FRI Natural language processing course 2022
Project report

University of Ljubljana
Faculty of Computer and Information Science

Cross-lingual Question Answering

Nina Hostnik, Ajda Markič, Benjamin Plut

Abstract

In this paper we investigated what models are best for question answering in two languages: Slovene and English. We also investigated if any language features transfer between the two languages when given a question answering task. To achieve that we used a modified version of the Squad v2 dataset and a machine translated version of the same dataset and used it to train 4 different models. Our results show that while language features do not transfer well between Slovene and English, training a model on a mixed language dataset can yield good results in both languages.

Keywords
Question-Answering, Multilingual Question-Answering, roBERTa, SQuAD v2

Advisors: Slavko Žitnik

Introduction

With the amount of websites and data sources on the internet rapidly increasing with each passing year, it's becoming harder and harder for users to find answers to any questions they might have [1]. They want short and precise answers instead of having to sift through long documents and extraneous information, as this process can be quite time consuming due to the colossal amount of data on the internet. This is why Question Answering Systems (QAS from here on) are becoming more and more necessary. QAS enable the user to input questions in natural language. They then compute them, find the answer in their databases or on the internet and present it back to the user in the form of a natural language answer. In order to complete this task, QAS must be able to understand natural language. This can prove to be a challenge [2], as computers cannot yet process natural language as easily or effectively as humans. And while progress is being made, it is often limited to the English language. Most datasets used for training language models used to train QAS are made for the English language and while there exist some datasets for other languages, they often contain far less documents and may not offer sufficient training to language representation models.

This lack of datasets in other languages is often solved by translating documents from English into the target language. However, the results obtained often depend on the quality of translation. In this project, we train a BERT language representation model [3] on a fully English dataset, a machine translated fully Slovene dataset, and a mix of both. We then evaluate the quality of the question answering task performed on a Slovene language corpora.

Related works

Typically, cross-lingual QAS will solve the issue of having two languages by first translating one into the other using machine translation [4]. However, the success of such an approach is limited by the quality of the translation itself. Another possible approach is to consider alternate translations of target words. However, the alignment information for extracting target-source word pairs necessary in training such models makes this approach rather difficult. Deep neural networks have been employed to better the results of information retrieval(IR), using either pre-trained word embeddings or training based on IR objectives. While the latter have shown impressive results on monolingual datasets, their reliance on large amounts of annotated data makes it unclear whether they can perform as well on often much smaller cross-lingual datasets.

Recently however, pre-trained language models such as BERT [3] have been developed, that have outperformed traditional word embeddings. BERT models the underlying data distribution in languages to its linguistic patterns. It has successfully been applied to monolingual IR and was extended to perform ranking in cross-lingual IR, without the need for alignment data. It has also been used in the development of an evaluation dataset for multilingual QAS.

We found three appropriate monolingual question answering datasets and one multilingual. The most important requirement is that answers to the questions aren't scattered across multiple paragraphs. We allowed datasets where answers were sometimes scattered over multiple sentences. The first of the datasets we considered is SQuAD [5], a dataset consisting of over 100.000 question-answer pairs. Questions and answers were extracted from Wikipedia article paragraphs by crowd-workers. Each question was then given at least two additional answers, except in rare cases when the crowdworker could not find an answer. In the first version of SQuAD, there are no unanswerable questions, but they were added in SQuAD 2.0. Questions were divided into sets by dividing original articles - 80% of Wikipedia articles form the training set, 10% development set, and 10% a test set.

Similar datasets are QuAC and CoQA [6], which we do not consider appropriate because of their structure - questions can co-reference previous ones and yes or no answers are possible.

Another possible dataset is Google's Natural Questions (NQ) [7]. The dataset consists of sets of questions, Wikipedia pages that supposedly contain the answer, long answers, and short answers to the question (one of each per set). The source for the questions are google queries (8 words or longer and a specific set of rules to determine if the query is a question) that had a Wikipedia page in the first 5 search results. The long and short answers are extracted from said Wikipedia page and can be NULL if the answer is not found on the page. The dataset consists of 307.373 training examples with single annotations, 7.830 examples with 5-way annotations for development data, and 7.842 examples with 5-way annotations as test data.

Next possible dataset is TriviaQA [8]. Questions and answers were extracted from 14 different trivia and quiz websites and were then given a single (combined) evidence document from Bing search results or multiple evidence documents by using Wikipedia alone. Together there are 95.000 question-answer pairs that are organized into 650.000 training examples containing one document and 78.000 examples containing multiple Wikipedia articles. The problem with this dataset is that most of the automatically gathered evidence documents are large-scale and noisy, except 1975 human-annotated question-document-answer triples. Nearly 25% of the questions do not contain a clear answer in the evidence document.

Lastly, there is the multilingual dataset - Multilingual Knowledge Questions and Answers (MKQA) [9]. The dataset consists of 10.000 NQ questions that were translated into 26 languages or dialects from 14 language family branches. Answers are marked as either atomic value, entities, yes/no, short or long answers, or unanswerable. However, the dataset only contains questions and answers and would require the NQ dataset to provide Wikipedia articles.

Methodology

Data

We decided to use Squad v2 for our dataset due to problems with Google's NQ. We extracted data from JSON and translated it into Slovene using the CEF eTranslator [10]. We formatted both the original and the translated version into a new JSON format to turn the many-layered data into one layer that contained context, question, answers and title for each separate question, as shown in *Listing 1*.

To correct and clean our Slovene dataset, we first removed all unanswerable questions. We checked the remaining questions next, searching if at least one word from the question exists in the context. We used lemmatization to make sure the translation did not effect the search for the words. We removed the questions that shared no words with the context. Next we went through all the answers and checked if they are present in the context. If we did not find them, we lemmatizated both the context and the answer. If we still could not find any matching words between the answer and context, we also checked with a list of synonyms, which we obtained from [11]. If there still wasn't a match, we removed the question as unanswerable. However, if we managed to find the answer in the context through either lemmatization or synonyms, we replaced the answer with the one from the original context.

In our English dataset, only the original unanswerable questions were removed. Our mixed dataset consists of 45.000 questions from the cleaned Slovene dataset and 45.000 questions from the cleaned English dataset. We removed 10.000 questions from all original datasets and then used them to test the model.

After the cleaning and correction, our Slovene train dataset contained 76.270 answerable questions. Our English dataset contained 76.463 answerable questions. 37.439 answers were correctly translated from the start. 23.042 we corrected using lemmatization, 2.304 of which also needed synonyms to find the answer in the context. 43.400 questions were removed, either for being originally unanswerable or because we could not find the answer in the context in any way. 213 of them were removed due to the context not containing any of the words in the question.

Because official SQuAD test data is not available to the general public, we split both datasets into a train and test set. Both the Slovene test set and the English test set consist of 10,000 samples each.

Listing 1. Data structure

```
{
    "question" : "When was the first cafe opened
        in Paris?",
    "context" : "In addition to the classical
        restaurants, Paris has several other kinds
         of traditional eating places. The cafe
        arrived in Paris in the 17th century,
        ...",
    "answers" : [{
        "text" : "17th century"
    }],
    "title" : "Paris"
}
```

Models

For our base model, we initially decided to evaluate three pretrained models from Hugging Face [12] on the english version of our dataset and use the one with the best results for building our cross-lingual models. The first model was a cased multilingual BERT [13, 3], second was a base roBERTa [14, 15] and the third was BART [16, 17]. Because we are working with both Slovene and English data, we decided to also add SloBERTa [18] to the evaluation. All base models were fine-tuned using the Pytorch library [19].

For this comparison we trained all three models using the same hyperparameters: 1 epoch, an initial learning rate of 3e-5 on an Adam optimizer, a modified English SQUAD 2.0 training set, a max sequence length of 384, with a stride of 128 and a batch size of 8. Since roBERTa had the best results, we used it to test language transfer on a single model. In addition, we also decided to compare language transfer between the four models when only trained for 1 epoch.

To get predictions for the answers we used the *pipelines* function [20] from the Transformers library [21]. The pipeline is initiated by specifying question answering then adding the model you want to use and the tokenizer for that model. Then you can call on the variable with a context and question and it returns a prediction.

Listing 2. Prediction example

```
{
    "question" : "Kje so odkrili 3,7 milijarde let
         stare metasedimentarne kamnine?",
    "context" : "... Drugi zgodnji fizicni dokazi
        o biogeni snovi so grafit v 3,7 milijarde
        let starih metasedimentarnih kamninah,
        odkritih na zahodni Grenlandiji ...",
    "prediction" : "zahodni Grenlandiji",
    "real answer" : "zahodni Grenlandiji."
}
```

Training

Model training was done on a PC with an AMD Ryzen 7 3700x processor, 16GB of RAM and a Nvidia RTX 3080 10GB graphics card. We used a batch size of 8 because of GPU VRAM limitations. For training, we used an example script from Hugging Face's Transformers GitHub repository [22] and modified it slightly to work with our data. Most notably, we had to add the *answer_start* value during preprocessing rather than it being already present in the dataset. This is because the positional index of the answer within the context was no longer accurate after the dataset was translated.

The preprocessing included stripping the question of any leading whitespace, followed by tokenizing the question and context. The maximum sequence length was set to 384 characters. Longer sequences had their context turncated and the overflow returned and mapped back to sample. Answers were then mapped to context by finding the start and end tokens of the answer in the context.

The training hyperparameters were set to a learning rate of 3e-5

Model comparison

To compare our models, we decided to use 3 different statistical measures: F1 score [23], similarity based on Levensthein distance [24] and exact match. F1 score was chosen due to it being a measure of similarity between the predicted string and the real answer string. It is calculated using Precision and Recall, which are calculated using *true positive* (tokens that appear both in the prediction as well as in the real answer), *false positive* (tokens that appear in the prediction but not in the real answer) and *false negatives* (tokens that appear in the real answer but not in the prediction). Similarity based on Levensthein distance is a measure that compares 2 strings and returns how similar they are. It is based on Levensthein distance, which calculates how many edits it would take to turn one string into the other. And the last statistical measure is the exact match, which counts the number of predictions that exactly match the real answer.

To calculate the F1 score, we first remove any punctuation from both the prediction and the real answer. We then tokenize both and remove the stopwords from both using the NLTK library [25]. We obtain the true positive, false positive and false negative values and use them to calculate Precision, Recall and the F1 score. We save the F1 score and compute a macro average F1 score for the whole test dataset. We also save the true positive, false positive and false negative values for every prediction/true answer pair and use them to calculate a micro average F1 score for the whole test dataset.

$$F1 = 2 * precision * recall / (precision + recall)$$

To calculate the similarity, we first find the edit distance between the 2 strings using the edit distance library [26] for python and then we input it into the formula[add formulas q.q], from which we get the similarity. We also use the edit distance to figure out if a prediction is an exact match. If the edit distance equals 0 then the prediction is an exact match.

$$similarity(a,b) = 1 - (levensteinDistance / max(len(a), len(b)))$$

Results

Language transfer for multiple base models

As shown in table 1, the best model trained on English data and used on English test data is roBERTa. It achieved the best results in all metrics. SloBERTa performed the worst; particularly bad was the exact match score. From the results of testing English trained models on Slovene test data, shown in 2, we can see that both English only models, roBERTa and BART performed badly on the F-scores. RoBERTa, however, had the highest similarity of the four models. The model with the highest F-scores is SloBERTa, however, it had by far the worst exact match score out of all four models. The highest exact match score was achieved by Multilingual BERT, which had mediocre results in other metrics.

Model	Multi-BERT	roBERTa	BART	SloBERTa
F1 macro	0.84843	**0.86659**	0.85582	0.76508
F1 micro	0.80761	**0.83664**	0.81745	0.71912
exact match	614	**623**	612	19
Similarity	0.79812	**0.81509**	0.80754	0.65476

Table 1. Results of testing on English train data with 1000 samples of English test data

Model	Multi-BERT	roBERTa	BART	SloBERTa
F1 macro	0.45379	0.24557	0.21493	**0.53601**
F1 micro	0.42293	0.22767	0.20004	**0.52553**
exact match	**262**	119	100	26
Similarity	0.31818	**0.81509**	0.28233	0.56102

Table 2. Results of testing on English train data with 1000 samples of Slovene test data

Model	Multi-BERT	roBERTa	BART	SloBERTa
F1 macro	**0.77593**	0.62950	0.59290	0.66839
F1 micro	**0.72830**	0.57292	0.55038	0.62345
exact match	**520**	406	382	14
Similarity	**0.72949**	0.59484	0.57105	0.57253

Table 3. Results of testing on Slovene train data with 1000 samples of English test data

Model	Multi-BERT	roBERTa	BART	SloBERTa
F1 macro	0.49040	0.40122	0.38512	**0.54253**
F1 micro	0.46554	0.36261	0.34336	**0.52434**
exact match	**304**	240	233	24
Similarity	**0.57087**	0.48514	0.46131	0.56296

Table 4. Results of testing on Slovene train data with 1000 samples of Slovene test data

Results of testing Slovene trained models on English test data are shown in table 3. Multilingual BERT had the highest scores in all metrics, with the other three models all achieving results similar to each other, but somewhat worse than Multilingual BERT. The results of testing Slovene trained models on Slovene test data, shown in 4, show that while SloBERTa had the best F-scores, highest exact match and similarity metrics were achieved by Multilingual BERT. Both roBERTa and BART gained lower overall scores, however, they both outperformed their English trained counterparts.

While roBERTa gave the best scores with English to English, it achieved mediocre scores on mixed language and Slovene only tests. In mixed language tests, the best performing were Multilingual BERT and SloBERTa, but the latter only when testing on Slovene test data. BART had no interesting results in any of the four tests. Throughout all four tests, SloBERTa had consistently terrible exact match results, despite good results in other metrics. All the models performed worse when trained or tested on Slovene. This is likely a result of the Slovene dataset being smaller due to unfixable translation errors.

Language transfer on roBERTa

The models were tested on both the Slovene and English test sets. The results of testing the models on Slovene test data are shown in table 5. As expected, the model that has been trained on English data performed the worst on Slovene test data. The model trained on mixed data performed much better. It achieved only slightly worse results than the model fine-tuned on Slovene, which performed the best. The results of testing the models on English test data are shown in table 6. Here, the model trained on Slovene data performed the worst. The model model trained on mixed data performed almost as well as the model trained on English and, interestingly, even produced more exact matches than the English trained model. In both cases, the model trained on mixed data came close to matching the results of the model trained and tested on the same language. All of the models performed worse on the Slovene test set than they did on the English test set. This is because we used a roBERTa model that had been pretrained on English corpora.

Trained on	English	Mixed	Slovene
F1 macro	0.21533	0.44463	**0.45107**
F1 micro	0.20881	0.41192	**0.42077**
exact match	917	2877	**2939**
Similarity	0.29427	0.54085	**0.54268**

Table 5. Results of cross-lingual testing of language models on Slovene test data

Trained on	English	Mixed	Slovene
F1 macro	**0.87551**	0.87263	0.63424
F1 micro	**0.86121**	0.85773	0.60633
exact match	6852	**6896**	4427
Similarity	**0.84354**	0.84292	0.62260

Table 6. Results of cross lingual testing using multiple models tested on English

Discussion

We conclude that fine-tuning a model on a second language will allow it to have much better results when used on that second language, however, it will come at a cost of decreased performance on the original language. In this paper, we used roBERTa due to its performance on English data, however, model comparisons showed that Multilingual BERT, while not always the best, achieved good results throughout when compared to other models. Models that were only trained on English had very bad results when attempting to answer questions in Slovene and vice versa. This suggests that there is not much language transfer between the two languages. However, this is partially expected, as English and Slovene are in separate language branches. In further works, it would be interesting to explore language transfer between languages in the same language branch.

Acknowledgments

We would like to thank the FRI NLP 2022 discord server for their help and prof. Šikonja for helping us find Slovene language resources.

References

[1] Poonam Gupta et al. A survey of Text Question Answering Techniques. *International Journal of Computer Applications*, 2012.

[2] Patrick Lewis et al. MLQA: Evaluating Cross-lingual Extractive Question Answering. *arXiv*, 2020.

[3] Jacob Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv*, 2019.

[4] Zhuolin Jian et al. Cross-lingual Information Retrieval with BERT. *arXiv*, 2020.

[5] Pranav Rajpurkar et al. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[6] Mark Yatskar. A Qualitative Comparison of CoQA, SQuAD 2.0 and QuAC. *arXiv*, 2019.

[7] Tom Kwiatkowski et al. Natural Questions: a Benchmark for Question Answering Research. *Transactions of the Association of Computational Linguistics*, 2019.

[8] Mandar Joshi et al. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *arXiv*, 2017.

[9] Shayne Longpre et al. MKQA: A Linguistically Diverse Benchmark for Multilingual Open Domain Question Answering. *arXiv*, 2021.

[10] CEF eTranslator. Accessed: 28. 4. 2022.

[11] CJVT - center za jezikovne vire. https://www.cjvt.si/. Accessed: 2022-05-24.

[12] Hugging Face. https://huggingface.co/. Accessed: 2022-05-03.

[13] Hugging Face: bert-base-multilingual-cased. https://huggingface.co/bert-base-multilingual-cased. Accessed: 2022-05-24.

[14] Hugging Face: roberta-base. https://huggingface.co/roberta-base. Accessed: 2022-05-24.

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *arXiv*, 2019.

[16] Hugging Face: bart-base. https://huggingface.co/facebook/bart-base. Accessed: 2022-05-24.

[17] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, 2019.

[18] Clarin.si - sloberta. https://www.clarin.si/repository/xmlui/handle/11356/1397. Accessed: 2022-05-24.

[19] PyTorch. https://pytorch.org/. Accessed: 2022-05-24.

[20] Hugging Face: Transformers - pipelines. https://huggingface.co/docs/transformers/main_classes/pipelines. Accessed: 2022-05-24.

[21] Transformers. https://huggingface.co/docs/transformers/index. Accessed: 2022-05-03.

[22] GitHub: huggingface/transformers - examples/pytorch/question-answering. https://github.com/huggingface/transformers/tree/main/examples/pytorch/question-answering. Accessed: 2022-05-24.

[23] Wikipedia: f-score. https://en.wikipedia.org/wiki/F-score. Accessed: 2022-05-24.

[24] Wikipedia: levenshtein distance. https://en.wikipedia.org/wiki/Levenshtein_distance. Accessed: 2022-05-24.

[25] Natural Language Toolkit. https://www.nltk.org/. Accessed: 2022-05-24.

[26] PyPI: editdistance. https://pypi.org/project/editdistance/0.3.1/. Accessed: 2022-05-24.