Saarland University
Department of Language Science and Technology

MSc Thesis in Language Science and Technology

Evaluation of Transfer Learning Approaches for Cross-Lingual Question Answering

Author:
Christine Schäfer
Student number: 2553704

Supervisors:
Prof. Dr. Günter Neumann
Prof. Dr. Josef van Genabith

October 22, 2020

Declaration
I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged. I assure that the electronic version is identical in content to the printed version of the Master's thesis.
Saarbrücken, 22 October 2020, Signature:

Abstract

In cross-lingual question answering systems try to find answers to natural language questions in languages they were not (mainly) trained on. This thesis looks at different approaches for cross-lingual transfer on the XQA corpus [Liu *et al.*, 2019a]. It first investigates the corpus and compares it to other cross-lingual question answering datasets. The next chapters explore several potential enhancements to the XQA baselines. The first investigates whether cross-lingual word embeddings can be used for cross-lingual transfer in a QA-model. The next part asks the question if small amounts of target language training data can improve a model that was trained in the source language. Another section explores how well training on one cross-lingual dataset transfers to others. The last investigated questions are if shallow input features that proved helpful in non-neural baselines can enhance mBERT and if the paragraph selection features in the baselines are suitable for the XQA dataset.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Question Answering (QA) is a subfield of natural language processing where systems automatically answer natural language questions. There are two types of cross-lingual question answering: One where questions and context are in different languages and one with cross-lingual transfer. This thesis deals with the second type. There, a question answering model is trained in a resource-rich source language and later applied to a different target language. For this application the information gap between source and target language has to be bridged. While some cross-lingual-QA models are combinations of a translation part and a QA-part, the focus here lies on methods that transfer the training without explicit translation – in one case through shared word embeddings, in another case through multi-lingual language models.

The main corpus in this thesis is XQA [Liu *et al.*, 2019a]. It is a corpus for open question answering where answers can be either extracted from a context or freely formulated with information from that context. Its main difference from other question answering corpora is its large context size. It does not have a specific domain but all questions and contexts stem from Wikipedia. The source language of this corpus is English and it includes eight target languages from different language families. In this thesis we will explore several modifications to the XQA-baselines. Its main contributions are analysing the XQA corpus with simple baselines, using cross-lingual word embeddings for transfer and exploring the usefulness of small amounts of target language training data.

The next chapter summarizes cross-lingual question answering in the context of its two parent fields – question answering and general cross-lingual NLP. It gives an overview over common approaches and challenges of cross-linguality. Chapter 3 describes both XQA and other corpora used in later experiments and compares them. It introduces several trivial baselines that show which parts of the corpus are easier to solve and where potential weaknesses lie. The next chapters contain different approaches for cross-lingual transfer. Chapter 4 adapts the mono-lingual DocumentQA-baseline for cross-lingual word embeddings to transfer its monolingual training to the target languages and evaluates different kinds of embeddings. While this transfer is better than using monolingual models for other languages, it works much worse than transfer with mBERT. Chapter 5 assumes a case where small amounts of target language training data are available and tries to gauge how large this amount has to be to improve

over purely mono-lingual training. Additional training data that only consists of 5,000 or less examples is too small to be effective. The next chapter (6) compares how well the previous approaches transfer to other corpora. It finds that both the transfer from and to XQA is difficult but transfer between other QA corpora works well. The following two chapters modify XQA's BERT baseline with additional input data and compare different paragraph selection processes for DocumentQA. This reveals that the original method for paragraph selection is not the most effective and both a new method and a simple first-n-paragraphs method surpass it. The thesis concludes with ideas for future work.

Chapter 2

Cross-lingual Question Answering

Question answering (QA) is the task where systems automatically answer questions posed by humans in natural language. It can be distinguished from information retrieval because the system has to provide a concise answer and not just a list of documents which contain the answer [Bos and Nissim, 2006]. Practical use cases for question answering include search engines [Etzioni, 2011] and digital assistants. Cross-lingual question answering refers to question answering tasks with multiple languages involved and interacting with each other.

2.1 Question Answering

Question answering can be divided into subfields based on the format of the data the system uses to answer questions. QA-systems for structured data are based on knowledge bases, systems for semi-structured data are based on tables. This thesis will focus on unstructured data where the knowledge is available as plain texts. Like in the datasets here, the plain text often comes from Wikipedia but every large corpus, e.g. a general web corpus or a domain-specific corpus, could be used.

There are also different types of questions: For *information seeking questions* the question poser does not know the answers yet but wants to know them. This means, that there may be no answer to a question or that the asker might not know enough of the topic to form an unambiguous or meaningful question. The focus lies on the answer: **What is the answer?** The goal in this scenario is just to find an answer if it exists. This can be further specified to finding an answer that provides the right amount of background and explanation for the asker.

For *reading comprehension questions* the question poser knows the answer and wants to know whether the answerer does too. Here the focus lies on the answerer: **Who knows the answer?** To test this, these questions should be unambiguous in their context. Reading comprehension questions are paired with a context that anchors the question. Sometimes a question is only clearly defined in this narrow context, which can be a problem when reading comprehension datasets are modified for a broader context, e.g. to include an information retrieval step. Some datasets for this type focus solely on reading comprehension and contain only answerable questions (e.g. SQuAD 1.0 [Rajpurkar *et al.*, 2016]) while others explicitly also

test the ability to recognize if a question can be answered (e.g. SQUAD 2.0 [Rajpurkar *et al.*, 2018]).

Trivia questions are another question category where the asker knows the answer (e.g. Quizbowl [Rodriguez *et al.*, 2019]). Unlike reading comprehension questions they are not paired with a text. Compared to other question types, they are usually long and contain much context that helps disambiguating the question. Boyd-Graber and Börschinger [2020] push for more trivia questions in QA-system evaluations because this question type is more challenging and more discriminative.

After looking at the *questions*, question answering can also be categorized by its answers: Common answer types are factoid answers, list answers, long explanatory answers or opinion-based answers [Prager, 2006]. For factoid answers the categorization is often even more fine-grained and distinguishes, e.g. numbers, locations, persons etc.

On another axis there is a difference between retrieved answers that are an excerpt from the given context, generated answers, multiple-choice answers and yes/no answers. Some datasets mix answer types, e.g. retrieved answers and yes/no-answers. A newer answer category are answers with attached support or evidence that the answer is correct like in Yang *et al.* [2015].

Classic architectures of QA-systems include two-stage retrieval-reader approaches, end-to-end learning and retrieval-free models. The different possible steps of multi-stage approaches and their relation to general question answering are discussed in the next paragraphs:

Question Answering vs. Information Retrieval Information retrieval (IR) is the process of finding relevant documents in a large corpus. Relevance is determined with regard to a prompt which may be a question but also just a keyword or set of keywords. The other main difference between question answering and information retrieval is the length and specificity of the answer. Information retrieval returns a list of documents which – if the prompt was a question – are likely to contain the answer. Besides the answer, they also contain much unrelated information. A question answering system, however, should just return information relevant to the answer. If sentences or phrases in the documents are viewed separately, one could see information retrieval systems as very high-recall, low-precision answer candidate finders for extractive QA. Therefore, information retrieval systems are often used as first step in QA-systems that narrow down where subsequent parts have to search for answer candidates.

Reading Comprehension Reading comprehension (RC) or machine reading comprehension (MRC) is the second step of two-stage approaches. In this step, answers are extracted from short given paragraphs. Question answering systems that have reading comprehension as a second step are for example Chen *et al.* [2017] and Yang *et al.* [2019]. Jing *et al.* [2019] work just on reading comprehension but name question answering as an application for their research. Other researchers such as Lewis *et al.* [2019] work effectively on reading comprehension but refer to it more generally as question answering.

There are also different views on the terminology for reading comprehension: Lai *et al.* [2017] regard reading comprehension, that is the ability of systems to understand text, as their

research goal and use question answering just as an evaluation method to measure the systems comprehension ability.

In this thesis, *question answering* will refer to entire process of finding an answer, while the last extraction step will be called *reading comprehension*.

Span prediction Span prediction is a variant of reading comprehension where the answer has to occur literally in the context. The task is to find the minimal token span in a given text that answers the question. What *minimal* answer span means, has to be defined further for every task or dataset: Does it include function words and punctuation? Should it always be full sentences or grammatical phrases or should it just contain the relevant information without necessarily being grammatical?

Cloze prompts Cloze prompts are sentences with missing words. They are often used as questions in QA-tasks even though linguistically they are usually other sentence types. For cloze prompts the answers are the words that plausibly fill the gap in the sentence.

Classic question answering tasks have just isolated question-answer pairs. There are also more complex tasks like multi-hop question answering, where the model has to combine several passages to find an answer, conversational question answering where questions can refer to earlier questions or answers and reasoning based question answering. For these complex tasks there typically only exist monolingual English datasets.

2.1.1 Evaluation of Question Answering Systems

While older question answering systems were often evaluated manually [Prager, 2006], today correctness is usually measured as overlap with gold answers with exact match and F1 as measures. Some datasets have several possible answers per question, others provide only one gold answer. Depending on the type of expected answers, this can lead to underestimating the performance as there likely exist more acceptable answers than there are explicitly listed in the answer set. This is a similar problem as using BLEU scores as a measure for machine translation or natural language generation.

To quantify the difficulty of a dataset the "human performance" on it can be measured. Sometimes it is computed by giving human annotators the same task the system has to solve, sometimes it is estimated differently. Clark *et al.* [2020]'s approach of taking two annotations as gold and treating the third human annotation as if it was produced by a QA-system under evaluation is given as a lower bound for human performance but it is also a set-up that makes it comparable to systems because the system's answers are also just checked against gold answers and not directly. For some monolingual QA-datasets [Rajpurkar *et al.*, 2016], the best systems surpass human performance on the standard evaluation. However, high accuracy on question answering datasets does not mean that the task of question answering is solved. Ribeiro *et al.* [2020] criticize that "held-out datasets are often not comprehensive, and contain the same

biases as the training data [...], such that real-world performance may be overestimated". They also propose to not use accuracy as the sole evaluation measure but also evaluate fairness, robustness and how the system deals with linguistic phenomena like named entities, coreference or negation. With regard to question answering, they test systems for the standard dataset SQUAD and still find many errors on challenging linguistic phenomena even though the accuracy of the systems on the test set is better than human performance. There are also approaches that want to improve evaluation of QA-systems by not only looking for an answer but also for an explanation or justification for that answer [Inoue *et al.*, 2020].

2.2 Cross-linguality

Multi-lingual systems are NLP systems that deal with multiple natural languages. Cross-lingual systems are a subgroup of multi-lingual systems where these languages are not only present in the same system but also interact with each other. There are two forms of cross-linguality. One is a form of transfer learning: A system is trained in the source language and later evaluated in the target language. With the other, the system bridges between both languages both at training and at test time. In the case of question answering this means that question and context are in different languages.

So why is multi-linguality important and why is it especially important for QA-systems? Many NLP systems will be used by speakers of different languages. This is in particularly true for web-based systems as much of the information there is in English but should also be accessible to speakers of different languages. If several mono-lingual systems are assembled to a joint model, there has to be a language recognition step to choose the correct model for the user input. Otherwise, the system has to deal flexibly with multiple input languages.

Related to cross-linguality is also *language independence*: The older idea of language independence comprises using the same model for different languages but with new training data. In a sense, cross-lingual approaches are even more extreme: They use not only the same model but also partially or fully the same training data. Bender [2011] states the benefits of language independence: cost and time efficiency and a higher likelihood that systems for smaller languages will be built. She also expresses the hope that language independent system could teach something about the nature of language. These points are also advantages of cross-lingual learning.

It has often been noted that for most languages it is hard or even impossible to get annotated training data [Snyder, 2010; Conneau *et al.*, 2018]. This is especially true for low-resource languages with even only small amounts of unlabelled, raw data. For higher-resource languages one might get some training data but not enough to train a system only on that. With cross-lingual transfer, available training data in a high-resource language can be used for a model in the target language [Conneau *et al.*, 2018; Cui *et al.*, 2019].

Full model training can also be computationally expensive. With pre-trained language models, there is a huge movement adapting and fine-tuning trained models. If there is a

trained model available in the source language, it might be easier to just adapt this model to the target language instead of training another model from scratch. Artetxe *et al.* [2020a] compare incorporating a new language into a trained model to human life-long learning. Even if the model was planned to be multi-lingual from the beginning, the cross-lingual model only has to be trained once compared to training separate mono-lingual models.

There are also different words for the same concept: Lin *et al.* [2019] call the source language *transfer language* and the target language *task language*. This highlights how the source language facilitates the transfer learning, and the target language is the language that is later used during a task.

The second kind of cross-linguality is needed if a speaker of the source language wants to access information only available in the target language [Bos and Nissim, 2006]. This includes QA-systems with the question in the source and supporting documents in the target language but also the entire field of cross-lingual information retrieval. As Jing *et al.* [2019] note, this is a realistic scenario for QA-systems. Large text collections or knowledge bases are only available for some languages and a specific collection is often not translated. Cross-lingual systems increase the group of users which can get information from these collections.

For some of these systems source and target languages are fixed and they only work for this language pair and one direction. This is mostly the case for older translation-based models with statistical machine translation. Systems with multiple possible language pairs sometimes have an explicit language tag that specifies source and target language.

There are several obstacles that make cross-lingual transfer non-trivial. In her pre-deep learning paper, Bender [2011] describes how n-gram models work better for languages with fixed word orders because n-gram models can only capture dependencies between two words if they appear less than n tokens apart from each other. In languages with fixed word orders, it is more consistent which words with which dependencies appear close to each other. Word order effects could also be important for neural models, e.g. if an attention module learns that question words are in the beginning for the source language but in a different place or not explicitly present in the target language. Also different levels of inflectional morphology can influence portability. This is obvious in Bender's example how n-gram models find enough occurrences in languages with less inflection and run into sparse data problems with more inflection, but also neural networks with word embeddings as input layer usually embed tokens without lemmatization. In more inflecting languages more words will have no pre-trained embeddings.

Another problem are missing pre-processing tools. If the first step of a QA-model is a keyword extraction module that relies on external POS-taggers or named entity recognition, there is no use in transferring the QA-model to a target language for which these pre-processing tools don't exist. This also holds for models that don't use explicit language-specific pre-processing but make assumptions that hold only for some languages, e.g. that word segmentation can be approximated with white-space tokenization.

From these draw-backs follows that even unsupervised models might contain underlying

assumptions which mean that they still only work for some languages or work better for some languages, namely the languages the developers used to develop and test or the languages of which the developers have implicit linguistic knowledge [Bender, 2011].

Ponti *et al.* [2019] give an overview over types of multi-lingual models. They distinguish unsupervised models, learning joint models, learning models with multi-lingual representations and cases where the data or model is transferred to the target language. Language transfer can be achieved through different means: by projecting annotation to the target language, by transferring a delexicalized model or through translation.

Zero-shot transfer refers to approaches where a model is trained on a (often high-resource) source language and then directly applied to evaluation data in the target language without additional steps for cross-lingual transfer.

One factor that influences success of cross-lingual transfer is the similarity and relatedness between the languages [Cotterell and Heigold, 2017]. They write:

> While we only experiment with languages in the same family, we show that closer languages within that family are better candidates for transfer. We remark that future work should consider the viability of more distant language pairs

Dubossarsky *et al.* [2020] compare results for bilingual lexicon induction (BLI) and several downstream tasks like machine translation (but not QA) and find that performance is better if the mono-lingual word embeddings before the transfer were already in a similar space. Anastasopoulos and Neubig [2020] show better results in BLI for genetically related languages but remark that the gap is greatest for bilingual embeddings and narrows if the word embeddings are aligned for many languages. They also find that the choice of the hub language, which is used as the centre to map all embeddings to, matters.

Another hurdle for cross-lingual transfer are different scripts between source and target language. With different scripts, the language cannot share subword embeddings or character embeddings. This holds both for dedicated embedding layers and possible learned embeddings in the lower levels of a transformer.

The question how to select the best source language for a problem is actively discussed in different fields of cross-lingual NLP. Lin *et al.* [2019] work on the problem of how to select the best source language for a given problem and target language. Their use cases are cross-lingual dependency parsing and POS tagging. In practice the source language of often dictated by the availability of training data.

A subproblem of cross-linguality is code-switching. Code-switching is a form of multi-linguality where the same utterance has parts from different languages. Code-switching in the context of question answering was explored by Daniel *et al.* [2019] when they built a platform to answer questions about health-care for pregnancy and breast-feeding in South Africa. This system is not just a research model but used in a real-world application. Questions can be asked in all eleven of South Africa's official languages and users might code-switch.

2.3 Cross-lingual Ability of mBERT

One model that is often used for cross-lingual NLP is multi-lingual BERT (mBERT) [Devlin *et al.*, 2019]. mBERT is a pre-trained transformer model that was trained on Wikipedia texts in 104 languages. None of the training data is parallel and languages were sub- or super-sampled for more balance. All languages share a word-piece vocabulary and there are no markers for the current input language. The training tasks were masked language modelling and next sentence prediction. This pre-trained model is then usually fine-tuned on labelled data for a specific task.

Zero-shot transfers, that is fine-tuning only on source language training data and then directly applying mBERT to target language evaluation data, work quite well. Pires *et al.* [2019] find that they are often strong baselines (e.g. for Named Entity Recognition, POS-tagging).

Several publications try to answer the question why zero-shot transfer with mBERT works so well: Pires *et al.* [2019] assume that it is due to the shared parts of the vocabulary, namely numbers or fixed strings like urls that occur in multiple languages. K *et al.* [2020] disagree that shared vocabulary is the main cause because the transfer also works for language pairs with no word-piece overlap. In their experiment they use an artificial language *fake English* which is English where all letters were substituted with Unicode characters that don't occur in any language's Wikipedia version. Depending on language pair and task, word piece overlap contributed between 0.5 and 2.9 points accuracy or span f-score. Instead, they suggest similar structures between languages as the cause.

Wu *et al.* [2019] state that mBERT learns cross-linguality through shared parameters in the top layers of the model. They also show that by training the language model on different monolingual corpora a shared embedding space is created. This phenomenon is also explored in Wu and Dredze [2019].

Artetxe *et al.* [2020a] also argue that shared vocabulary or even shared subwords are not the cause of mBERT's cross-linguality. However, they find that the raw size of the vocabulary is important. Comparing different vocabulary configurations, they get the best results with a disjoint vocabulary that guarantees that every language gets a minimum amount of subwords allocated.

While cross-lingual mBERT systems generally work well, they are also potentially fragile: Hardalov *et al.* [2019] point to the risk of training too long on the monolingual dataset and forgetting the multi-lingual pre-training.

2.4 Approaches in Cross-lingual Question Answering

While monolingual question answering has been a subfield of natural language processing at least since Simmons [1965], cross-lingual question answering is newer. In the early 2000s it started with the CLEF (Cross-Language Evaluation Forum) and QALD (QA over Linked Data)

shared tasks. Before that cross-lingual NLP was already established in other fields like information retrieval.

CLEF and QALD published small corpora for factoid question answering. DISEQUA [Magnini *et al.*, 2003] for CLEF consists of 180 question-answer pairs in Dutch, Spanish and Italian with their English translations. At this size only evaluation corpora are possible. As a comparison: modern evaluation corpora are usually 10 times as large. A history of cross-lingual question answering can be found in Loginova *et al.* [2020]: It also includes later shared tasks like NTCIR, SemEval for community question answering and MSIR & FIRE for question classification of code-mixed questions.

There are two broad types of cross-lingual question answering: based on translations or on abstract representations. Translation-based approaches are more classic but still used as baselines or as core of more elaborate systems. What needs to be translated depends on the type of cross-linguality. If question and document are in different languages, either of them has to be translated. Saleh and Pecina [2020] find that in their experiments question translation works better than document translation. For question translation, it can either be translated completely or preprocessed in the source language by extracting keywords or slot fillers and then translating only the keywords. For training-evaluation cross-linguality, plain translation approaches can be divided into those that translate the training data and those that translate the evaluation data.

Bos and Nissim [2006] is an example for an early translation-based model and they also mention that the previous 28 participants of the shared task QA@CLEF use machine translation approaches. The baselines in Liu *et al.* [2019a] are a two-stage process with first offline translation and then using a QA-model on translated texts. This is the simplest kind of translation-based cross-linguality. Ture and Boschee [2016] use multiple translations of the same phrase in the same question answering model and learn how to translate best for the current task. Asai *et al.* [2018] also translate first. Later, they use not only the resulting translation but also other information from the translation step in the integrated QA-model.

However, Loginova *et al.* [2020] show that relying on machine translation for cross-lingual QA is often not adequate, especially for cases where questions or contexts are code-mixed or contain transliterations.

Approaches with abstract representations first transform the question into the representation and then work with the representation to find the answer. Possible types of abstract representations are sequences of word embeddings, logical forms, SQL queries or hidden layers of a neural model (e.g. a transformer).

Recently pre-trained language models like mBERT described in section 2.3 became common. Examples of using mBERT for cross-lingual question answering are Hardalov *et al.* [2019] for zero-shot reading comprehension with the target language Bulgarian and fine-tuning on the RACE dataset [Lai *et al.*, 2017] in the source language English. Liu *et al.* [2019b] combine mBERT and an answer extraction model based on BiDAF [Seo *et al.*, 2017] to answer cloze prompts for reading comprehension in English and Chinese.

Chapter 3

Corpora

There are many corpora for cross-lingual question answering with different characteristics and goals. Some include only one language pair [Jing *et al.*, 2019] while others compare many languages [Clark *et al.*, 2020]. If the corpus is parallel, the performance on different languages can be easily compared. If the corpus just contains unrelated questions from different languages, performance differences may also stem from different difficulties in the collected questions that are language independent.

Some cross-lingual question answering corpora are translations of monolingual QA-corpora. Translated corpora have several problems compared to corpora that were directly collected in the target languages [Clark *et al.*, 2020]: The translation may contain artefacts from the source language, e.g. the word order of the source language is kept for a target language with free word order. There are general differences between native text and *translationese* [Baroni and Bernardini, 2006]. Besides these problems with translated text for all tasks, question answering has the additional problem of different underlying world knowledge. As languages are typically tied to cultures or regions, people asking questions in different languages might ask questions about different things. All of this makes translated cross-lingual corpora problematic, but they are a comparatively easy way to create cross-lingual corpora – especially parallel corpora.

An ideal corpus should be similar to expected applications. Most academic question-answering datasets have some traits that make them artificially easy and thus not comparable or trivially transferable to real-world applications. Some of these traits are a high lexical overlap between question and answer sentence, small preselected contexts, assurance that all questions are answerable or that answers can be extracted from a continuing span of the context. While most QA-datasets don't share all of these traits, they often have enough to make the evaluation unnatural.

The main corpus in this thesis is XQA by Liu *et al.* [2019a]. It was selected because it contains several language pairs including English-German, it is not a translated corpus and it is challenging due to its large context sizes. The other corpora are MLQA, XQUAD – both evaluation corpora in the SQUAD-format – and TYDI QA that focuses on diverse languages.

3.1 XQA

XQA [Liu *et al.*, 2019a] is a multi-lingual question answering corpus for open domain question answering. It treats question answering as a three-part problem: information retrieval, paragraph selection and span extraction. In the corpus variant used in this thesis, the information retrieval part is already included, so only the paragraph selection and span extraction steps have to be performed by the tested models.

The XQA corpus was extracted from Wikipedia's *Did you know*-sections.[1] On Wikipedia a *Did you know*-fact has the form:

> [Did you know that] ... that scientist **Emma Teeling** of the BatLab in Dublin studies a genus of bats which do not appear to die of old age?

For each fact the linked article title – in this case *Emma Teeling* – is extracted and taken as one gold answer. Other gold answers are synonyms to the title according to the knowledge base Wikidata. The question is the fact with a special *query* token instead of the article title. So the resulting question is

> scientist <Query> of the BatLab in Dublin studies a genus of bats which do not appear to die of old age?

The context from which the answer has to be selected consists of the ten Wikipedia articles which are most relevant to the question. The relevance is measured with BM25 [Robertson *et al.*, 1995]. If other articles than the linked articles are closer to the question, the question will have no answer in the provided context. Additionally, the first paragraph of all context documents is not included in the dataset.

This construction process means that questions and answers were not intended as questions by the writers and are therefore a different style as the input to a QA-system. However they are naturally produced, not translated language.

The corpus consists of an English training set with ∼56,000 QA-pairs and development and test sets in nine languages: Chinese, English, French, German, Portuguese, Polish, Russian, Ukrainian and Tamil. The sizes of development and test sets are between 350 (Portuguese) and 3800 (German) instances. All parts are in the same format, but instances were collected independently so there is no parallel data and different distributions of topics.

Because the answers are Wikipedia article titles, all questions are factual questions with (mostly) named entities as answers. Due to the construction of the dataset, questions may have ambiguous answers that are not reflected in the dataset. In the example above, Emma Teeling presumably has colleagues who work at the same institution and study the same bats.

As XQA is just short for *cross-lingual / x-lingual question answering* which is an obvious name for such a dataset, there is at least one other cross-lingual QA-dataset with the same name by Huang *et al.* [2019]. This other XQA only includes three languages (English, French and German) and has a different subtask, namely to determine if a given answer is relevant to the current question. Huang *et al.* [2019]'s XQA is not used in this thesis.

[1] `https://en.wikipedia.org/wiki/Wikipedia:Did_you_know`

Language	Question length	Article length tokens (bytes)	Answer length tokens (bytes)	Answer occurrences	% with answer
English	18.68	747.23 (8065.80)	2.01 (63.53)	13.32	77.48
German	14.77	450.51 (6014.48)	1.97 (68.37)	6.63	60.14
French	21.30	889.47 (10584.05)	2.06 (68.94)	12.55	74.59
Tamil	13.61	198.17 (3632.30)	1.87 (102.46)	6.00	60.44
Polish	14.48	262.00 (3868.78)	2.03 (73.87)	3.54	52.60
Portuguese	17.41	463.96 (4296.48)	2.13 (69.90)	6.88	57.57
Russian	14.56	510.06 (7315.01)	1.97 (100.00)	4.29	49.77
Ukrainian	17.88	614.63 (8573.33)	2.07 (102.3)	11.29	65.12
Chinese	29.77	1097.39 (2503.56)	5.09 (83.97)	11.00	70.52

Table 3.1: Corpus statistics for XQA. Question, article and answer lengths are given in tokens where only non-punctuation tokens are counted. Article and answer lengths are also provided in bytes. Answer occurrences describes the number of times one of the answer strings appear in the context. The numbers in the column *with answers* are from Liu *et al.* [2019a].

3.1.1 Corpus Analysis

As comparison to the evaluation in Clark *et al.* [2020], this section lists some basic statistics about the XQA corpus. Table 3.1 shows lengths of questions, articles and answers, the number of answer occurrences in the context and the percent of questions with an answer. Note that an XQA context consists of 10 articles, so it is ten times the average article length from the table. Ukrainian, Chinese and Tamil were tokenized with the BERT-tokenizer, the other languages with their respective nltk-tokenizer. Like in the XQA-analysis, punctuation tokens were not counted towards the token length. With this there are still small differences in lengths between Liu *et al.* [2019a] and this analysis (e.g. article length for English 735.91 vs. 747.23, question length German 14.61 vs. 14.77) which could stem from different tokenizers. Article and answer lengths are given in tokens to compare them with question lengths and bytes to make them comparable to the TYDI statistics. Average article length varies considerably between languages: The average French article is four times as long as the average Tamil article. The same holds for question lengths and number of passage candidates. The answers are usually around two tokens with the exception of Chinese. TYDI-questions are with 5 to 7 tokens much shorter than average XQA-questions which are for all languages over 10 tokens, mostly 14 to 18. A single article in TYDI is larger than in XQA: between 5,000 and 30,000 bytes per language. For XQA, the lower bound is similar with 3,600 but the upper limit is 10,500 which is lower than nine of the eleven TYDI-languages. However, XQA has 10 articles for each question. Nevertheless, answer lengths are similar for both corpora: depending on the language between 60 (English) and 100 (Russian). Languages with different answer lengths from TYDI – Kiswahili with 39 on the lower end, Telugu with 279 on the higher end – don't occur in XQA.

An extractive QA-model can only find gold answers that are present in the context. The column answer occurrences counts how often gold answers occur in the given context. XQA has often multiple overlapping gold answers like the last name and full name of a person. This means the same string in the can count to multiple answer occurrences. The values between 4

Language	TyDi	MLQA	XQuAD	XQA
English	0.38	0.91	1.52	2.53
German	-	-	-	1.10
Tamil	-	-	-	1.24
French	-	-	-	2.19
Polish	-	-	-	1.07
Portuguese	-	-	-	-
Russian	0.16	-	1.13	1.06
Ukrainian	-	-	-	1.42
Chinese	-	-	-	0.0657
Arabic	0.26	0.61	1.29	-
Bengali	0.29	-	-	-
Finnish	0.23	-	-	-
Indonesian	0.41	-	-	-
Kiswahili	0.31	-	-	-
Korean	0.19	-	-	-
Telugu	0.13	-	-	-

Table 3.2: Comparing lexical overlap in XQA to other datasets. Numbers for other datasets from Clark *et al.* [2020]. The overlaps are the average number of tokens that occur both in the question and in a 200-character window around the gold answer in the context.

and 13 still show that the answers tend to occur multiple times in the context. Between half and three quarters of the questions have answers in the extracted context. This is generally a higher fraction than for TyDi where between 22% and 69% of questions have a span answer. Only for Russian which has the lowest fraction for XQA and second highest for TyDi, the fraction is around 50% in both datasets. A question word analysis comparable to TyDi is not possible because XQA questions are cloze prompts.

Table 3.2 shows the lexical overlap between the question and the context close to the answer. While TyDi has annotated answers spans, for XQA possible spans first have to be found to compute their close context. This is done with part of the DocumentQA-baseline. The close context are the tokens in a 200-character window around the answer span. The average lexical overlap for XQA is mostly between one or two tokens for XQA. English with less morphology has 2.53 and Chinese with a logographic script has very little overlap. Lexical overlap is a bit higher than for XQuAD and much higher than for TyDi. This means that answers are potentially easier to find by looking for a matching context.

3.2 Comparison to Other Corpora

Compared to other corpora, XQA includes some (unique) challenges: It has large context sizes and its answers are given as tokens not as answer spans.

To investigate whether our adjustments to the models are specialized for XQA or generalize also to other contexts, we also evaluate on other QA-corpora, which are briefly introduced and compared to XQA below.

3.2.1 MLQA

MLQA [Lewis *et al.*, 2019] is an evaluation dataset for cross-lingual question answering. Its task is extractive QA: Highlighting an answer span in a given context. It contains seven languages: English, Arabic, German, Spanish, Hindi, Vietnamese and Simplified Chinese. The English part is the largest with 12,700 instances. The other languages have between 5,000 and 6,000 instances which are again split into a development and a test set. The languages are selected to be diverse in terms of language family, script and available resources but the selection process is not as rigorous as for TYDI QA and is limited to languages with large Wikipedia versions.

MLQA is parallel with instances that are typically aligned between four of the seven languages. Because some items were dropped after the alignment, some instances are only three-way or two-way aligned. There are alignments between all language pairs. Unlike the other datasets in the chapter, MLQA is not only cross-lingual between training and evaluation but has also a part where question and context/answer language are different. This part is not used here.

Like XQA, MLQA is based on Wikipedia. All context documents are directly from the different Wikipedia versions, but its questions are translated. It was constructed by first aligning similar passages in different language versions of Wikipedia articles. In a second step, questions are created from the aligned sentences in English and then translated to the other languages. Finally, the answer spans for the translated questions are annotated in the context. Questions without answer spans in a language were discarded only for this language.

3.2.2 XQUAD

XQUAD [Artetxe *et al.*, 2020a] is a subset of the SQUAD1.1 development set with human translations into 10 languages: Spanish, German, Greek, Russian, Turkish, Arabic, Vietnamese, Thai, Chinese and Hindi. It contains 1190 question-answer pairs which means it is only an evaluation dataset. As context, it has only context paragraphs not full articles. As part of the translation process, each question only got one gold answer. Named entities were transliterated during translation. After the translation, the answer spans were checked.

3.2.3 TYDI QA

TYDI QA [Clark *et al.*, 2020] stands for *typologically diverse QA* and includes 11 typologically diverse languages. With 204,000 QA-pairs it is a larger corpus. For TYDI there are two tasks: finding the passage that contains the answer and finding the exact answer span.

Its thematic focus are information seeking questions and it was created to feature linguistic phenomena that differ between languages. There is no common topic for the questions. The only requirement for questions is that they are not opinion-based but fact-based.

The corpus was manually produced. The questions were written by humans who didn't know the answer while formulating the question. The questions were collected in all languages separately and later controlled for fluency. So there are no translations and scores are not com-

parable between languages. The contexts for the questions were retrieved automatically with a Google search but annotations for answer passages and answer spans were again conducted by human annotators.

In the answer span task which is comparable to the evaluation of XQuAD and MLQA, there are three possible answer types: A span, yes/no and no answer. A large portion of the dataset (between 46% and 82% per language) are unanswerable questions. If the answer is a span, it usually spans between a few words and up to a sentence. TyDi has a lower lexical overlap between questions and contexts which should make it more challenging. There are two baselines published with the dataset: a first passage baseline and the performance of mBERT.

3.3 Evaluation

XQA is evaluated with the TriviaQA [Joshi *et al.*, 2017] evaluation scripts. The evaluation metrics are F1 and exact match. If there are several ground truths, the metric is used on the ground truth with the best match, e.g. if *ground truth one* is identical to the prediction and *ground truth two* has no token overlap, it is still a perfect match.

Before the comparison the answers are normalized. Here normalization means that no determiners are considered, all white-spaces are considered the same regardless of which white-space characters they consist of, punctuation is stripped and all characters are converted to lower case.

F1 is computed on token basis as given in equation 3.1. Exact match is computed by normalizing ground truth and prediction and then comparing whether they are equal.

$$
\begin{aligned}
precision &= \frac{\#shared\,tokens}{\#prediction\,tokens} \\
recall &= \frac{\#shared\,tokens}{\#ground\,truth\,tokens} \\
F1 &= \frac{2 \times precision \times recall}{precision + recall}
\end{aligned}
\tag{3.1}
$$

3.4 Original XQA Baselines

With the release of the XQA corpus, Liu *et al.* [2019a] also release results on several baselines that are based on the systems *DocumentQA* and BERT. Most of the baselines are translation-based. Additionally, there is also a zero-shot mBERT baseline.

3.4.1 DocumentQA

DocumentQA [Clark and Gardner, 2018] is a question answering model specifically designed for large context sizes. It does this by focussing on paragraphs that are likely to contain the answer

and ignoring the rest. DocumentQA reminds of a traditional pipeline approach but splits the reading comprehension step into paragraph selection and actual reading comprehension.

The paragraph selection depends on the number of context documents. For contexts of one document, the paragraph(s) closest to the question according to tf-idf are selected. For multiple documents a linear classifier selects the document. This is the case for the XQA baseline. The classifier takes into account how close question and candidate paragraph are both through tf-idf and word overlap and has additional features for the position of the paragraph within the context document. The number of the paragraphs selected by the classifier is a hyperparameter. In the XQA baseline it is set to five.

There are some challenges the paragraph selection has to solve: During training it has to know the paragraph that contains the gold answer. As the gold data only contain the answer string, not the answer span, and this string might occur in several paragraphs, this is not trivial. At this step, the model treats all paragraphs that contain the answer as correct. This makes sense because the answer can be extracted from them but they might not include enough other information to actually answer the question. Unlike DISEQuA [Magnini *et al.*, 2003], corpora for which DocumentQA provides answers (including XQA) do not distinguish between correct and unsupported answers. This might be the case because they evaluate automatically while DISEQuA had manual evaluation for span prediction.

DocumentQA's reading comprehension model is shown in figure 3.1. The input is the question and the current paragraph. Both are embedded with pre-trained word embeddings and character embeddings. These two embeddings for both parts are combined and further preprocessed with a bidirectional GRU layer. Question and context are then combined with Bi-Directional Attention Flow (BiDAF) attention [Seo *et al.*, 2017]. In addition to the question-based attention, there is also self-attention on the context paragraph which is independent from the question. Both attention and self-attention are summed up to predict start and end scores for possible answers in the paragraph. The scores are again computed with a bidirectional GRU and a linear layer.

3.4.2 BERT and mBERT

The other two translation baselines are based on a monolingual BERT model with translation at training or test time. The last baseline is a zero-shot baseline with mBERT as described in section 2.3.

3.4.3 Results

For the monolingual English baselines, DocumentQA, English BERT and mBERT perform all close to each other with English BERT as the best (EM: 33.72, F1: 40.51), DocumentQA in the middle (EM: 32.32, F1: 38.29) and mBERT the worst (EM: 30.85, F1: 38.11). Results for cross-lingual experiments both through translation and mBERT are much lower: Only Chinese with mBERT has a similar f-score and even there the exact match is five points lower. There

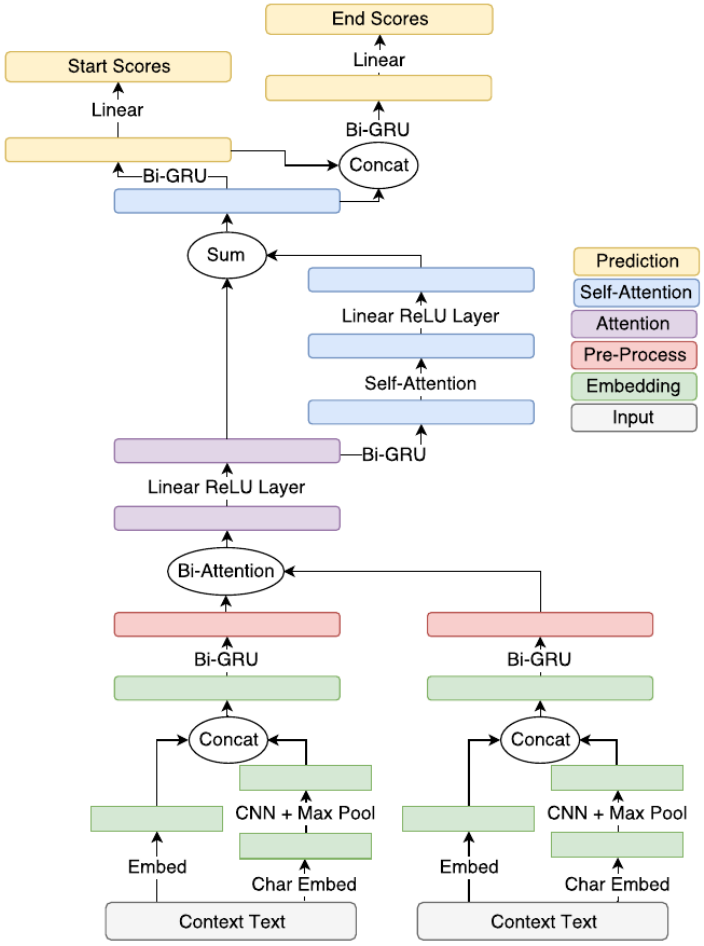

Figure 3.1: Outline of DocumentQA-model as published in Clark and Gardner [2018].

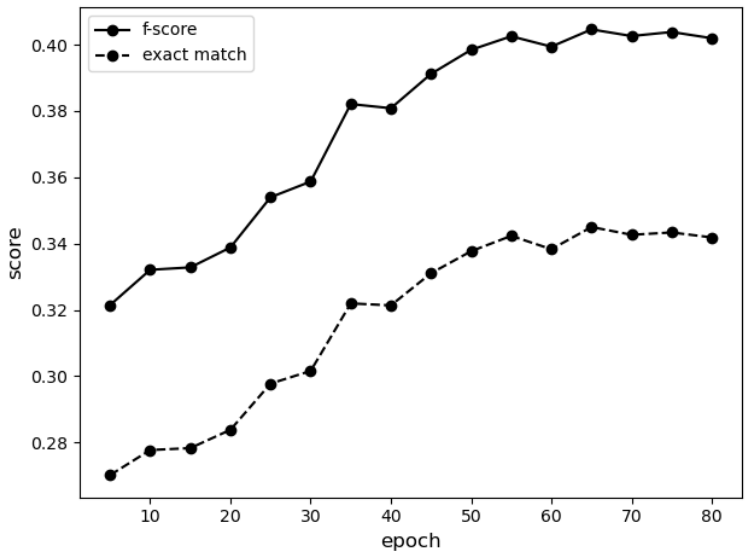

Figure 3.2: Results for DocumentQA baseline on English dev-set every 5 epochs.

Model	Translate-Test				Translate-Train				Zero-shot mBERT	
	DocQA		BERT		DocQA		BERT			
Languages	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
English	32.32	38.29	**33.72**	**40.51**	32.32	38.29	**33.72**	**40.51**	30.85	38.11
Chinese	7.17	17.20	9.81	23.05	7.45	18.73	18.93	31.50	**25.88**	**39.53**
French	11.19	18.97	15.42	26.13	-	-	-	-	**23.34**	**31.08**
German	12.98	19.15	16.84	23.65	11.23	15.08	19.06	24.33	**21.42**	**26.87**
Polish	9.73	16.51	13.62	**22.18**	-	-	-	-	**16.27**	21.87
Portuguese	10.03	15.86	13.75	21.27	-	-	-	-	**18.97**	**23.95**
Russian	5.01	9.62	7.34	13.61	-	-	-	-	**10.38**	**13.44**
Tamil	2.20	6.41	4.58	10.15	-	-	-	-	**10.07**	**14.25**
Ukrainian	7.94	14.07	10.53	17.72	-	-	-	-	**15.12**	**20.82**

Table 3.3: Results of the original XQA baselines. All numbers are from Liu *et al.* [2019a]. The numbers for English translate-test and -train are for untranslated English models.

Language	Liu *et al.* [2019a]		Reproduction	
	EM	F1	EM	F1
English	30.85	38.11	27.20	34.01
German	21.42	26.87	15.56	20.22
Russian	10.38	13.44	06.41	10.93
French	23.34	31.08	18.09	26.34
Tamil	10.07	14.25	02.68	04.36
Chinese	25.88	39.53	15.72	35.16

Table 3.4: Results of mBERT baseline as reported by Liu *et al.* [2019a] and in the reproduction.

are large differences between target languages: The best model for Russian achieves exact match 10.38 and F1 13.44, while the same model for Chinese has 25.88 and 39.53. One trend is that mBERT generally has better results for cross-lingual experiments than translation-based DocumentQA or BERT. The only exception is one setting for Polish and this is fairly close. All in all, the large gap between monolingual and cross-lingual experiments shows room for improvement.

Figure 3.2 shows the evaluation on the English development set after each fifth training epoch for the DocumentQA baseline. Even after five epochs the model is much better than naïve baselines. After that the performance rises continually until it plateaus after around 50 epochs. Liu *et al.* [2019a] give 80 epochs as standard for training. By this time there are clearly no improvements.

While Liu *et al.* [2019a] report exact match 32.32 and F1 38.29, the reproduction had with 32.43 EM and 38.37 F1 nearly the same results in the monolingual English setting. The reproduction of the BERT baseline has consistently lower results (about 5 points) when training for one epoch as suggested in the code (see table 3.4). This difference is stable across languages except Tamil which has a larger drop.

Baseline	Input Data	German		English	
		EM	F1	EM	F1
most frequent named entity	top-1 document	12.76	17.60	22.32	26.92
	top-10 documents	03.65	06.21	03.73	05.71
random named entity	top-1 document	02.54	04.61	03.66	05.67
	top-10 documents	00.51	01.41	00.58	01.23
first named entity	top-1 document	11.99	16.46	17.12	21.30
most frequent noun	top-1 document	10.06	19.00	20.57	29.66
	top-10 documents	01.87	05.27	01.71	05.35

Table 3.5: Results of simple baselines on XQA.

3.5 Simple Baselines

This section presents several simple baselines that try to predict the answer only with basic heuristics. These baselines are meant as a sanity check for the corpus: Is this corpus difficult enough that a QA-system has to actually "understand" the question and context or can it be solved with a trivial shortcut? Naturally, even a system that vastly exceeds trivial baselines does not have to have true understanding but at least it learned "something interesting".

Ideally, these simple baselines should perform significantly worse than the available translation and zero-shot baselines. If they worked, it would be possible that also deep learning based approaches find the underlying heuristics and base their performance primarily on them.

3.5.1 Named Entity Baseline

The first heuristic is *most frequent named entity*: Because answers are only article titles, most of them should be named entities. Just considering named entities instead of all n-grams makes the space of potential answers much smaller. As article title and thus article topic, the answer should also feature prominently in the text. This baseline therefore assumes that the most frequent named entity in the given context is the correct answer. Because the XQA context preselects ten articles, there are two versions of the baseline: *Top-10 documents* gets the same complete context as the neural baselines. This compares similar conditions to the neural baselines but the simple baseline will be thrown off if the wrong context articles are longer than the correct article. *Top-1 document* assumes that the retrieval part worked perfectly and only considers the best retrieved document. Here is a summary of this baseline:

1. find all named entities in the context
2. count their frequency
3. return the most frequent named entity

The second heuristic is *random named entity*. It just takes a random named entity from the context. It works on token-basis instead of type-basis so more frequent named entities are more likely to be selected. Another heuristic takes the *first occurring named entity*. The last heuristic uses *nouns* instead of named entities.

Table 3.5 shows the results for these baselines on the German and English development

	XQA	TYDI	MLQA	XQUAD	XQUAD CONTEXT
EM	22.32	02.42	05.66	03.62	01.26
F1	26.92	04.40	08.45	06.38	03.09

Table 3.6: Comparison of most frequent named entity baseline in English on different corpora.

sets. Generally the baselines work better for English than for German which can be at least partially explained by less morphology. The noun-based heuristic leads to higher F1 scores, but the named entity heuristic leads to higher exact matches. This is expected because the noun baseline only predicts one token while the gold answers are often several tokens long. Therefore, it cannot find exact matches for longer gold answers. F-score also counts partial answers. So a baseline that only finds incomplete answers but does this well, will have a moderate f-score but low exact match. The much better results for top-1 compared to top-10 highlight that the information retrieval step performed well enough that the answer is indeed often in the first context document. Looking at the full context, the baselines are nowhere near the neural baselines which seems to make the dataset challenging. However, taking just the first document as context, the baselines give very good results – still 10 to 15 points lower for English than a neural baseline but better than cross-lingual transfers with neural baselines.

The previous analysis shows that the simple baselines perform well on XQA's English part. To find out if this is just a quirk of XQA or more inherent to QA-datasets, the same baseline is evaluated on other corpora. The comparison of different corpora in table 3.6 only uses the top-1 baseline because the other corpora also only have one context document. This simple baseline achieves much better results on XQA than on the other corpora. This points to some potential problems in the construction of XQA. On its own it might be evidence that XQA is generally easier than other datasets. However, BERT models performing worse on XQA than on the other datasets suggests another reason: The other datasets are built with different mechanisms and don't have mostly named entities as answers e.g. answers from the XQUAD development set include numerals (e.g. *four*) and common nouns (e.g. *patents*). Due to its data collection, XQA has a strong focus on named entities which can be exploited by this baseline.

3.5.2 Overlap Baseline

Word overlap between question and answer or close answer context is also a part of QA-dataset quality. Liu *et al.* [2019a] call questions where the answer occurs in the sentence with the highest word overlap with the question *easy questions*. They compute the fraction of easy questions in XQA and find that this varies between languages: Less than 18% of Tamil or Polish questions are *easy questions* but about a third of Chinese questions. They also find that languages with a higher fraction of easy questions get better results for reading comprehension.

The baseline in this section measures the influence of word overlap by selecting the sentence with the highest word overlap and than taking a named entity from this sentence that is not in the question. This baseline should mostly capture instances where the question was basically a quote from the document with one phrase substituted.

Method	Language	EM	F1
Named Entitiy	en	07.04	09.08
	de	04.52	06.87
Noun	en	03.73	06.35
	de	01.16	02.70

Table 3.7: Results for overlap baselines on the first document of the ten XQA context documents.

Training	Evaluation	EM	F1
with question	with question	32.43	38.37
with question	without question	13.93	15.33
without question	with question	23.78	28.81
without question	without question	26.44	31.62

Table 3.8: Results of DocumentQA-Baseline with and without question information for English.

The results in table 3.7 show that the overlap baselines perform much worse than the simple baselines and are generally in the single digits. This result implies that the lexical overlap between questions and context – even though it is high when measured directly – will not make this dataset trivial. Again the baseline has better results on English than on German. Unlike the simple baseline in the previous section, this baseline benefits from a larger context. The gains are however only 1 to 2 points.

3.5.3 DocumentQA Baseline without Question

This baseline also hinges on the fact that answers are article titles. It tries to solve a different task. Instead of *What is the answer to this question in this text?* it asks *What is the title of this text?* For this, all tokens in the question were substituted with a new token that is not part of the pre-trained embeddings. Just leaving the question empty is not possible because the baseline model assumes a non-empty question sequence and changing this part of the model makes comparison impossible. The neural DocumentQA-baseline is then trained and evaluated both with its original input with questions and this modified input without questions.

On the one hand, table 3.8 shows that the original baseline uses information from the question as performance more than halves if this information is not available. Compared with later cross-lingual experiments this model is still surprisingly good. On the other hand, question information is not generally crucial to solving this dataset: If a baseline model is trained and evaluated with substituted tokens, it performs only 6-7 percentage points worse than a baseline with questions. Evaluating this on the original task leads to a further performance drop due to the mismatch of training and evaluation data. However, it is still in the upper range of Liu *et al.* [2019a]'s reported cross-lingual baselines.

Chapter 4

DocumentQA with Cross-lingual Embeddings

The DocumentQA-baseline in the previous chapter is language-specific and is only adapted for a cross-lingual setting by translation. This is the case because its first layer are mono-lingual word embeddings, GloVe [Pennington *et al.*, 2014].

Word embeddings are semantic representations for words or tokens. They are based on the distributional hypothesis of Harris [1954] that the meaning of a word is defined by the words that it usually occurs with. While earlier representations just counted the occurrence of context words, word embeddings distil this information into a dense floating-point vector. The length of this vector is the embedding dimension. Dense word embeddings have been used successfully for many tasks [Glavaš *et al.*, 2019; Upadhyay *et al.*, 2016]

4.1 Cross-lingual Word Embeddings

Monolingual embeddings map words from one language to a multi-dimensional vector in a feature space. Cross-lingual embeddings map words from two or more languages into the same feature space in a way that words which are translations of each other get mapped to similar vectors. Ideally they would be mapped to the same vector but the alignment is not perfect and words are often not exact translations of each other. Multi-lingual word embeddings also map words from multiple languages but don't require their feature spaces to be aligned.

The embedding spaces for different languages often show similar structures. These structures can be used to align embeddings. While monolingual embeddings are typically trained unsupervised, e.g. through language modelling, cross-lingual mapping can be done in a supervised, semi-supervised or unsupervised way [Ruder *et al.*, 2019]. Some approaches use cross-lingual information from bilingual dictionaries while others rely only on monolingual data [Artetxe *et al.*, 2018b]. Some forms in between have a mapping dictionary that only uses numbers or untranslated words such as some named entities [Artetxe *et al.*, 2017].

Drawbacks of cross-lingual approaches based on word embeddings are that word embeddings are needed for every language in the system. Ideally these embeddings are already pre-trained

and aligned for source and target language(s). This is usually available for a pair of high-resource languages. Getting alignments for all languages will be harder for models that should work for many languages. Furthermore, every additional language needs additional memory for its embeddings. Besides independent token embeddings there are also approaches with contextual embeddings [Schuster *et al.*, 2019b].

In a QA-context, Da San Martino *et al.* [2017] use cross-lingual embeddings for question re-ranking. Their task is to find old questions in a web forum that are similar to a newly posed question in case one of the old questions already has a posted answer that is also relevant to the new one. The new questions are in Arabic and the old questions in English.

In the experiments in this chapter, DocumentQA is modified to support cross-lingual word embeddings to facilitate language transfer. Here follows an overview of different cross-lingual word embeddings and compares which of them are best suited to be integrated into the *DocumentQA* model. FastText embeddings [Joulin *et al.*, 2018] with alignments between 44 languages were not included in the comparison.

4.1.1 Polyglot

Released in 2013 by Al-Rfou *et al.* [2013], Polyglot embeddings[1] are one of the oldest multilingual word embeddings and cover 117 languages. The language selection stems from including all languages that at the time of training had at least 10,000 Wikipedia articles. With Wikipedia as training corpus, Polyglot word embeddings are also from the same domain as the Xqa corpus. The vocabulary per language is up to 100,000 words which is much less than e.g. GloVe with about 2,000,000 words for English.

Unlike other word embeddings (e.g. Muse), Polyglot distinguishes between lower- and uppercase versions of words to preserve more linguistic features. In English this helps distinguishing names from proper nouns, in German it quickly identifies nouns. This is especially helpful for one of their first test cases: POS tagging. When preprocessing a corpus for Polyglot embeddings, this should be taken into account.

Polyglot embeddings are based on the monolingual Senna embeddings [Collobert and Weston, 2008]. They are trained by learning to distinguish between an original phrase and a corrupted phrase.

The most prominent advantage of Polyglot embeddings is their large language set. They include 2-3 times as many languages as other large embedding sets. Their disadvantages are that they are comparatively old and have only small vocabularies. A crucial point is also that they are multi-lingual embeddings but not cross-lingual embeddings. To use them in cross-lingual experiments, they have to be aligned first.

[1] `https://sites.google.com/site/rmyeid/projects/polyglot`

4.1.2 MUSE

MUSE embeddings were introduced in 2017 by Conneau *et al.* [2017]. Pre-trained embeddings in 30 languages are available[2]. They are trained on Wikipedia texts – the same domain as the XQA-corpus – and only on monolingual texts. Their training method consists of two steps: The first step is adversarial training with a discriminator that distinguishes between the source and target embeddings and a mapping that is trained to fool the discriminator. In the second step a synthetic dictionary is extracted from this mapping and the model is fine-tuned.

With a vocabulary size of 200,000 they are in the mid-range of embedding sizes and have 300 dimensions which is a standard size for word embeddings. The embeddings are all lower-case but have no stemming or lemmatization. Some pre-processing errors like the token `,,renaissance` are present. Initially MUSE seems better suited for this task than POLYGLOT because it is larger and newer. Still, the vocabulary is only a tenth of the monolingual GLOVE embeddings used in the baseline. MUSE embeddings were for example used by Conneau *et al.* [2018] for natural language inference and Schuster *et al.* [2019a] for utterance interpretation in dialogs.

4.1.3 VECMAP

VECMAP embeddings are cross-lingual embeddings that were created by aligning monolingual embeddings with a range of methods described in Artetxe *et al.* [2018b,a] inter alia. There are supervised, semi-supervised and unsupervised methods to create VECMAP embeddings. The experiment in this chapter uses supervised embeddings.

Unlike POLYGLOT and MUSE embeddings, they are not released as pre-trained embedding files but Artetxe *et al.* [2018b] published training scripts along with download links to all necessary training resources:[3] monolingual embeddings and mapping dictionaries. Supervised training with this script takes less than 10 minutes on CPU. The mapping dictionary `en-de.train.shuf.txt` with 5,000 entries was used for alignment.

With a vocabulary size of 200,000, the used VECMAP embeddings are in the same range as the MUSE embeddings.

On the one hand, VECMAP provides flexibility as new language pairs can be trained as needed. On the other hand, there is always training needed before embeddings can be used. Also, the two-way alignment does not produce embeddings for systems dealing with more than two languages.

4.2 Experiments

The embeddings used for training and evaluation have to have the same dimensions, otherwise the model cannot load the embeddings at test time. To save space the model usually takes only words from the pre-trained embedding file that actually occur in the corpus during training

[2] `https://github.com/facebookresearch/MUSE`
[3] `https://github.com/artetxem/vecmap`

and uses the same embeddings while testing. This means the embeddings used in the model are only a subset of the embeddings from the resource and without knowing the training corpus it cannot be determined how large this subset will be. Additionally, the used embeddings and their size only depend on the training and not on the evaluation.

There are three possible solutions: Using shared word embeddings both during training and test time, fixing the embedding size with a new hyperparameter or substituting each training embedding with its translation.

The first solution seems easiest but has several problems. Without changing the embedding pruning, all words from the target language would be pruned out as they don't occur in the training set. That way, even though the pre-trained embeddings were from two languages, the embeddings effectively used in the model are monolingual. If we relax the pruning, the embedding size increases drastically and the model becomes too large. While a model with pruning will use embeddings for around 120,000 words if they are available, a model without would use embeddings for all words in the resource for at least two languages. In the case of MUSE and the pair English-German this would be 400,000 words. This is especially true if training a cross-lingual model for more than two languages. As the model grows linearly with each target language it is unsuitable for a larger number of language pairs. Another problem of this solution is its inflexibility. The target language embeddings have to be already available during training and it is impossible to incorporate another target language later.

The second solution introduces an additional hyperparameter: the embedding size of the model. It determines how many embeddings the model uses. How do we choose for how many words we should have embeddings? And what do we do if source and target language aim for different embedding sizes – either because they have different amounts of pre-trained embeddings or because a different fraction is present in the data? These different sizes can be levelled by padding the smaller embedding space and/or pruning part of the larger embedding space. For padding, we use dummy words called `PAD_<number>` with arbitrary embeddings as the words should never occur in the corpus. For pruning, we discarded the n last word embeddings. As pre-trained word embeddings are often ordered by frequency we discard infrequent words.

The third solution – substituting each training embedding with its translation – has difficulties dealing with polysemic words. If there are several common translations for a word, which one should we choose? And if we decide on one, its vector will be different from the mixed-meaning vector in the source language. Therefore, the hyperparameter solution is used in the embedding experiments.

There is also another tweak to the original DocumentQA model. DocumentQA falls back on an embedding for a lowercase word if there is no embedding for an upper case word. Because some of the cross-lingual embeddings also include uppercase tokens (see POLYGLOT), the adjusted model also implements the other direction: If there is no embedding for a lowercase word, the model now falls back on a potentially existing embedding for the upper case spelling.

Table 4.2 shows results for different cross-lingual embeddings. Even in the mono-lingual case English-English MUSE and VECMAP embeddings perform worse than GLOVE embeddings.

Embeddings	English		German	
	EM	F1	EM	F1
GloVe	32.43	38.37	02.56	05.17
Muse	28.64	34.04	09.40	12.56
VecMap	28.70	34.04	06.77	09.66
Polyglot	27.61	33.07	03.19	06.06

Table 4.1: Results for DocumentQA baseline with different embeddings. GloVe has monolingual English embeddings for both languages. Muse and VecMap have cross-lingual embeddings in the respective languages. Polyglot are unaligned multi-lingual embeddings.

This is probably because of the smaller embedding size. GloVe contains pre-trained embeddings for 2,196,017 words whereas Muse and VecMap only have embeddings for 200,000 words. Not all of the pre-trained embeddings actually occur in the evaluation set, but the difference there is still notable: 81,261 words (56%) with embeddings for Muse versus 119,917 (83%) for GloVe. This difference is even more pronounced for Polyglot with only 93,571 (German) and 91,268 (English) pre-trained vectors.

For German in the cross-lingual transfer condition, cross-lingual embeddings perform better (in the case of Muse twice as good as) than English embeddings but are still far worse than translation baselines or the zero-shot mBERT model. The f-score for Muse on the German dev-set is only 12.56 compared to 26.87 with mBERT. While the English evaluation is fairly close for all three new embeddings, there are differences in the transfer to German. There, VecMap is half-way between Muse, the other cross-lingual embedding set, and Polyglot that has no cross-lingual alignment.

So, why are the models with cross-lingual embeddings so bad? While cross-lingual embeddings don't transfer perfectly and thus introduce another possible point of failure, they perform reasonably well on other NLP tasks. One factor is the difference in embedding size. This can explain the drop of 4 points (exact match and f-score) in the monolingual English setting. Not only the vocabulary coverage has an influence; also the total size of the embedding layer including words that are never seen in the training data is important. The original DocumentQA-baseline that prunes embeddings for unseen words has an exact match of 32.43% and an f-score of 38.37 for monolingual English. With the adjusted Document-QA baseline, that fixes the embedding size for GloVe around the size of the pre-trained embeddings but otherwise has the same settings, this increases to 34.18% exact match and 40.19 f-score.

However, embedding sizes can only explain the four-point-difference already present in the monolingual setting. The majority of the drop needs another explanation. Artetxe *et al.* [2020a] observe the same phenomenon with their CLWE model (Cross-lingual word embedding mappings). Their model uses cross-lingual skip-gram embeddings as input to an English BERT model. The embedding layer is frozen during training and fine-tuning. Their evaluation tasks are natural language inference, document classification, paraphrase identification and question answering on Xquad. They comment how CLWE performs well on smaller, easier tasks but the gap to more elaborate models widens for more complex tasks like question answering. Still, at least for languages related to the source language English the gap between CLWE and the

best performing models is substantial but not as large between the DocumentQA + crosslingual embeddings and DocumentQA + translation.

One important disadvantage of word embedding approaches for more complex tasks is a difference in syntax. Cross-lingual word embeddings can transfer vocabulary from the source to the target language but all parts of the model that can take word order into account, namely the deeper layers of the model, are fixed.

Chapter 5

Additional Target Language Training Data

Zero-shot approaches assume that there is no target-language training data. While large training corpora are hard to find for languages besides English and (maybe) Chinese, smaller datasets might be available or possible to create for a task. This observation is similar to Artetxe *et al.* [2020b]'s claim about parallel training data but for annotated task-specific data. This chapter explores how large additional target language training data has to be to improve substantially over zero-shot transfer approaches.

Because the XQA corpus only includes English training data, we first have to find additional, similar training data. All the experiments will be conducted on the language pair English-German.

There are several possible ways to collect additional training data. The first is to use the same approach with which the original corpus was collected: To **scrape data** from the *Did you know*-section of Wikipedia. This would result in training data with the same quality and format as the original corpus. However, there does not exist enough raw data to be scraped. In the German version of Wikipedia there are fewer *Did you know*-questions than in the English version and most of them are already included in the dev or test set of XQA. Only the new questions that have been written after XQA was collected could be used as training data. From November 2018 to August 2020, there are ca. 1300 new questions while the XQA training set contains over 50,000 questions. This limited data is used in the following experiments.

A second approach would be to **translate** the English training data. This would be more similar to the translation baseline. (But not the same because the translation baseline does not train on the English training data). Again, the corpus would have the same format. The drawback of this approach is, that we would not train on natural but on translated texts (see chapter 3 for why this a problem).

A third approach uses the same link extraction method as XQA but applies it to **regular Wikipedia articles**. This ensures original language from the same domain but the data has a slightly different format. The bigger challenge is to ensure that the answer can be answered from the context.

The fourth approach is a mixture of **translation and data scraping**: The English queries are translated to German but the context stems from German Wikipedia articles which has parallels to the collection of MLQA. As Saleh and Pecina [2020] show in their research on medical texts, query translation is much less error-prone than document translation. This approach is used in the following experiments.

5.1 Scraping German Training Data

To find suitable contexts, we look up the English Wikipedia article for each answer in each answer set[1]. One answer in the answer set should correspond to the matching article title. In fact for 99.70% of questions in the development set a matching article is found.

For the 27 missing questions, the article has disambiguation brackets that the dataset omits (*Operating Passenger Railroad Stations Thematic Resource* vs. *Operating Passenger Railroad Stations Thematic Resource (New Jersey)*) or is only a redirection to another article. Another error source is that Wikipedia changes constantly: One article was recorded as missing on June 9th 2020 but a corresponding article was created on June 25. In the other direction, one article seems to have been deleted since the original dataset was collected.

From the English Wikipedia article the corresponding German Wikipedia article is linked if it exists. For about half the questions (dev: 55.02 %, train: 51.40 %) there is a German version of the article.

Questions and answers are translated automatically with a Google Translate API[2]. For about 27% (both dev and train) of questions, the German article title is not in the translated question set. There are several causes for this mismatch: One is that the article title sometimes contains additional disambiguation information in brackets that was stripped from the answers. Given that in only 22% of cases none of the translated answers is found in the article, this probably accounts for nearly a fifth of mismatches. Other mismatches occur when proper names were translated, e.g. *Shukria Barakzai* was translated to *Vielen Dank, dass Sie Barakzai*. For these cases (which make a large portion of mismatches), it would be better to leave the answer set untranslated. As a compromise, the untranslated answer is also added to the answer set. This might be introducing noise but in most not-proper-name cases the English words should not appear in the German context.

This method also introduces some additional noise, that is, it has lower quality than the English training data. How much this quality matters is shown in table 5.1 which compares fine-tuning with high-quality German data from approach 1 (scraping new questions from Wikipedia) and low-quality training data from this approach.

Figure 5.1 shows an example where the target article extraction went wrong: One of the answer in the answer set, *Eugenia*, is both the first name of the person the question is looking for and the name of a plant. Because the person does not have a German Wikipedia article, the article for the plant is returned.

[1] with `https://pypi.org/project/Wikipedia-API/`

[2] `https://pypi.org/project/googletrans/`

```
question:
    "<Query> included men on the board of the Romanian women's
    suffrage association that she founded because she believed
    their skills would help the cause?",
answers: ["Eugenia", "Ianculescu", "Eugenia de Reuss Ianculescu"],
source_title: "Eugenia de Reuss Ianculescu",
target_title: "Kirschmyrten",
target_text: "Die Kirschmyrten (Eugenia), selten auch Eugenien genannt, ...
```

Figure 5.1: Example of extracting a wrong target language article

The XQA context consists not of only one but of ten documents. For every question, the 10 documents with the highest BM25 score are selected from the previously harvested documents. For better comparison with the XQA corpus, we first tried selecting the 10 closest documents from the entire German Wikipedia dump but compared to the expected gains, the process would have been too time-consuming. The BM25 score for ranking is computed with *OkapiBM25* from the python library rank-bm25 [3]. We created the scraped corpus both in a version where the original document was always part of the context documents (`XQA_scraped_*_gold`) and in a version where strictly the 10 closest documents are included (`XQA_scraped_*`). That is, if the correct article somehow has a lower BM25 score than 10 other articles, there are unanswerable questions. An alternative would have been to select the German version of the 10 included XQA-articles if they exist but that would have required a second solution for missing documents. As in the original corpus, the first paragraph of every context document was removed.

With this approach there are about 28,000 questions left (21,000 if we exclude questions where the answer is not found in the article). From this pool, subsets of several sizes are selected randomly: 5,000 QA-pairs, 1,000 QA-pairs and 500 QA-pairs.

To compare this noisy "half-translated" data with "good" training data, data was also scraped with approach 1. This includes gathering all questions from `https://de.wikipedia.org/wiki/Wikipedia:Hauptseite/Schon_gewusst/Archiv` subpages between November 2018 and August 2020, extracting the linked article as one context article and the link text as one gold answer. After that the corpus was created similarly to approach 4.

5.2 Experiments with Additional Target Language Data

There are two possible ways of incorporating the target language training data. In one the model is first trained on source language training data and later fine-tuned on target language training data, the other trains jointly on source and target language data. Advantages of the former are that it is still possible if one has only access to the trained model but not the original training data and that the effort of training the source model – which needs more data and thus takes longer to train – has to be done only once and can then be used for all target languages.

Experiments with the additional training data are performed on two models. First on the

[3] `https://pypi.org/project/rank-bm25/`

Additional Training Data type	size	Epochs	EM	F1
-	-	0	**09.40**	12.56
approach 4, gold	500	5	08.66	11.62
	1000	5	08.75	11.88
	5000	5	**09.48**	**12.84**
approach 4, gold	1000	5	08.75	11.88
		10	08.96	12.19
		15	**09.40**	12.56
		20	09.22	12.55
		25	09.35	12.70
		30	09.33	12.67
		35	**09.39**	**12.82**
approach 1	1000	5	**09.40**	12.50
		10	**09.40**	12.50
		20	08.47	11.59

Table 5.1: Experiments with English-trained DocumentQA-model as basis and different amounts of German data. All are evaluated on the XQA German development set. The marker *gold* refers to the corpus version where the original document is always part of the context.

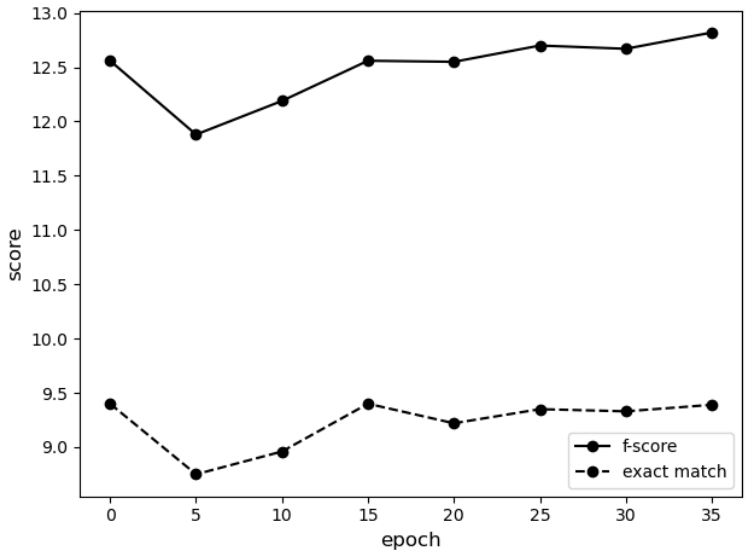

Figure 5.2: Results of DocumentQA-model pre-trained on the full English training data evaluated after every five epochs of training on 1000 German QA-pairs.

Language	monolingual		bilingual 1000		bilingual 5000	
	EM	F1	EM	F1	EM	F1
English	27.20	**34.01**	18.38	24.16	**28.33**	**34.01**
German	**15.55**	**20.22**	11.22	14.49	14.58	18.82
Russian	06.40	**10.93**	05.57	08.68	**07.49**	10.71
Tamil	**02.68**	**04.36**	02.34	**04.40**	02.10	03.30
Chinese	15.71	**35.16**	12.56	25.23	**17.38**	31.83
French	**18.08**	**26.34**	13.26	18.77	15.11	21.83

Table 5.2: Results for BERT trained on English XQA training data and 1000 or 5000 German QA-pairs.

modified DocumentQA-model from chapter 4 and in the next section on mBERT to compare also to languages for which no shared embeddings are available. The experiments in table 5.1 want to answer three questions: how important is the size of the additional data? How long should the additional training be? And how important is the quality of the additional data? If one wants to adapt a QA-model to a new language, one might not get training data that is equally good as the original training data. Table 5.1 shows the results for different sizes of German training data, different lengths of training and both data gathering approaches. While there are small differences between the configurations, they are basically the same or slightly (less than a point) worse than the configuration without additional training data. Thus, training data in the amounts that are feasible to gather for target languages is not helpful. As the results are equally bad for all configurations, the three initial questions cannot be answered from them.

5.3 How does Bilingual Training Help with Third Languages?

Anastasopoulos and Neubig [2020] show that including more languages can help for bilingual lexicon induction. The experiment in this section investigates if this is also the case for cross-lingual question answering. The idea is if and if so, how, training on two languages helps a third language compared to monolingual training.

Table 5.2 shows that monolingual training works best in term of f-scores. This is even the case for German, the language of the additional data. Unlike the DocumentQA-models where additional training data just didn't lead to an improvement, here the performance actually drops with German training data. One observation is that training with the larger additional training set is better than with the smaller. Both models were trained for an additional epoch, so training with the larger set also means training longer and thus more opportunity to weight changes. This stands in contrast to the fact that additional training is harmful in general.

In terms of exact matches, additional training seems good for some languages and bad for others with no clear pattern (e.g. script, relationship to English, fraction of easy questions) to explain this. All in all, there are surprisingly large difference between similar configurations. This opens also the possibility that the performance on BERT is instable and the differences

seen in the table are just random. Weak evidence against this randomness is that the strong differences between the languages track the differences in Liu *et al.* [2019a]'s BERT baseline.

The other side of the coin is how training on additional target language data affects the performance on the source language. This concern was among others voiced by Hardalov *et al.* [2019]. After training DocumentQA for 80 epochs – the full original training cycle – on XQA-SCRAPED-5000-GOLD the performance for English drops dramatically to 22.34% exact match and an f-score of 27.38. However, such a long training procedure is typically not needed because the performance on the target language plateaus earlier. After 5 epochs, when this plateau is usually reached, the results (EM: 28.64, F1: 34.04) are the same as without German training data. 5.2 shows that surprisingly the shorter additional training of mBERT hurts English performance dramatically, while the longer training even boosts it slightly. Taken together, it is more likely that additional training on mBERT is a risk, that may lead to adverse effects.

Chapter 6

Transferability on Other Corpora

This chapter investigates how well models trained on one cross-lingual QA-dataset generalize to other datasets in the same domain. High transferability would be a sign that a model actually learns facets of questions answering and not just narrowly how to "solve" one dataset. Low transferability would be an indicator of overfitting to one corpus and very specific subtask.

In addition to that, different ease of transfer between different corpora shows which corpora are similar to each other.

6.1 Corpora Conversion

To run experiments on the same models, the different corpora have to be in the same format. Thus, the other corpora (XQA, MLQA and XQUAD) cannot be used directly for XQA models. They first have to be converted to the XQA format.

The first step for all three corpora conversions is reading in the original corpus and extracting items with the following attributes: question text, gold answer text(s), context document, question id and document id. In a second step this item list is saved in the new format.

TYDI originally has all languages in one file. This is great for the language-agnostic QA-models for which TYDI was developed. However, it doesn't allow for a separation between training and test languages for cross-lingual transfer. It is also harder to treat languages different during pre-processing, e.g. by using different taggers or embeddings. One step of the TYDI-to-XQA-conversion is therefore to split the large training and test files into separate files for each language.

Additionally, TYDI gives only answer spans from the context. Therefore, the item-extraction step includes cutting the answer text from the context document. Note that the answer spans are counted in bytes instead of characters. TYDI also has a substantial portion of unanswerable questions. For them the answer text is set to *NULL*. TYDI has several annotations of a possible answer for every question. The set of annotations is converted to the set of gold answers.

MLQA includes both the answer text as well as the answer start. For the conversion just the answer text is used. It only has one gold answer for each question. For datasets with many questions about the same paragraph like MLQA the conversion means duplicating this

Corpus	language	EM	F1
XQA	en	**28.64**	**34.04**
	de	09.40	12.56
	ru	01.77	03.13
TYDI	en	02.93	05.80
	ru	00.63	01.30
MLQA	en	06.18	10.60
	de	04.69	10.80
XQUAD	en	07.08	12.28
	de	04.06	09.29
	ru	02.45	07.88
XQUAD CONTEXT	en	03.18	06.06
	de	01.50	04.07
	ru	01.20	03.64

Table 6.1: DocumentQA model trained on XQA and MUSE embeddings.

paragraph to save with every question. This results in larger data files that don't contain more information.

XQUAD splits the original Wikipedia documents into paragraphs and then asks questions about a specific paragraph. To be closer to the long contexts of XQA, this is modified in one version of the converted corpus: In addition to the simple conversion (called XQUAD in the tables), there is also XQUAD CONTEXT that uses the entire document reconstructed from the paragraphs as context.

6.2 Experiments

The following tables show how well models transfer to other cross-lingual QA-corpora. Comparing the raw numbers between different corpora is not possible due to the different traits of the corpora but it can show some tendencies. The comparison is clouded by the different difficulties of the datasets as partially indicated by context size, word overlap or human performance. Corpora without translations (e.g. XQA, TYDI) also give no guarantees that the evaluation sets in different languages have the same difficulty.

The different corpora include different sets of languages. The comparisons include English as the source language and monolingual baseline which is also available in all four datasets. German is also part of the comparison because of the experiments in previous chapters. It is part of XQA, MLQA and XQUAD. The third language is Russian to include cross-lingual experiments for TYDI. It was selected because it is part of three of the four datasets and has pre-trained cross-lingual word embeddings. A bonus point is that is uses a different script than the source language English.

Table 6.1 shows that training on XQA does not transfer well to the other corpora. For all of them, exact match scores drop into the single digits and sometimes nearly to zero.

That transferability is not generally infeasible, can be seen in table 6.2. A DocumentQA-

Corpus	Language	DocumentQA		mBERT	
		EM	F1	EM	F1
Tydi	en	14.78	19.71	17.26	24.78
	ru	04.70	08.35	08.62	18.77
Xqa	en	03.54	05.96	10.94	14.36
	ru	00.14	00.57	04.15	05.85
	de	01.67	02.76	08.37	11.57
Xquad	en	12.89	20.74	**35.05**	**46.70**
	ru	05.73	12.58	24.54	38.43
	de	05.25	12.36	24.62	37.21
Xquad Context	en	05.56	09.20	27.04	37.29
	ru	02.15	04.96	16.02	25.33
	de	01.86	05.01	17.01	26.18
Mlqa	en	**15.16**	**24.75**	31.45	44.91
	de	07.23	15.24	21.68	33.73

Table 6.2: Models trained on English Tydi.

model trained on Tydi performs even slightly better on Mlqa. On the other hand, Xqa and Xquad Context – with larger contexts – achieve again much lower results. This means that both the transferability from Xqa to other corpora as well as the other direction – transferring from other corpora to Xqa – is not good. An mBERT-model also trained on Tydi produces generally higher results but the gap is much larger in the out-of-domain corpora. Again, Xqa has the lowest results but Xquad Context is much better. That three of the corpora Xquad, Xquad Context and Mlqa get much higher scores than the training corpus Tydi, might be just because they are inherently easier but also because they get a substantial part of their training already from the pre-trained model, not just from training with labelled data. This would also explain the gap between DocumentQA and mBERT in the transfer settings as DocumentQA has no pretraining.

It is also noticeable that DocumentQA trained on Tydi only has moderate results even for the monolingual, in-domain setting. This might be because DocumentQA is selected well for Xqa but not suited for datasets with less relevant context.

All in all, the difference between training language and target languages is visible not only in the training corpus but also in the out-of-domain corpora. Also, mBERT achieves higher results than DocumentQA in the transfer settings.

Chapter 7

Tagged BERT

The previous chapter showed that XQA has a special structure which complicates transfer. The way XQA's answers are selected, linguistic features already give clues for possible answers. The simple baselines in chapter 3 only rely on them. Do neural models also use these cues? This chapter tries to answer this question by pre-computing features and giving them to a neural model, mBERT, as input. This way, it does not have to figure out these cues on itself. A boost in performance would indicate that mBERT cannot utilize all of the cues on its own and can be improved with shallow syntactic knowledge. No performance change would imply that mBERT already used this information. A drop in performance would show that this additional information interferes with mBERT's answer finding process.

Two kinds of information could be particularly useful to the model to exploit these cues: part-of-speech tags (POS) and named entity tags (NE). Both are tagged by SpaCy[1] with language-specific models. The POS-tags are inserted after each token. The NE-tags are new tokens that are wrapped around named entities. The tagging is done in the last pre-processing steps of the training script. Figure 7.1 gives an example of the tagging format.

The results in table 7.1 show that POS-tagging does not work at all. The model doesn't find a single exact answer, not even through chance. NE-tagging produces very bad results. A possible cause for these results is that BERT was pre-trained as a language model with plain text. The tagging mark-up is an input format it has not seen during its pre-training phase. So the model has to deal with a new kind of input it cannot use its language model for. While

[1] https://spacy.io/

```
["%%DOCUMENT%%", "Geschichte", "%%PARAGRAPH%%", "Das", "Binion",
"###START###", "'", "s", "Horseshoe", "###END###", "in",
"###START###", "Las", "Vegas", "###END###", "war", "von",
"1970", "bis", "2004", "Schauplatz", "der", "###START###",
"WSOP", "###END###", ".", "Seit", "2005", "werden", "alle",
"Turniere", "im", "###START###", "Rio", "All", "-", "Suite",
"Hotel", "and", "Casino", "###END###", "veranstaltet", ".", [...]
```

Figure 7.1: Example of Named Entity tagged input for mBERT.

Tagging	Language	EM	F1
part-of-speech	en	0.00	0.01
	de	0.00	0.01
Named entity	en	01.78	06.02
	de	00.77	03.61

Table 7.1: Tagged BERT models for XQA.

other models use tags for BERT they are much rarer – in particular than the POS-tags which come after every token. This also explains the difference between both tags. The tags are not an enhancement but noise that disturbs the model, so the version with fewer tags – less disturbance – is better.

An option to investigate the effectiveness of syntactic information without disturbing the model could be to give mBERT this information in a different form, e.g. by leaving the input text as it is and having a separate input for named entities or POS tags.

Chapter 8

Paragraph Selection

The paragraph selection step chooses a pre-defined number of paragraphs from the context documents that are likely to contain the answer and forwards only those paragraphs to the reading comprehension step. Which paragraphs are chosen is determined by a ranker which gives each paragraph a score with regard to the current question. DocumentQA comes with three rankers: a linear classifier, tf-idf and the first n paragraphs (truncate) (see section 3.4.1).

These rankers were designed for different corpora and might not capture the characteristics of the XQA-corpus. In particular, the linear ranker contains features for the first paragraph in each document and the position of a paragraph in its document. However, the actual first paragraphs are removed from XQA so this feature will have a different meaning there. Besides, the only ranker that takes the document order from a previous information retrieval step into account is the truncate ranker with a hard cut-off: The truncate ranker is similar to a information retrieval step with fewer documents and no additional paragraph selection.

Therefore, the three DocumentQA-rankers are compared with two new rankers: *reweighted* is the linear ranker without the features that describe the position of the paragraph in the document – if it is the first and the actual position. So it only has three features: tf-idf, cased word-overlap and uncased word-overlap. *Tf-idf + truncate* is an interpolation between the tf-idf ranks and the rank in the paragraph list as retrieved in the previous step. This favours the best document from information retrieval (and earlier paragraphs in this document) but still

Ranker	Language	EM	F1
linear	en	28.64	34.04
	de	09.40	12.56
tf-idf	en	27.96	33.00
	de	08.93	12.22
truncate	en	**32.87**	**38.87**
	de	**13.01**	**17.12**
reweighted	en	30.89	36.47
	de	10.06	13.54
tf-idf + truncate	en	06.51	09.72
	de	02.29	03.80

Table 8.1: Different rankers for paragraph selection.

allows lower paragraphs if they have high tf-idf-scores.

The results in table 8 are all obtained with MUSE embeddings. The model was trained with the linear ranker (ShallowOpenWebRanker) and evaluated with the ranker in the first column. The good performance of *truncate* highlights the quality of the information retrieval. That *reweighted* yields better results than *linear* shows that DocumentQA's default ranker is not the best ranker for all datasets and XQA is different enough from TRIVIAQA to benefit from different rankers. In particular, the deleted position features were slightly harmful. However, the results are fairly close. The mix of *tf-idf* and *truncate* is much worse than either of its components alone. This is surprising because both features on their own perform quite well.

Chapter 9

Conclusion and Future Work

The previous experiments show that bridging the step for cross-lingual transfer is still hard for more complex tasks like question answering. The zero-shot mBERT approach which was meant as a baseline works best. The experiments in this thesis compared different approaches for cross-lingual transfer and found that while many improve over using mono-lingual models for a different language, they don't reach the results of the zero-shot mBERT approach. Chapter 5 shows that small amounts of target language training data don't improve over purely mono-lingual training with the used approaches. The findings from these experiments on the XQA-dataset don't necessarily hold for other QA-datasets because there are significant differences between datasets as can be seen in chapter 6.

The experiments in the previous chapters evaluated just performance in terms of exact match and f-score. Evaluating question answering, especially when considering real-world applications, should also take other factors into account, e.g. in which way a systems fails (not finding an existing answer vs. finding a wrong answer) or time and space requirements. Most state-of-the-art QA-models are huge [Lan *et al.*, 2020] and recent work on cross-lingual QA also uses very large models, e.g. XLM [Artetxe *et al.*, 2020a]. It would be interesting to see if methods that reduce these requirements such as the DeFormer [Cao *et al.*, 2020] or TinyBERT [Jiao *et al.*, 2019] can also be applied successfully to cross-lingual question answering.

The word embeddings in chapter 4 are all independent word embeddings. If their bad performance compared to mBERT is due to lacking context during the cross-lingual transfer, could be investigated by comparing contextual word embeddings. Cross-lingual contextual are e.g. CrossLingualELMo [Schuster *et al.*, 2019b]. The general DocumentQA-model though not the baseline used here also has a mode for ELMo embeddings.

Also helpful would be a detailed qualitative analysis: Which kind of questions are typically answered correctly and where are common difficulties? Are there differences between the models so that simple baselines answer different kinds of questions correctly than neural models?

Bibliography

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

Antonios Anastasopoulos and Graham Neubig. Should All Cross-Lingual Embeddings Speak English? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8658–8679, Online, July 2020. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019, 2018.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. On the Cross-lingual Transferability of Monolingual Representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online, July 2020. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, Dani Yogatama, Gorka Labaka, and Eneko Agirre. A Call for More Rigor in Unsupervised Cross-lingual Learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7375–7388, Online, July 2020. Association for Computational Linguistics.

Akari Asai, Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. Multilingual extractive reading comprehension by runtime machine translation. *arXiv preprint arXiv:1809.03275*, 2018.

Marco Baroni and Silvia Bernardini. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274, 2006.

Emily M. Bender. On achieving and evaluating language-independence in nlp. *Linguistic Issues in Language Technology*, 6(3):1–26, 2011.

Johan Bos and Malvina Nissim. Cross-lingual question answering by answer translation. In *In Working Notes for the CLEF*. Citeseer, 2006.

Jordan Boyd-Graber and Benjamin Börschinger. What Question Answering can Learn from Trivia Nerds. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7422–7435, Online, July 2020. Association for Computational Linguistics.

Qingqing Cao, Harsh Trivedi, Aruna Balasubramanian, and Niranjan Balasubramanian. DeFormer: Decomposing Pre-trained Transformers for Faster Question Answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4497, Online, July 2020. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 2020.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 160–167, 2008.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

Ryan Cotterell and Georg Heigold. Cross-lingual character-level neural morphological tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 748–759, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. Cross-lingual machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1586–1595, Hong Kong, China, November 2019. Association for Computational Linguistics.

Giovanni Da San Martino, Salvatore Romeo, Alberto Barroón-Cedeño, Shafiq Joty, Lluís Maàrquez, Alessandro Moschitti, and Preslav Nakov. Cross-language question re-ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1145–1148, Tokyo, Japan, August 2017. ACM.

Jeanne E. Daniel, Willie Brink, Ryan Eloff, and Charles Copley. Towards automating healthcare question answering in a noisy multilingual low-resource setting. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 948–953, Florence, Italy, July 2019. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Haim Dubossarsky, Ivan Vulić, Roi Reichart, and Anna Korhonen. Lost in embedding space: Explaining cross-lingual task performance with eigenvalue divergence. *arXiv preprint arXiv:2001.11136*, 2020.

Oren Etzioni. Search needs a shake-up. *Nature*, 476(7358):25–26, 2011.

Goran Glavaš, Robert Litschko, Sebastian Ruder, and Ivan Vulić. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 710–721, Florence, Italy, July 2019. Association for Computational Linguistics.

Momchil Hardalov, Ivan Koychev, and Preslav Nakov. Beyond english-only reading comprehension: Experiments in zero-shot multilingual transfer for bulgarian. In *Proceedings of Recent Advances in Natural Language Processing*, pages 447–459, 2019.

Zellig S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, and Ming Zhou. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494, Hong Kong, China, November 2019. Association for Computational Linguistics.

Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. R4C: A Benchmark for Evaluating RC Systems to Get the Right Answer for the Right Reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750, Online, July 2020. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

Yimin Jing, Deyi Xiong, and Zhen Yan. Bipar: A bilingual parallel dataset for multilingual and cross-lingual reading comprehension on novels. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2452–2462, Hong Kong, China, November 2019. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations*, 2020.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*, 2019.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, et al. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy, July 2019. Association for Computational Linguistics.

Jiahua Liu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. XQA: A cross-lingual open-domain question answering dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2368, Florence, Italy, July 2019. Association for Computational Linguistics.

Pengyuan Liu, Yuning Deng, Chenghao Zhu, and Han Hu. Xcmrc: Evaluating cross-lingual machine reading comprehension. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 552–564. Springer, 2019.

Ekaterina Loginova, Stalin Varanasi, and Günter Neumann. Towards end-to-end multilingual question answering. *Information Systems Frontiers*, pages 1–15, 2020.

Bernardo Magnini, Simone Romagnoli, Alessandro Vallin, Jesús Herrera, Anselmo Peñas, Víctor Peinado, Felisa Verdejo, and Maarten de Rijke. Creating the disequa corpus: a test set for multilingual question answering. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 487–500. Springer, 2003.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics.

Edoardo Maria Ponti, Helen O'horan, Yevgeni Berzak, Ivan Vulić, Roi Reichart, Thierry Poibeau, Ekaterina Shutova, and Anna Korhonen. Modeling language variation and universals: A survey on typological linguistics for natural language processing. *Computational Linguistics*, 45(3):559–601, 2019.

John Prager. Open-domain question-answering. *Foundations and Trends in Information Retrieval*, 1(2):91–100, 2006.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical*

Methods in Natural Language Processing, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online, July 2020. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, et al. Okapi at trec-3, 1995.

Pedro Rodriguez, Shi Feng, Mohit Iyyer, He He, and Jordan Boyd-Graber. Quizbowl: The case for incremental question answering. *arXiv preprint arXiv:1904.04792*, 2019.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631, 2019.

Shadi Saleh and Pavel Pecina. Document Translation vs. Query Translation for Cross-Lingual Information Retrieval in the Medical Domain. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6849–6860, Online, July 2020. Association for Computational Linguistics.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Robert F. Simmons. Answering english questions by computer: a survey. *Communications of the ACM*, 8(1):53–70, 1965.

Benjamin Snyder. *Unsupervised multilingual learning*. PhD thesis, Massachusetts Institute of Technology, 2010.

Ferhan Ture and Elizabeth Boschee. Learning to translate for multilingual question answering. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 573–584, Austin, Texas, November 2016. Association for Computational Linguistics.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1670, Berlin, Germany, August 2016. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China, November 2019. Association for Computational Linguistics.

Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. Emerging cross-lingual structure in pretrained language models. *arXiv preprint arXiv:1911.01464*, 2019.

Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.