1. Notice the lack of "man" in the regular expression for gathering questions concerning men, since "man" would garner too many false positives due to "man" also meaning "one" or "you" as in "How does one change a lightbulb?", or "How do you change a lightbulb?".

Women	`kvinn[a|or]|tjej(er)*`
Men	`män |kill[e|ar]`

Table D.1: Regular expressions used to gather gender specific questions. Notice the space after "män".

TRITA-EECS-EX-2021:612