Question-answering using KB-SBERT for The Swedish Transport Administration

Author
Axel Jönsson - axejo347
TDDE16 - Text Mining

Abstract

This project aims to explore the possibility of implementing a question-answering model at Trafikverket, The Swedish Transport Administration. The main goal of the model is for a user to be able to ask question about Trafikverket, its organization and operations using a chatbot deployed on Trafikverket's website. KB-SBERT and a fine-tuned version of the same model were implemented with a dataset of 1784 unlabeled question-answer pairs. The models works by taking the users question and comparing it to all questions in the dataset using cosine similarity. The answer connected to the question with highest similarity is then returned to the user. The accuracy for the base and fine-tuned model were 53% and 38% respectively. Because of the unlabeled dataset, common evaluation metrics could not be used. Instead, ChatGPT were given input in the form of paragraphs from Trafikverket's website and then asked to produce questions for each paragraph. The questions were then used as input in the models and the yielded answers were then manually checked if correct. Even though the performance of the models were subpar, it shows that a question-answering model can be implemented, but that other methods like generative models should be explored.

1 Introduction

Chatbots are software systems that have the ability to answer questions and converse in natural language in real-time (Gupta et al., 2020). With the increase in use and improvements of Machine Learning (ML), Artificial Intelligence (AI) and technologies like Natural Language Processing (NLP) and Neural Networks (NN) the use of chatbots has drastically increased (Gupta et al., 2020).

The increase in use of chatbots can especially be seen in the are area of business and customer support. Customers do not have to wait in a queue to be offered help nor do they need to adapt to the opening hours of customer service as chatbots are available 24/7. Chatbots can also minimize an organizations costs by allowing human resources to do more complex tasks and leave the more routine tasks like customer support and requests to the chatbot (Raj et al., 2023). Lastly, a well implemented chatbot can be easier for a user to handle than to look up their question(s) at the website.

There are several different types of chatbots, but the focus of this report will be on contextual chatbots and question-answering models. These models are more advanced than many others as they use ML and AI to understand the context of the conversation and sentiment of the user (Gupta et al., 2020). The model used is called SBERT and is described further in Sections 2.1 and 2.2.

The Swedish Transport Administration, or Trafikverket, is a government agency responsible for roads, rails, airways and waterways in Sweden. As they are one of the largest agencies in the Swedish government they need to be able to efficiently and easily answer a broad range of questions from the public as well as their customers. Today, Trafikverket has a spread out its FAQ on several different locations on their website and are therefore seeking a different solution in the form of a chatbot.

1.1 Aim

This report looks to analyze if a chatbot can be implemented using open-source models to be able to answer questions from a pre-defined dataset using SBERT and cosine similarity at Trafikverket. The insights gained from the project can be valuable, not only for Trafikverket, but for applications such as customer support chatbots, information retrieval systems, and interactive virtual assistants.

2 Theory

The following section presents relevant theory to the project.

2.1 The BERT-model

BERT, or Bidirectional Encoder Representations from Transformers, is an NLP-model able to use context both left and right of a sentence or word, hence Bidirectional (Gillioz et al., 2020). Models before BERT could only use context to the left of the target, which decreased the performance when doing tasks like question-answering as tasks like this depends on context from both sides of the target (Gillioz et al., 2020). BERT's architecture is based on the Multi-Head Attention layer encoder. In simple terms, the encoder transforms a series of symbol representations

$$(x_1, ..., x_n)$$

into a corresponding sequence of continuous representations, denoted as

$$z = (z_1, ..., z_n).$$

The decoder, on the other hand, yields a sequence of symbols

$$(y_1, ..., y_n)$$

as an output one element at a time (Vaswani et al., 2017). The previously yielded symbols are added as input when yielding the next symbol, which makes the model auto regressive.

The BERT-model learns its language representation by an unsupervised learning phase (Gillioz et al., 2020). However, because of the bidirectional nature of BERT it must be trained using different methods than other models. This is because each word can observe itself and as a result can predict the following token. To mitigate this BERT use something called Masked Language Model (MLM). MLM hides, or masks, a random number of tokens in the input to then use the context to predict the vocabulary id of the masked words (Devlin et al., 2018).

2.2 KB-BERT

KB-BERT is a Swedish BERT-model developed at the National Library of Sweden, Kungliga Biblioteket (KB) (Malmsten et al., 2020). Finding data for smaller languages like Swedish to use for training a BERT-model can be challenging. To circumvent this issue, and to make sure that their BERT-model was trained on a broad range of Swedish, KB used parts of the library's database with focus on colloquial language like newspapers and reports from the government. The model is trained on modern language sources ranging from the 1940's to late 2019 and was pretrained using code and instructions from Devlin et al., 2018.

2.3 Cosine similarity

Cosine similarity is a metric to measure how similar two vectors are (Guo, 2022). The cosine similarity score is defined by the following

$$s = cos\theta = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$

where s is the cosine similarity score, A and B are the two vectors, $\|A\|$ and $\|B\|$ are the length of the vectors A and B, and a_i and b_i are the i-th element in the corresponding vectors A and B. As all elements a_i and b_i are positive the cosine similarity score is in the range [0, 1] (Guo, 2022). A score of 1 means that the vectors are identical while a score of 0 means that the vectors are orthogonal, or completely different.

2.4 Sentence BERT

Sentence BERT, or SBERT, is a modified version of the BERT model which can be used to attain sentence embeddings that are semantically meaningful (Reimers and Gurevych, 2019). It uses semantic search to do tasks like clustering, information retrieval and semantic similarity comparison. Semantic similarity comparison is applicable when comparing two sentences semantic content to, for example, find similar user queries. A disadvantage with BERT is that it does not compute any independent sentence embeddings (Reimers and Gurevych, 2019). This leads to a difficulty with acquire sentence embeddings. A workaround is to attain these fixed vectors by averaging the outputs when passing a single sentence or using the *CLS* tokens output, but using these methods results in quite bad sentence embeddings. SBERT however, allows for input in the form of fixed size vectors and semantically similar sentences can be acquired by methods like cosine similarity.

KBLabs sentence-bert-swedish-uncased is The National Library of Sweden's own SBERT-model built upon KB-BERT (KBL, a).

3 Data

The data used in the project is a previously scraped dataset from Trafikverket's website. The dataset

consist of 1 784 unlabeled question-answer pairs in Swedish. Below is an example of how the dataset is constructed:

```
0#Hur många frågor är det i kunskapsprovet?
0#Det är 70 st frågor i kunskapsprovet [...]
1#Hur lång tid är provet?
1#Du har 50 minuter på dig [... ]
2#...
2#...
...
```

i.e., the question is first stated and then follows the answer to that question.

4 Method

The method section is divided into four distinct parts: data collection, implementation, and evaluation where each subsection goes into detail about how each step was done.

4.1 Data collection

The data was readily available at Trafikverket before the start of the project. It had been scraped from Trafikverket's website in early 2022 and was intended to be used to implement a chatbot but was never used until this project. How the data was scraped and what software that was used, is information that is lost. However, the data was never preprocessed in any way or form.

4.2 Implementation

The implementation was done in the following steps: encode data, inference and fine tuning.

4.2.1 Encode data

To be able to use the data it needs to be encoded. This was done by loading all questions and answers from the data file to a dictionary with the question-answer index as key and the question and answer as the value. This dictionary was then used to create the embeddings for the questions with the encoder from the SBERT-model. The corresponding answer was also saved in its own list. KBLab's Huggingface site for their SBERT-model, KBL, b, was used to understand how the encoding worked and was implemented.

4.2.2 Inference

The inference is done by calculating the cosine similarity between the asked question and all the question embeddings that was obtained from the encoding. The cosine similarity is then sorted in descending order as the desired output is the question pair with the highest cosine similarity. The function then returns the answer if the similarity score is above a certain threshold, in this case > 0.5.

4.2.3 Fine tuning

To be able to see if the SBERT-model could be improved it was fine-tuned with all data, i.e., all question-answer pairs according to code from Reimers and Gurevych, 2019. This was done by training the sentence transformer with batch hard triplet loss like the SBERT model described in Section 2.4.

4.3 Evaluation

As the dataset is unlabeled, trivial and common evaluation metrics could not be used. Instead, another, manual, approach was taken where Trafikverket's website was manually scanned, and paragraphs were used in a prompt in ChatGPT to generate questions to said paragraph. The translated prompt can be seen below

> Imagine that you are a person looking through Trafikverket's website to find answers to your questions. You read the following paragraph and have some questions:
>
> $Paragraph$
>
> List those questions for me below

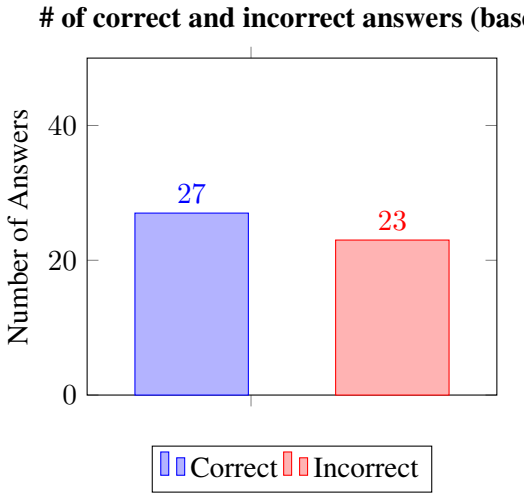
These questions were then used in the model to find the most similar question and answer in the dataset. If the cosine similarity is below 0.5 it is seen as a failed attempt to answer the question correctly and an answer is not given. If the cosine similarity is larger than 0.5 an answer is given and is then manually checked if it answers the question correctly. All questions, answers and checks if correct was then written to an Excel file. A total of 50 generated questions were used. This was done both for the base model and the fine-tuned model.

One metric that can be calculated with these results is accuracy. The accuracy was calculated by taking all questions that the model answered correctly divided by the total amount of questions asked.

5 Results

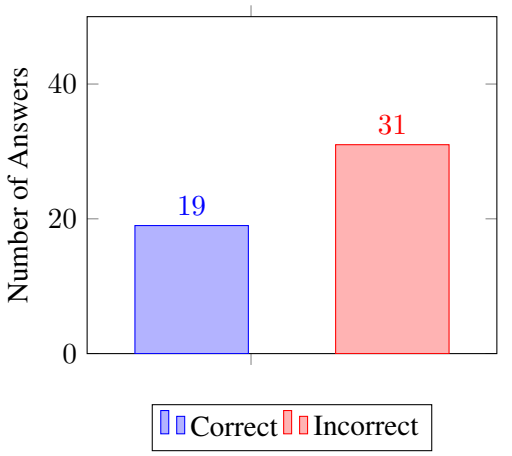
The results from the evaluation can be seen below. As presented, the number of correct answers were 29 and the number of incorrect answers were 21 out of 50 for the base model, which is an accuracy

of 54%.



The fine-tuned model performed worse with 19 correct and 31 incorrect answers out of 50. This is an accuracy of 38%.

This model could answer some questions correctly which the base model could not, but it did however answer more questions incorrectly as can be seen in the bar plot below.



Another important result was that the fine-tuned model had more questions where the cosine similarity was below the threshold and could therefore not answer these questions. The base model could not answer two questions, 4% of all questions, while the fine-tuned model could not answer five questions which is 10% of all questions.

6 Discussion

The models does work as expected, as it can answer question that a potential user may have. However, the accuracy of correctly answered question is barely over 50% for the base model. Deploying a chatbot which only answers questions correctly about 50%-55% of the time is not optimal. The fine-tuned model performed even worse with only 38% correctly answered questions.

The reason for the fine-tuned model performing worse than the base model is not clear. It could be that using batch hard triplet loss for fine tuning is not optimal with the kind of data used. The time constraint for the project did not leave ample time to investigate this issue.

One reason for the poor results for both models is probably that cosine similarity was used to find the most similar question to the one being asked. If the question asked is not close enough to the question in the data, the model will have issues matching the two and therefore yield incorrect answers. It works well when the questions asked are similar to the ones in the dataset, but it cannot be assumed that a user will ask questions that are similar enough. This was clear when using ChatGPT to come up with questions. Even though most questions generated could be answered, the questions were not close enough for the model to match. A solution could be to not match questions using cosine similarity, but instead training a model on a broader dataset with continuous text and not question-answer pairs.

This could be done by fine tuning generative models like ChatGPT or AI-Sweden's GPT-model GPT-SW3. These models can be fine-tuned on data using similar API:s as used in this project. One advantage of using these models is that a user can hold a conversation and ask follow up question which SBERT cannot. The reason that these models were not assessed is because of several factors. One being that it costs money to fine tune Open AI:s GPT-models and another that AI-Sweden requires permission to use their models which was not granted to this project. The last reason is that the data was in question-answer pairs which is not optimal for fine tuning these GPT-models.

The data used could also be improved. It is a limited data set with only 1784 question-answer pairs and when scanning the data most questions are about extremely specific operations at Trafikverket. A better dataset would be more general knowledge about Trafikverket and possibly in continuous text using a generative model as described above. Even though the dataset is specific, it does not affect the models results, but rather the actual use of the

chatbot.

The evaluation metric used was also subpar. Using an unlabeled dataset with limited time excluded several common metrics. Even though using ChatGPT to generate questions worked quite well it took time to achieve the desired results and it may not be the best way to do it. Having a test group to come up with questions would have been a better option as it would have been real people coming up with the questions, but this was not possible in the projects scope. Another test group could also be used to acquire user-based evaluations where the group can test the chatbot and then answer a survey on their experience.

7 Conclusion

To conclude the project, a working model was implemented to answer questions regarding Trafikverket and its organization and operations. The results were however not satisfactory for a question-answering model at Trafikverket at this time. It did, however, show that a chatbot can be implemented, but other methods, models and possibly datasets need to be evaluated for it to be a deployed.

References

a. Introducing a swedish sentence transformer. https://kb-labb.github.io/posts/2021-08-23-a-swedish-sentence-transformer/. Accessed: 2023-12-27.

b. Kblab/sentence-bert-swedish-cased. https://huggingface.co/KBLab/sentence-bert-swedish-cased. Accessed: 2023-12-20.

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

Anthony Gillioz, Jacky Casas, Elena Mugellini, and Omar Abou Khaled. 2020. Overview of the transformer-based models for nlp tasks. In *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, pages 179–183. IEEE.

Ken Guo. 2022. Testing and validating the cosine similarity measure for textual analysis. *Available at SSRN 4258463*.

Aishwarya Gupta, Divya Hathwar, and A Vijayakumar. 2020. Introduction to ai chatbots. *International Journal of Engineering Research and Technology*, 9(7):255–258.

Martin Malmsten, Love Börjeson, and Chris Haffenden. 2020. Playing with words at the national library of sweden–making a swedish bert. *arXiv preprint arXiv:2007.01658*.

Rohit Raj, Arpit Singh, Vimal Kumar, and Pratima Verma. 2023. Analyzing the potential benefits and use cases of chatgpt as a tool for improving the efficiency and effectiveness of business operations. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations*, 3(3):100140.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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