CROSS-LINGUAL OPEN RETRIEVAL QUESTION ANSWERING SYSTEM (CORA)

Project to be submitted in partial fulfillment of the requirements for the degree

of

Bachelor of Technology

in

Computer Science and Engineering

by

Charugundla Vipul 1910110118
Amogh Krishna Padakanti 1910110056
Revanth Siva Sai Ram Balineni 1910110314

Under the guidance of

Dr. Sonia Khetarpaul

SHIV NADAR
INSTITUTION OF EMINENCE DEEMED TO BE
UNIVERSITY
DELHI NCR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECEMBER, 2022

SHIV NADAR
INSTITUTION OF EMINENCE DEEMED TO BE UNIVERSITY
DELHI NCR

NH-91, Tehsil Dadri
Gautam Buddha Nagar
Uttar Pradesh - 201314, India

Tel: +91-120-7170100
+91-120-2662002

Certificate

This is to certify that the project titled 'Cross-lingual Open-Retrieval Question Answering' is submitted by Amogh Padakanti, Charugundla Vipul and Revanth Siva Sai Ram Balineni, CSE, SNIOE in the partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Technology**. This work was completed under the supervision of Dr. Sonia Khetarpaul. No part of this dissertation/report has been submitted elsewhere for award of any other degree.

Signature with date: 7/12/22

Name of Supervisor: SONIA KHETARPAUL

Designation: ASSISTANT PROFESSOR

Affiliation: SHIV NADAR IOE

Acknowledgement

We want to start by sincerely thanking our mentor, Dr. Sonia Khetarpaul, for her unwavering encouragement, persistence, inspiration, passion, and vast expertise. This project would not have been completed without her invaluable advice and committed guidance at each phase. She taught us how to conduct research and how to convey the results as clearly as possible. She taught us the value of studying subjects thoroughly and not settling for knowledge that is only superficial.

It initially looked very difficult for us to comprehend the highly complex field of deep learning from scratch, but her leadership and encouragement throughout the process made the task simple. There is no one else we could have asked for as a mentor and advisor.

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Student: Charugundla Vipul

Roll Number: 1910110118

Signature:

Student: Revanth Siva Sai Ram Balineni

Roll Number: 1910110314

Signature

Student: Amogh Padakanti

Roll Number: 1910110056

Signature:

Contents

List Of Figures

Abstract

In the modern world, it is very easy to find answers to a varied set of questions from multiple resources using search engines but there may be some unanswered questions that are mostly language / culture-specific and most importantly when the questions which are asked in information scarce languages, this is where cross-lingual open retrieval question answering architecture comes into the picture. In the recent past, a lot of research papers were proposed on neural network-based OpenQA models, the challenge with these models is they require a gigantic amount of Training data which is specifically not available in other languages keeping in mind the low-resource languages.

The data set which is being used for this project is derived from the XQA dataset (Jiahua Liu and Yankai Lin, 2019), The dataset consists of a training set of English along with test sets in eight other languages. Besides, we provide baseline systems for cross-lingual OpenQA, including two machine translation-based methods and LSTM(Long Short Term Memory). A detailed analysis by (Jiahua Liu and Yankai Lin, 2019), indicates that the performance of cross-lingual OpenQA is related to not only how similar the target language and English are, but also how difficult the question set of the target language is.

Chapter 1

Introduction and Literature Review

In this Chapter, we discuss the motivation behind choosing this project and also the extensive review of literature which was conducted in initial stages for better understanding and gaining knowledge about existing cross-lingual QA systems.

1.1 Introduction

In the recent past, open-domain question answering (OpenQA), which aims to answer open-domain questions with a large-scale text corpus, has attracted lots of attention from NLP researchers. However, the drawback of these models is their requirement for dedicated labeled data. Collecting and labeling large-size training data for each language is often intractable and unrealistic, especially for those low-resource languages. In this case, it is impossible to directly apply existing OpenQA models to many different languages. To address this problem, an alternative approach is to build a cross-lingual OpenQA system which we call **CORA**. It is trained on data in one high-resource source language such as English and predicts answers for open-domain questions in other target languages. In fact, **CORA** can be viewed as a particular task of cross-lingual language understanding (XLU).

Most cross-lingual models focus on word or sentence level understanding, while the interaction between questions and documents as well as the overall understanding of the documents are essential to OpenQA. The Dataset XQA consists of a training set in English, and development and test sets in English, French, German, Portuguese, Polish, Chinese, Russian, Ukrainian, and Tamil. The training set contains 56, 279 English question-answer pairs along with relevant documents. The development and test sets contain a total amount of 17, 358 and 16, 973 question-answer pairs respectively.

Moreover, baseline systems in this Architecture that use the information of multilingual data from publicly available corpora for cross-lingual OpenQA, including two translation-based methods that translate training data and test data respectively and one mono-lingual method (BERT (Devlin et al., 2019)). The experimental results by (Jiahua Liu and Yankai Lin, 2019), demonstrate that there is a gap between the performance in English and that in the

cross-lingual setting. The LSTM model achieves the best performance in English as the model is fed with huge volumes of English training data, while translation-based methods suffer from the problem of translating name entities to perfection. We show that the performance on the XQA dataset depends on not only how similar the target language and English are, but also on how difficult the question set of the target language is. Based on the results, we further discuss potential improvement for cross-lingual OpenQA systems.

In this project we have built three QA models and have integrated two of them to form our final model where the query is given in english we have translated the query to telugu language and we have used Multi-Lingual Bert for short answer retrieval and we have also taken the query directly in english and have answered the question using BERT. We have used the LSTM model as a Baseline model for short answer retrieval.

Detailed explanation for all QA models which were used in this project will be provided in Chapter 4.

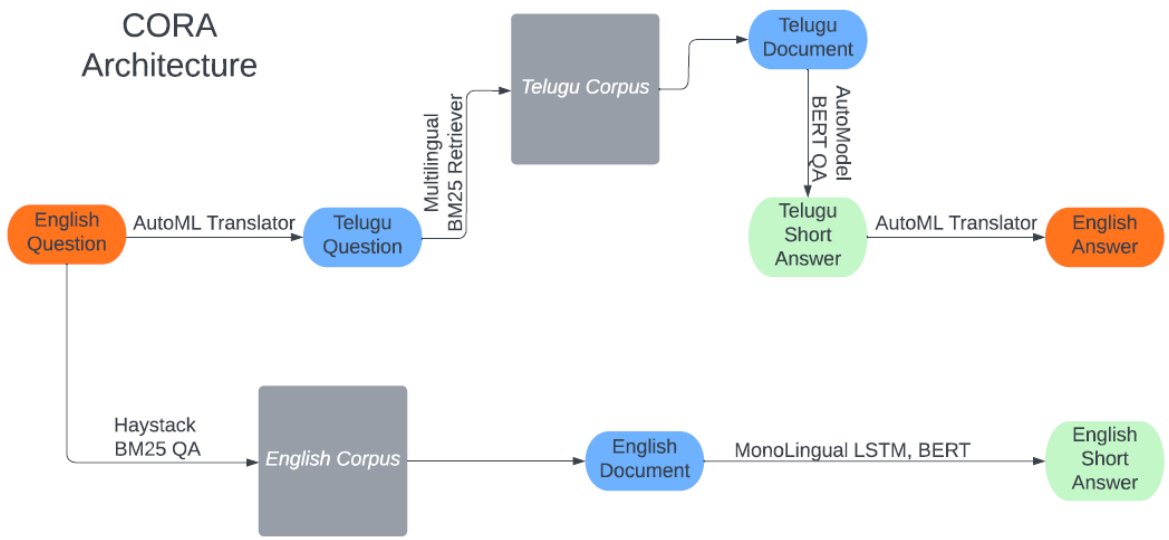


Figure 1: Architecture of final model(CORA)

1.2 Literature Review

As part of our literature review, we have reviewed 3 research papers which were published in the domain of Cross lingual Question Answering.

1) XOR QA: Cross-lingual Open-Retrieval Question Answering (Akari Asai et al., 2021)
2) XQA: A Cross-lingual Open-domain Question Answering Dataset (Jiahua Liu et al., 2020)
3) BCLQA: BERT-based Cross-Lingual Question Answering with DeepPavlov (Vasily Kanovalov, 2019)

XOR QA

Authors of this research Paper have introduced a task framework, called Cross-lingual Open Retrieval Question Answering (XOR QA), which aims at answering multilingual questions from non-English native speakers given multilingual resources. To support research in this area, a dataset (called XOR-TYDI QA) of 40k annotated questions and answers across 7 typologically diverse languages has been constructed by authors of this paper. Questions in this dataset are inherited from TYDI QA (Clark et al., 2020), which are written by native speakers and are originally unanswerable owing to the information scarcity or asymmetry issues. XOR-TYDI QA is the first large-scale cross-lingual open-retrieval QA dataset that consists of information-seeking questions from native speakers and multilingual reference documents. XOR-TYDI QA is constructed with an annotation pipeline that allows for cross-lingual retrieval from large-scale Wikipedia corpora.

Unanswerable questions in TYDI QA are first translated into English by professional translators. Then, annotators find answers to translated queries given English Wikipedia using our new model-in-the loop annotation framework that reduces annotation errors. Finally, answers are verified and translated back to the target languages.

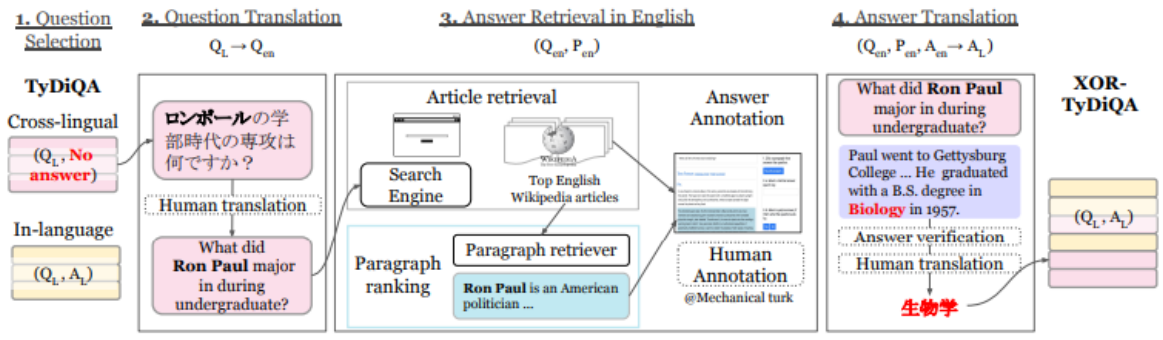


Figure 2: Overview of the annotation process for XOR-TYDI QA. Adapted from [15].

Building on the dataset, authors have also introduced three new tasks:

1) XOR-RETRIEVE
2) XOR-ENGLISHSPAN
3) XOR FULL

In **XOR-RETRIEVE**, a system retrieves English Wikipedia paragraphs with sufficient information to answer the question posed in the target language.

XOR-ENGLISHSPAN takes one step further and finds a minimal answer span from the retrieved English paragraphs.

Finally, **XOR-FULL** expects a system to generate an answer end to end in the target language by consulting both English and the target language's Wikipedia. XOR-FULL is our goal, and the first two tasks enable researchers to diagnose where their models fail and develop under less coding efforts and limited availability of resources.

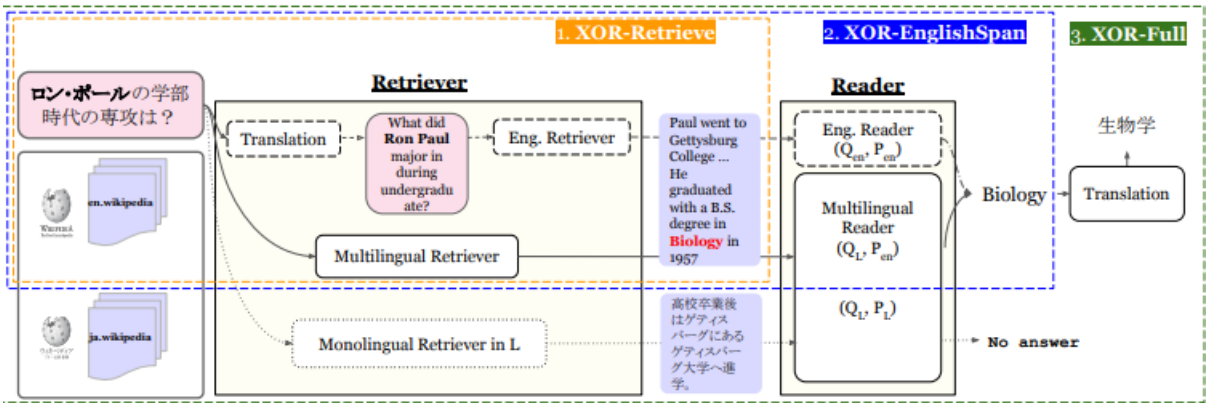


Figure 3: Overview of the tasks and baselines. Adapted from [15].

1.2.2 XQA

In the recent past, open-domain question answering (OpenQA), which aims to answer open-domain questions with a large-scale text corpus, has attracted lots of attention from NLP researchers. However, the drawback of these models is their requirement for dedicated labeled data. Collecting and labeling large-size training data for each language is often intractable and unrealistic, especially for those low-resource languages. In this case, it is

impossible to directly apply existing OpenQA models to many different languages. To address this problem, an alternative approach is to build a cross-lingual OpenQA system. It is trained on data in one high-resource source language such as English and predicts answers for open-domain questions in other target languages. In fact, cross-lingual OpenQA can be viewed as a particular task of cross-lingual language understanding (XLU).

Most cross-lingual models focus on word or sentence level understanding, while the interaction between questions and documents as well as the overall understanding of the documents are essential to OpenQA. The Dataset XQA consists of a training set in English, and development and test sets in English, French, German, Portuguese, Polish, Chinese, Russian, Ukrainian, and Tamil. The training set contains 56, 279 English question-answer pairs along with relevant documents. The development and test sets contain a total amount of 17, 358 and 16, 973 question-answer pairs respectively.

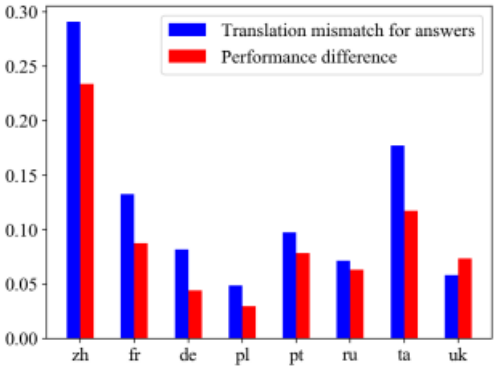
Language	English	Chinese	French	German	Polish	Portuguese	Russian	Tamil	Ukrainian
Avg. question len	18.82	36.83	20.09	14.61	14.49	17.66	14.21	13.29	16.73
Avg. document len	735.91	1159.28	913.72	450.65	256.87	482.74	503.28	200.45	584.93
Avg. paragraph num	10.54	8.66	25.95	8.85	5.34	8.42	10.36	13.78	25.09

Figure 4: Average length of questions, documents and num of paragraphs in various languages adapted from [16].

Moreover, baseline systems in this Architecture that use the information of multilingual data from publicly available corpora for cross-lingual OpenQA, including two translation-based methods that translate training data and test data respectively and one shot cross-lingual (Multilingual BERT (Devlin et al., 2019)). Authors have evaluated the performance of the proposed baselines in terms of text retrieval and reading comprehension for different target languages on the XQA dataset. The experimental results demonstrate that there is a gap between the performance in English and that in the cross-lingual setting.

Figure 5: Performance difference (EM) between translate-test BERT and multilingual BERT. Adapted from [16]

The Multilingual BERT model achieves the best performance in English as the model is fed with huge volumes of English training data, while translation-based methods suffer from the problem of translating name entities to perfection.



It is shown that the performance on the XQA dataset depends on not only how similar the target language and English are, but also on how difficult the question set of the target language is.

1.2.3 BCLQA (BERT-based Cross-Lingual Question Answering)

DeepPavlov is a conversational artificial intelligence framework that contains all the components required for building chatbots. DeepPavlov is developed using some high level open-source machine learning frameworks such as TensorFlow and Keras.

Author of this article has integrated BERT into solutions for the four popular NLP tasks:

1) Text
2) Classification
3) Tagging
4) Question Answering.

Pretrained BERT can be used for question answering over the text just by applying two linear transformations to the BERT outputs for each sub token. The first/second linear transformation is used for predicting the probability that the current sub token is the start/end position of the answer.

The Multilingual BERT (M-BERT) model allows zero-shot transfer from one language (source language) to another (target language). English is a good candidate as a source language, because the training data in English is easily accessible. As a target language you can use any of 104 languages from the list that were used to train M-BERT.

The M-BERT-based model is fine-tuned on one language and evaluates this model on the test set of different languages, thus allowing us to measure the extent to which the model generalizes information across languages.

A model is called by providing the batch of contexts and the batch of questions, and as an output, the model returns the batch extracted from the context's answers with their start positions. This multilingual QA model, while being trained on English datasets, is capable of extracting answers form French context even when the question is asked in a different language (English in the snippet).

Fine-tune/Eval	EN	RU	ZH
EN	**89.11/82.28**	73.76/43.22	61.88/61.88
RU	75.26/54.56	**83.09/64.13**	51.87/51.87
ZH	65.35/55.61	56.93/30.59	**85.07/85.07**

Figure 6: The SQuAD model performance (F1/EM). Adapted from [14]

The results from Figure 6 depict that M-BERT creates multilingual representations, which allows us to achieve promising results in a zero-shot cross-lingual model.

Chapter 2

Dataset and Pre-processing

In this Chapter, we discuss various available open-source datasets, few statistics about datasets and also the pre-processing steps which were carried out to clean data before implementing QA models on training datasets.

2.1 About the Dataset

As far as the dataset is concerned, we have considered 3 existing open-source dataset's which are available online and we will be further discussing technical aspects regarding the datasets in upcoming paragraphs.

1) XQA Dataset
2) XOR-TYDI QA
3) TensorFlow QA Dataset

2.1.1 XQA Dataset

Authors of the XQA dataset (Jiahua Liu and Yankai Lin, 2019) have devised average length of questions and documents in different languages, and the results are shown in Figure 7.

The average question length for most languages falls in the range of 10 to 20. The average question length in all languages is 18.97. The documents on the XQA dataset are considerably long, containing 703.62 tokens and 11.02 paragraphs on average. Documents in Tamil and Polish are among the shortest, with an average length of 200.45 and 256.87 respectively.

As we can see, there are some common topics across all languages, with human ranking first, and film and book ranking high. Besides, many questions in French are related to the commune of France, while the topic battle ranks high in Russian. This indicates that XQA captures different data distributions for different languages, which may be influenced by cultural differences to some extent.

Language	Train	Dev	Test
English	56,279	2,926	2,924
Chinese	-	2,532	2,535
French	-	1,946	1,749
German	-	3,895	3,804
Polish	-	924	922
Portuguese	-	359	348
Russian	-	3,590	3,490
Tamil	-	597	586
Ukrainian	-	589	615

Figure 7: Statistics of the XQA Dataset. Adapted from [16]

2.1.2 TensorFlow QA Dataset

TensorFlow QA Dataset is a highly trained dataset on English Wikipedia and Jsonl files contain below mentioned Data Fields:

1) Document_Text
2) Question_Text
3) Long_Answer
4) Annotations
5) Document_Url
6) Example_Id

TensorFlow QA consists of 17.67 GigaBytes of Training data which is used to feed the Classic Mono-Lingual LSTM Model with Long Answers for each Question enabling the model to precisely find the answer for the questions.

Comparison: TensorFlow QA and XQA Datasets follow quite similar structure and the primary LSTM model in the project will be using TensorFlow Dataset and we will be comparing accuracy and Precision of the model and after evaluating various performance metrics we will decide if it is required to translate Queries in TensorFlow dataset to target language if results using TensorFlow Dataset are quite significant when compared with XQA.

2.2 Pre-Processing

As the dataset is made of vast Wikipedia corpus Preprocessing is done in 3 stages.

1) Removal of stop words.

By getting rid of stop words, we can make our text more focused on the key information by eliminating the low-level information. In other words, we may say that the model we train for our task does not exhibit any negative effects as a result of the removal of such phrases.

Because there are fewer tokens involved in training, the removal of stop words reduces the size of the dataset and, consequently, the training time.

2) Removal of unwanted characters

Special characters do not convey valuable information to the model during training. Their removal helps in converting data into more accurate representation with which we can train model to be more precise.

3) Stemming

Stemming is the process by which suffixes are eliminated from words to produce the "word stem." For instance, the word "like," which is derived from the phrases "likes," "likely," and "liked," can be used as a synonym for all three words. In this manner, an NLP model can discover that all three words share some similarities and are used in comparable situations.

Regardless of their inflections, stemming enables us to normalize words to their underlying stem, which is useful for various applications including text classification and clustering. Regardless of the word form, search engines frequently employ these tactics to provide better results.

Chapter 3

Language Analysis (Telugu)

In this Chapter, we discuss Grammatical aspects of telugu language and how different telugu language is from English, correlation between both the languages and structural analysis of telugu language to get an detailed idea on how to examine sentence structures for smooth translation of Questions and Answers.

3.1 Grammatical Analysis

Because English grammar is extremely rich and vast in volume, only nouns, pronouns, verbs, articles, prepositions, and vibhakti (inflections) are covered in this report.

Verbs are an important aspect of the English language since they indicate the tense of a sentence. Because there is no direct translation for auxiliary verbs in Telugu, they are disregarded. A verb phrase is built with the next verb in mind.

Consider the statement, "Theja is going to see a movie." It contains two verbs: 'is' and 'going.' The auxiliary verb is 'is,' while the subsequent word is 'going.' As a result, 'is going' will be treated as a single verb.

Because there is no direct translation for 'is' in Telugu, the dictionary is constructed in such a way that 'is going' is regarded as a single verb phrase. In this statement, "verbs 'to' and 'watch' are also joined to form a single verb phrase as 'to watch.' The words 'watch' and 'to watch' are rendered differently, 'watch' is pronounced 'చూడటం' (chudatam) in Telugu, and 'to watch' is pronounced as "చూడటానికి" (chudataaniki).

In Telugu, this 'ki' is known as vibhakti. Vibhakti is a single or multiple letter that is added to a word in a sentence to emphasise the relationship with other words in the sentence. Because the English language lacks Vibhakti, certain words and prepositions are rendered as Vibhakti in Telugu.

"Chamanthi is playing in her room," for example, will be translated as "చామంతి తన గదిలో ఆడుక ుంట ుంది" (Chamanthi thana gadhi lo aadukuntundhi).

In this case, 'in' is rendered as "లో" (lo) and put after the word 'room,' resulting in gadhi (room) + lo (in).

Prepositions are the most difficult aspect of translating from one language to another. When no prepositions are used in any language, for example, Telugu, bangle, etc., they are termed prepositional phrases and are translated using vibhakti or its equivalent in their respective languages. The vocabulary utilised by the translation system should be large enough to handle them.

3.2 Structure Analysis of Telugu and English

For effective translation, a comparative examination of sentence structures in English and Telugu is required.

There are three sorts of English sentences: complicated sentences, compound sentences, and simple sentences. A compound sentence is made up of two or more sentences.

Subject + Verb + Object is the linguistic pattern for simple sentences (SVO).

Gowtham, for example, plays tennis (Gowtham + plays + tennis), The format for a basic sentence in Telugu is as follows: Subject + Object + Verb. The Telugu translation of the preceding statement is as follows:

గౌతమ్ టెనిస్ ఆడతాడు (Gowtham+Tennis+aadathaadu)

Grammatical analysis of both languages, similar to sentence analysis, should be performed to develop translation rules. The English and Telugu languages are based on separate grammars and must be correctly mapped.

Consider the following sentence: "Madhu understands both telugu and tamil."

The grammar pattern for this sentence is n + v + (d + n' + c + n"), where n is the primary noun, v is the verb, d is the determiner, n' is the noun one, n" is the noun two, and c is the conjunction.

The Telugu equivalent would be: “మధు తెల గు మరియు తమిళ్ చదువుత ంది”

The grammar pattern for this sentence is n + (n 1 + c + n2) + v.

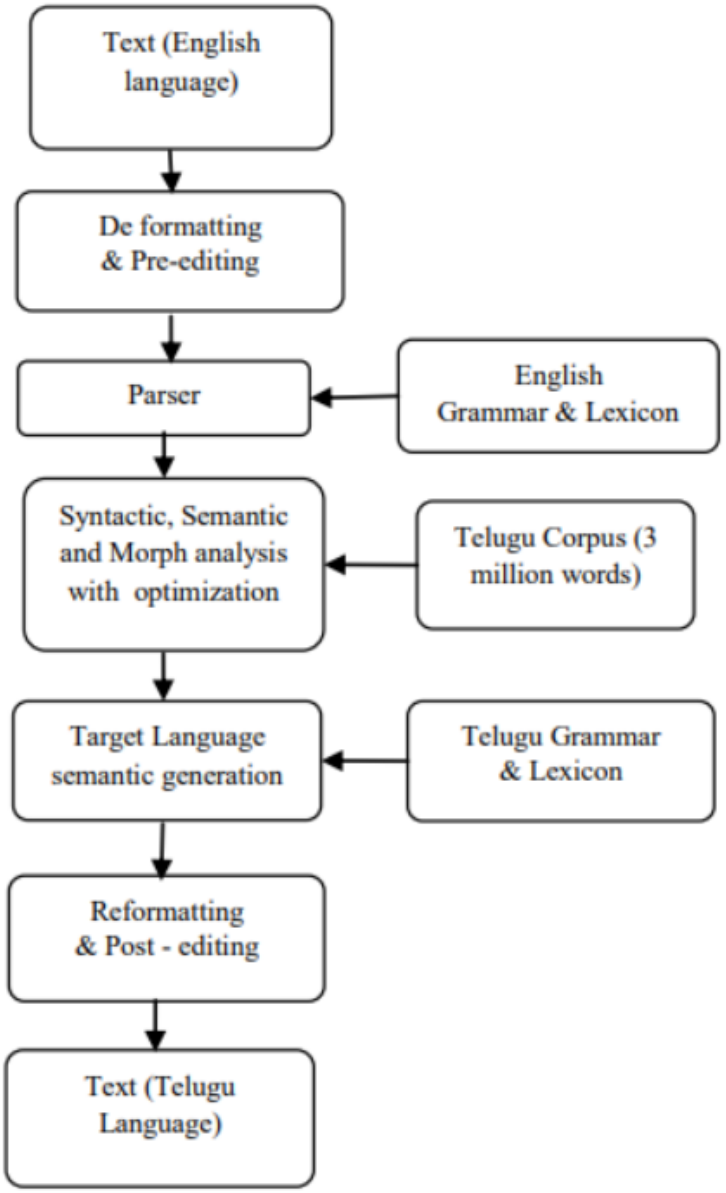


Figure 8: Open ML translation flowchart. Adapted from [12]

To map English grammar to Telugu grammar, morphological analysis is required. As a result, it is critical to compare the grammar structures of both languages in order to perform efficient language translation. Because English and Telugu are such distinct languages, prepositions and auxiliary verbs from English are not found in Telugu grammar. Similarly, Vibhakti, which is part of Telugu grammar, is not used in English grammar. As a result, auxiliary verbs will be deemed verb phrases by including the next verb and adding to it. Prepositions are considered prepositional phrases in Telugu sentences where Vibhakti is inserted as a suffix.

Chapter 4

Approach and ML Models

In this Chapter, we discuss our proposed approach for CORA Architecture and also three models which we have implemented, short answer extraction and how our baseline model differs from the final model (CORA).

4.1 Proposed Approach

Annotation Pipeline of the Cross Lingual Open Retrieval Question Answering Architecture is divided into 4 steps.

1) Question Selection

2) Question Translation

3) Answer Retrieval

4) Answer Translation

Our Proposed Architecture follows similar Architecture which was proposed by (Akari Asai et al.,2021) in XOR QA.

Question Selection: For Question Selection we randomly sample 5,000 questions without any passage answer annotations (unanswerable questions) from the training data and split them into training (4,500) and development (500) sets.

Question Translation: Our core idea is to use THUMT 3 (Zhang et al., 2017) which is a neural machine translation tool to translate languages such as German, French, Portuguese, Russian and Chinese Data into English.

Languages such as Polish, Ukrainian, Tamil and Telugu can be translated using AutoML by Google Cloud.

Answer Translation: Once the Questions are translated into Target Language, The Question dataset (Test Set) will be fed into pre-trained Haystack which is based on BM25 retriever to extract required documents from the dataset.

After document extraction, the extracted document is fed into Haystack based Cross-Lingual Bert model which will extract the short-answer (Meaningful Paragraph) from the Retrieved Document, thus retrieving answers and we can use various evaluation metrics like precision, recall to check the efficiency of this model.

Answer Translation: Once the LSTM model completes answering the questions in Target Language, answers are again translated back to English with AutoML API which was used at the time of Question Translation.

4.2 Implemented Models

Core focus of the project is to retrieve answers for questions which are not available in English language corpus using multilingual Bert by searching documents relevant to proposed questions in non-English languages and to translate back the answer to English language.

We have viewed the problem statement in a broader perspective, and we have also implemented another model where the question is given in target language, we translate the question to English, and we will try to answer the question using data in English corpus and to translate back the answer to target language.

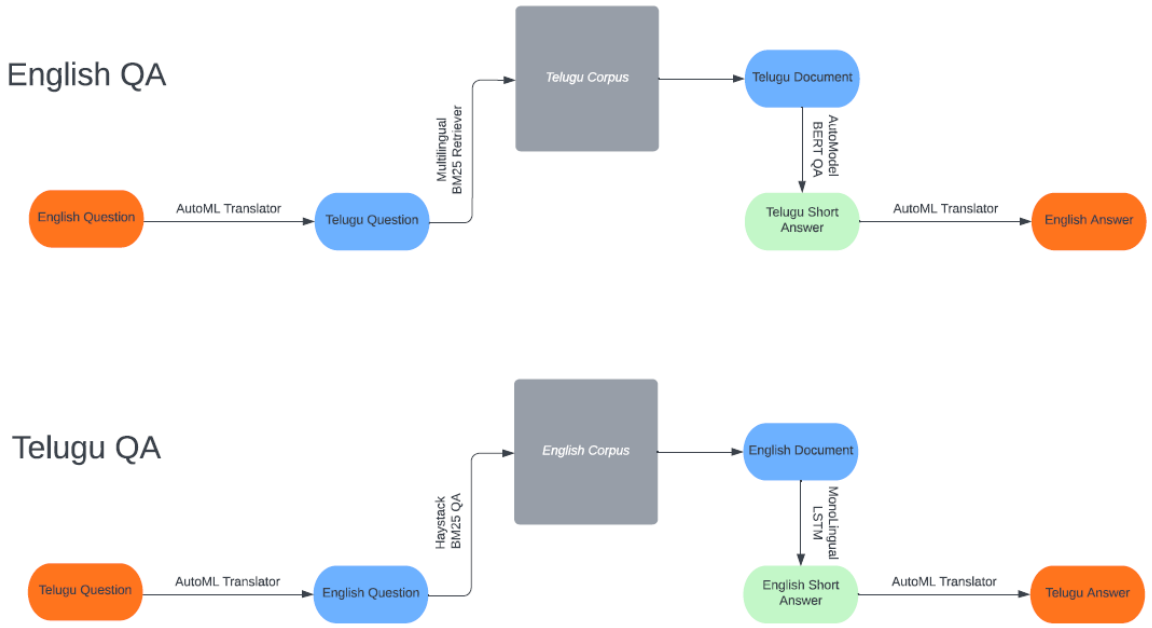


Figure 9: Model-1 and Model-2 for Cross Lingual QA

4.2.1 English QA Retrieval (Model-1)

4.2.1.1 Query Translation

Google AutoML Translator is used in the model for query translation.

Figure 10: Google AutoML Vision

Machine Translation (MT) refers to translations that are performed by a computer. Google AutoML is a well-known MT service.

Google AutoML has just begun to apply Deep Learning Strategies to improve its translations, with significant results. Even though Google AutoML isn't as good as a native-speaker translation, the outputs are usually "good enough" until they're proofread.

As a result, we used an API provided by AutoML Translator, which will be used for query translation.

4.2.1.2 Document Retrieval

In order to find a tiny relevant candidate set of documents, Retriever, a quick filter, goes through every document in the Document repository. Both sparse and dense modelling techniques are applicable to retrievers. ElasticSearch is a very effective model for retrievers. It is a legitimate search engine with sparse indexing.

BM25 is a bag-of-words retrieval algorithm that ranks a set of documents based on the query phrases that exist in each document, regardless of how close they are to each other. It refers to a group of scoring functions that have somewhat varied components and parameters. The following is one of the more notable instantiations of the function.

Given a query Q containing the keywords q1, q2,....qn, a document D's BM25 score is:

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot\left(1-b+b\cdot\frac{|D|}{\text{avgdl}}\right)}$$

Where f(qi, D) is the number of times qi appears in document D, |D| is the document D's length in words, and avgdl is the average document length in the language collection out of which documents are chosen. In the absence of an enhanced optimization, k1 and b are free parameters that are often chosen as k1[1.2,2.0] and b=0.75 is the IDF (inverse document frequency) weight of the query word qi. It is often calculated as follows:

$$\mathrm{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

where N is the total count of documents in the collection and n(qi) is the number of documents that contain qi.

IDF has numerous interpretations and modest changes on its formula. The IDF component is developed from the Binary Independence Model in the original BM25 derivation.

In simple words, BM25 is a simple Python tool that can be employed to index the data, in this case documents, depending on a search query. It operates on the TF/IDF principle, i.e., Word Frequency (TF) – Simply defined, it reflects the number of times the search term appears in our document.

IDF (Inverse Document Frequency) – This metric determines how important your search term is. Because TF values all words equally, we can't just use term frequencies to calculate the weight of a phrase in your text. We would need to weight the common terms while scaling up the rare terms to demonstrate their relevance to the topic.

In conclusion, the simplest TF-IDF penalizes document frequency and rewards term frequency. Beyond this, BM25 takes term frequency saturation and document length into account.

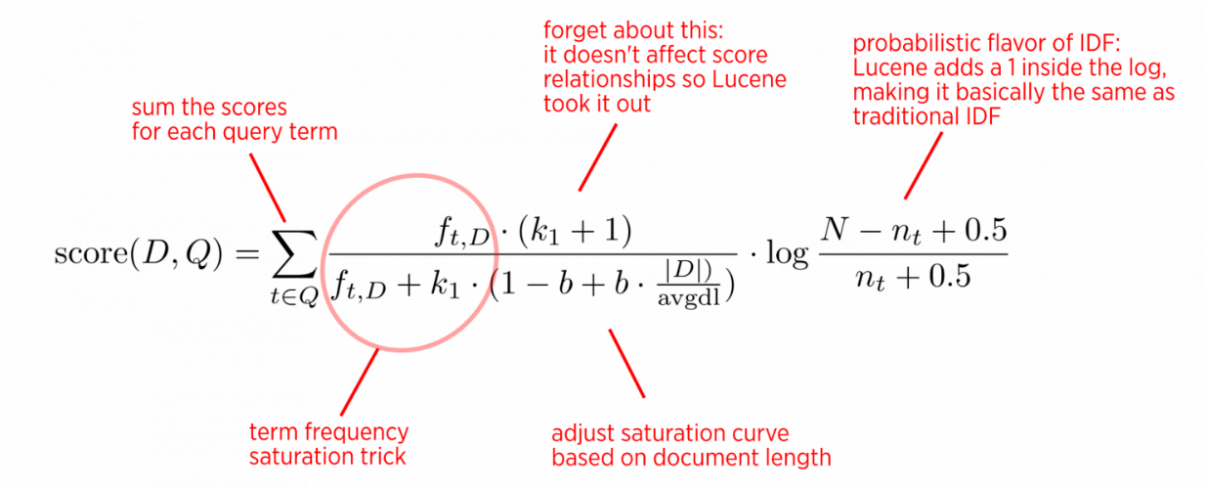


Figure 11: TF-IDF available at [13]

4.2.1.3 Short Answer Retrieval

Current model uses M-BERT for QA :

In order to learn the contextual relationships between words in a text, Transformer, an attention mechanism, is used by M-BERT. Transformer's basic design consists of two independent mechanisms: an encoder that reads the text input and a decoder that generates a job prediction. Only the encoder mechanism is required because BERT's aim is to produce a language model.

Although the Transformer architecture has an encoder-decoder structure, it does not use convolutions or recurrence to produce an output. In a nutshell, the encoder's job is to translate an input sequence into a sequence of continuous representations, which is then fed into a decoder, which is located on the left half of the Transformer architecture.

In order to create an output sequence, the decoder, located on the right half of the architecture, gets both the encoder's output and the decoder's output from the previous time step. The Transformer encoder reads the entire sequence of words at once, in contrast to directional models, which read the text input sequentially (from right to left or left to right). Although it would be more accurate to describe it as non-directional, it is therefore thought of as bidirectional. This trait enables the model to understand a word's context depending on all of its surroundings (left and right of the word).

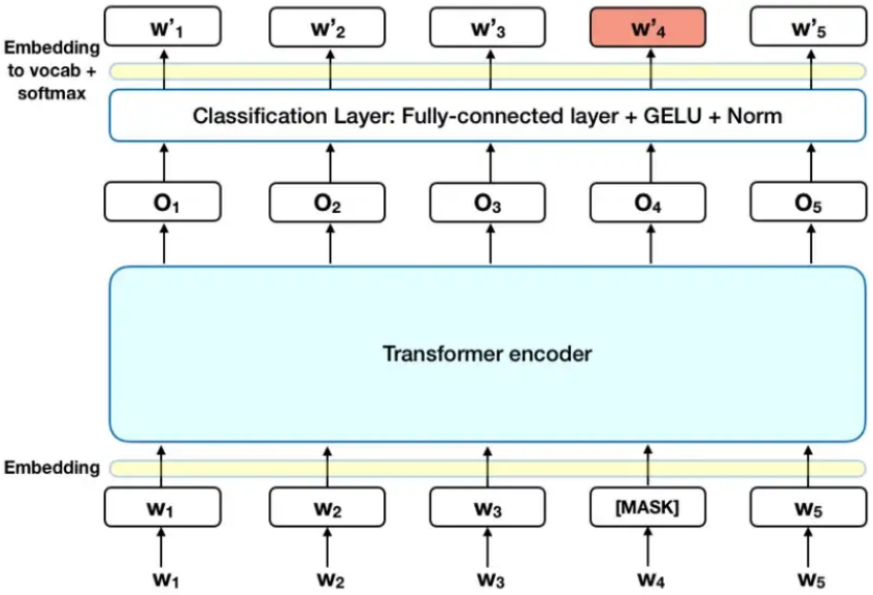


Figure 12: Architecture of BERT Model available at [17]

In our current model we are using the BERT model through haystack. Large neural networks do incredibly well both in extractive question answering and generative question answering, especially ones with transformer-based topologies (QA). These models, however, take a lot of time and money to compute. They can't be used in latency-sensitive applications because of this. HayStack addresses this issue by prefiltering the documents with quicker but less effective techniques. This makes it possible for the Neural Model to finish making inference quickly.

4.2.1.4 Answer Translation

Our current model uses AutoML translation to convert answers back to the target language. You can use supervised learning with AutoML Translate, which entails teaching a computer to identify patterns from translated sentence pairs. We can train a special model to translate domain-specific content you care about using supervised learning.

4.3.1 Telugu QA Retrieval (Model-2)

4.3.1.1 Query Translation

Google AutoML Translator is used in the model for query translation.

4.3.1.2 Document Retrieval

BM25 was used for retrieval in this model. BM25 is a ranking function that assigns a set of documents a score based only on the query phrases that each document contains, regardless of how closely related the query terms are to one another.

4.3.1.3 Short Answer Retrieval

Baseline model uses LSTM for QA:

An artificial neural network called LSTM(long short-term memory) is used for question answering. LSTM features feedback connections as opposed to typical feedforward neural networks. LSTM functions on a high level very similarly to an RNN cell. As can be seen in the image below, the LSTM is composed of three sections, each of which has a specific role.

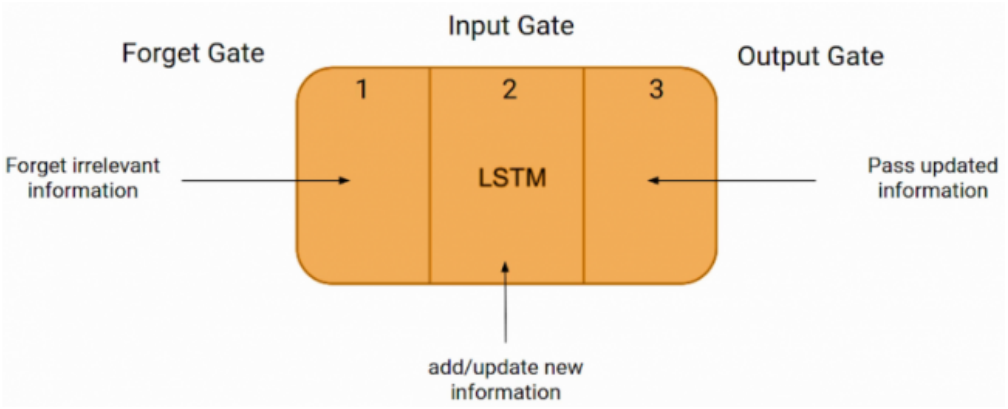


Figure 13: LSTM Architecture available at [18]

The first section determines whether the information from the preceding timestamp needs to be remembered or can be ignored. The cell attempts to learn new information from the input to this cell in the second section. The cell finally transmits the revised data from the current timestamp to the next timestamp in the third section.

Gates refer to these three LSTM cell components. The Forget gate, Input gate, and Output gate are the names of the three components, respectively.

4.3.1.4 Answer Translation

Our telugu retrieval model uses AutoML translation to convert answers back to the target language.

4.3 BERT vs BI-LSTM

Making any neural network have the sequence information in both directions—backwards (future to past) or forward—is known as bidirectional long-short term memory (bi-lstm) .

A bidirectional LSTM differs from a conventional LSTM in that our input flows in two directions. We may make input flow in one way, either backwards or forward, using the standard LSTM. To maintain both past and future information, bi-directional input can be made to flow in both ways.

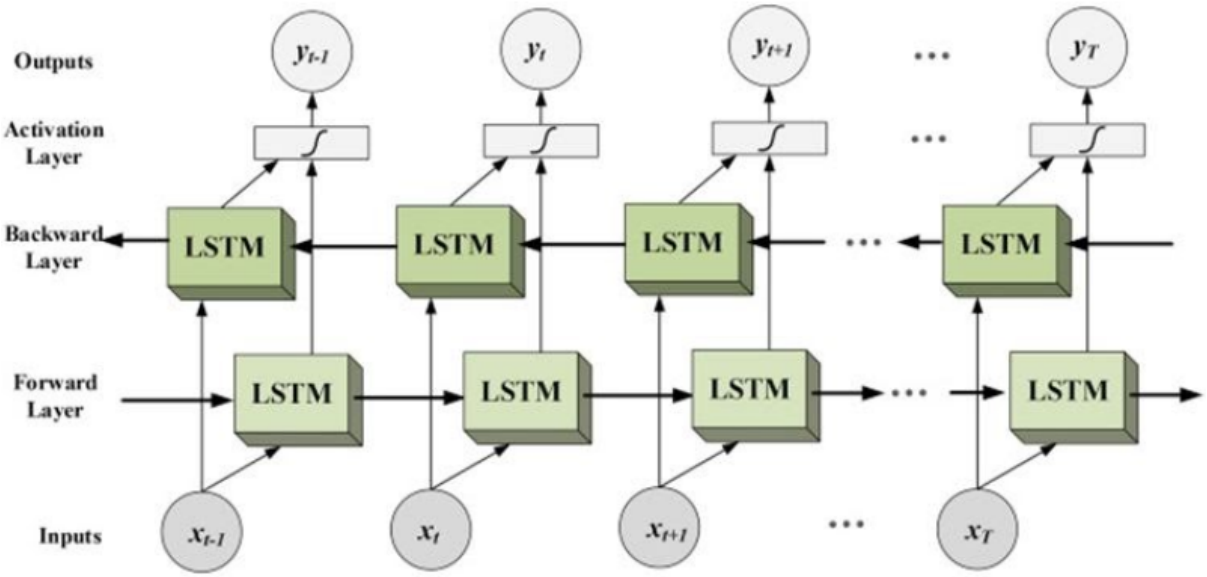


Figure 14: Bi-directional LSTM. Adapted from [19]

The flow of information from the backward and forward layers is depicted in the diagram. BI-LSTM is typically used when activities requiring sequence to sequence are required. Speech recognition, text categorization, and forecasting models can all employ this type of network.

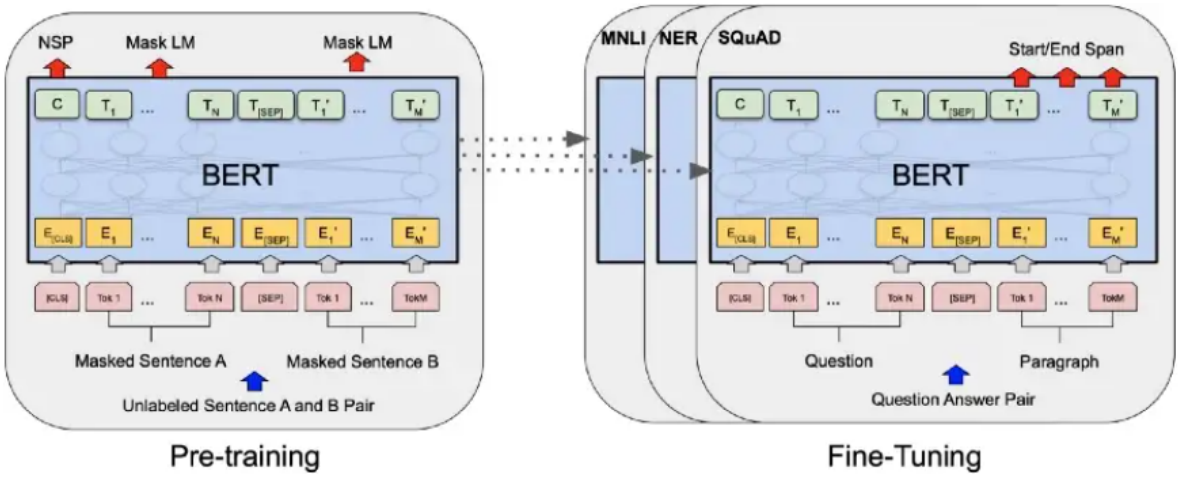


Figure 15: BERT behavior during pre-training vs Fine tuning. Adapted from [20]

Any given NLP technique aims to comprehend spoken human language in its natural setting. For BERT, this often requires finding a word out of a blank. Models must typically be trained using a sizable collection of specific, labeled training data to accomplish this. This calls for teams of linguists to laboriously label data manually.

However, just an unlabeled, plain text corpus was used to pre-train BERT. Even if it is being used in real applications, it still learns unsupervised from the unlabeled text and becomes better. Its pre-training acts as a foundational layer of "knowledge" upon which to build. From there, BERT can be adjusted to the user's preferences and the constantly expanding body of searchable content. Transfer learning is the name for this procedure.

The transformer is the component of the model that gives BERT its improved ability to comprehend linguistic ambiguity and context. Instead of processing each word individually, the transformer accomplishes this by analyzing each word in relation to every other word in the sentence. The Transformer enables the BERT model to comprehend the word’s complete context and, as a result, better understand the searcher's intent by taking a look at all the surrounding terms.

Large amounts of labeled data are needed for these word embedding models. However, because all words are in some way tied to a vector or meaning, they struggle with the context-heavy, predictive nature of question answering. To prevent the word in focus from

"seeing itself," or having a fixed meaning independent of its context, BERT employs a technique of masked language modeling. The masked word must then be determined by BERT only based on context. Instead of having a predetermined identity, words in BERT are defined by their context.

Large amounts of labeled data are needed for these word embedding models. However, because all words are in some way tied to a vector or meaning, they struggle with the context-heavy, predictive nature of question answering. To prevent the word in focus from "seeing itself," or having a fixed meaning independent of its context, BERT employs a technique of masked language modeling. The masked word must then be determined by BERT only based on context. Instead of having a predetermined identity, words in BERT are defined by their context.

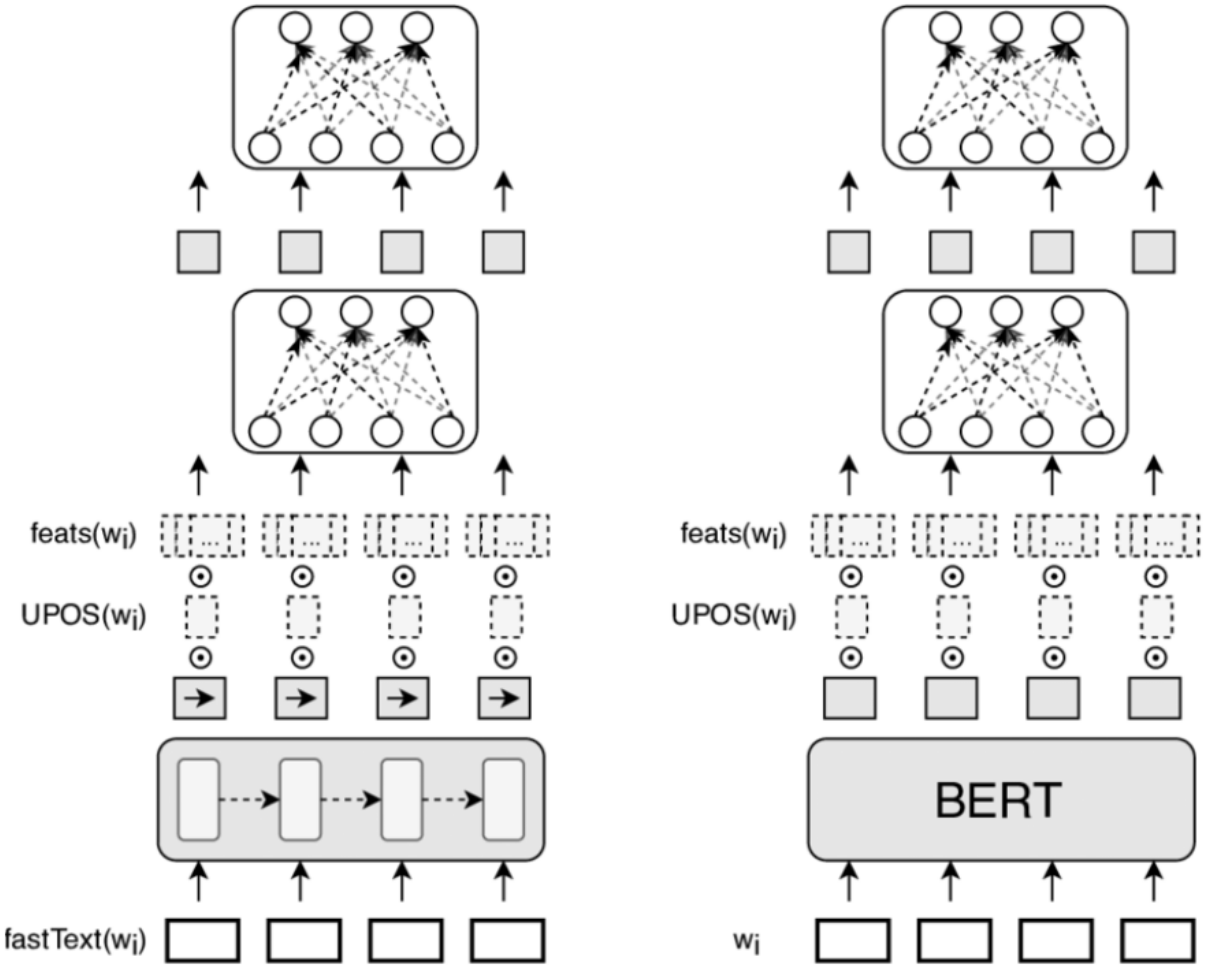


Figure 16: Comparison in architecture of LSTM vs BERT. Adapted from [21]

For a short dataset, bidirectional LSTM models can perform noticeably better than a BERT model, and they can be trained in a lot less time than their pre-trained equivalents. We come to the conclusion that a model's performance is reliant on the task and the data, so in our instance, we took these elements into account before selecting models for specific pipelines.

For our case when the baseline model was built the corpus contained small paragraphs of data to train on and then perform on the corresponding question. Here the data available to train on is very small, so by using a bidirectional LSTM we can train with smaller sets yet achieve good accuracy. For the second part, Multilingual BERT which is customized for telugu is more preferred as telugu has different language structure and the multilingual BERT model also has been trained telugu corpus so it’s more suited for telugu dataset. Here since we are using haystack we don’t need to have performance overhead of pretraining BERT and arranging for huge corpus of telugu data for the model to be trained upon.

Even in our data when the dataset is given in a lesser quantity LSTM excels at the task but BERT model has lesser accuracy. Whereas, when the dataset is given in a larger quantity the BERT model outperforms the baseline LSTM model. The inherent limitations of the models are still present but BERT is more suited here because of the amount of data we have at our disposal. Due to a customized pipeline which is set up for telugu based questions BERT is also accurate at answering individual questions to test out the how the models are working.

Our results are in line with the research carried out by other teams. For instance, Aysu Ezen-Can’s publication on “A comparison between BERT and LSTM on small corpus” shows accuracy vs the amount of dataset used for both the models. Our results are in line with their findings. Our BERT has a bit more accuracy as it is fine tuned to the Telugu dataset.

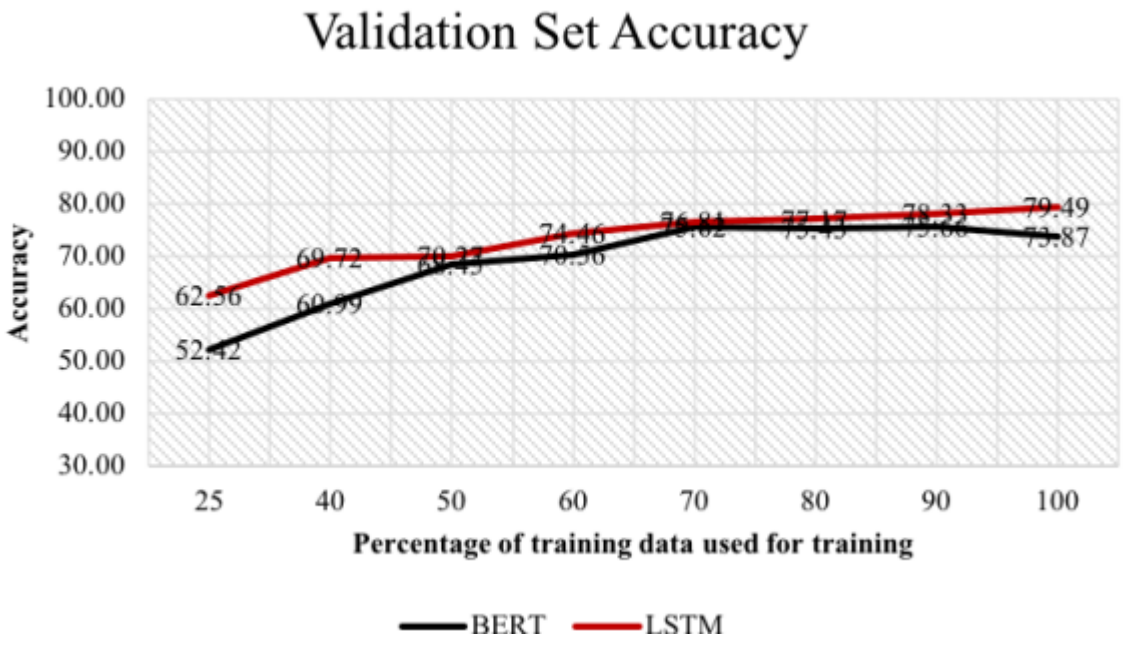


Figure 17: BERT vs LSTM accuracy for small corpus. Adapted from [23]

For general intent classification tasks the pipeline for LSTM vs BERT is as follows

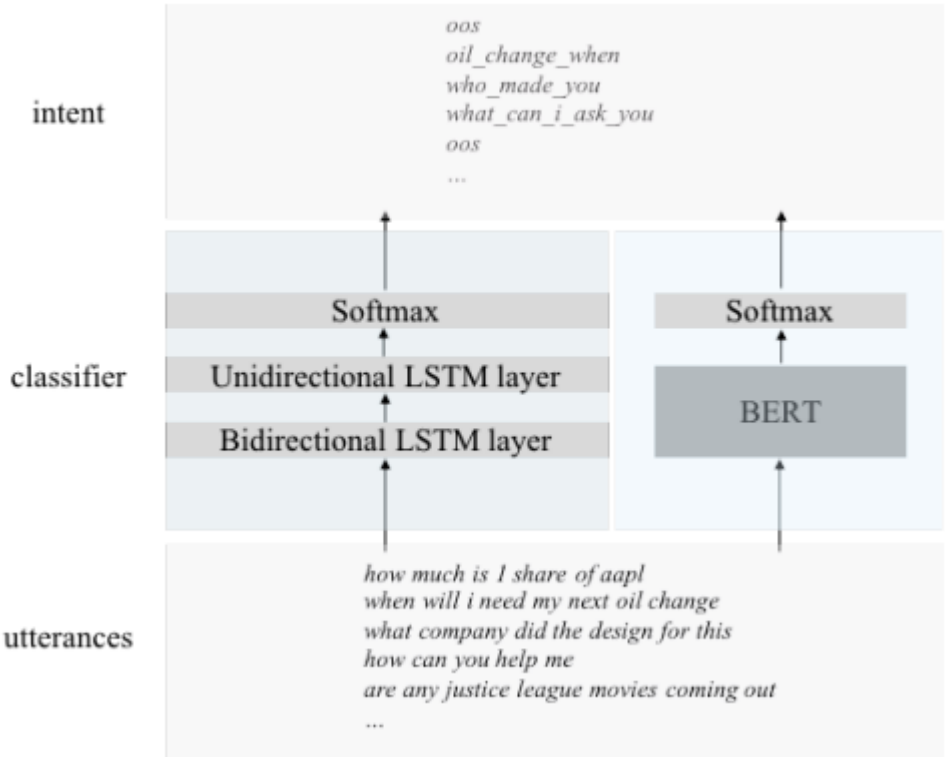


Figure 18: BERT vs LSTM pipeline for text classification. Adapted from [23]

4.4 Implemented model (CORA)

4.4.1.1 Approach

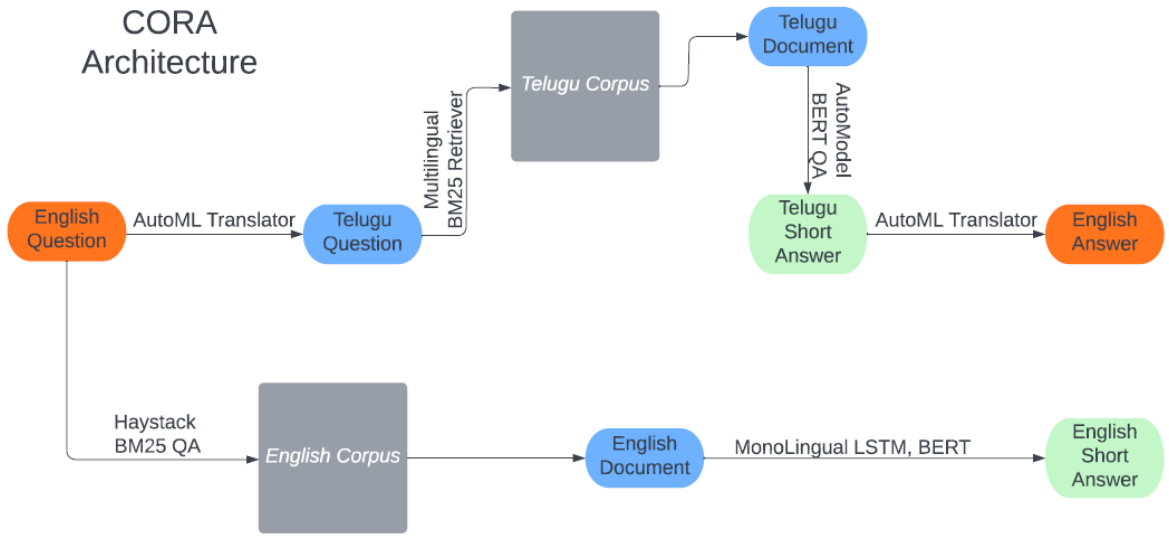


Figure 19: Architecture of the final model

4.4.1.2 Query Translation

Google AutoML Translator is used in the model for query translation in case of translating english to telugu question. But in the case of English corpus the query can be used as it is.

4.4.1.3 Document Retrieval

BM25 was used for retrieval in this model. BM25 is a ranking function that assigns a set of documents a score based only on the query phrases that each document contains, regardless of how closely related the query terms are to one another.

Multilingual BM25 is used for telugu retrieval whereas Haystack BM25 is used in case of English Retrieval.

4.4.1.4 Short Answer Retrieval

Question answering is handled by AutoModel BERT in case of telugu answer retrieval whereas it is done Monolingual LSTM in case of English.

4.4.1.5 Answer Translation

Our current model uses AutoML translation to convert answers back to the target language.

.

Chapter 5

Performance Evaluation

We compared our retriever model's performance with existing models to find the best performer. In terms of query speed and indexing speed our model outperforms the rest.

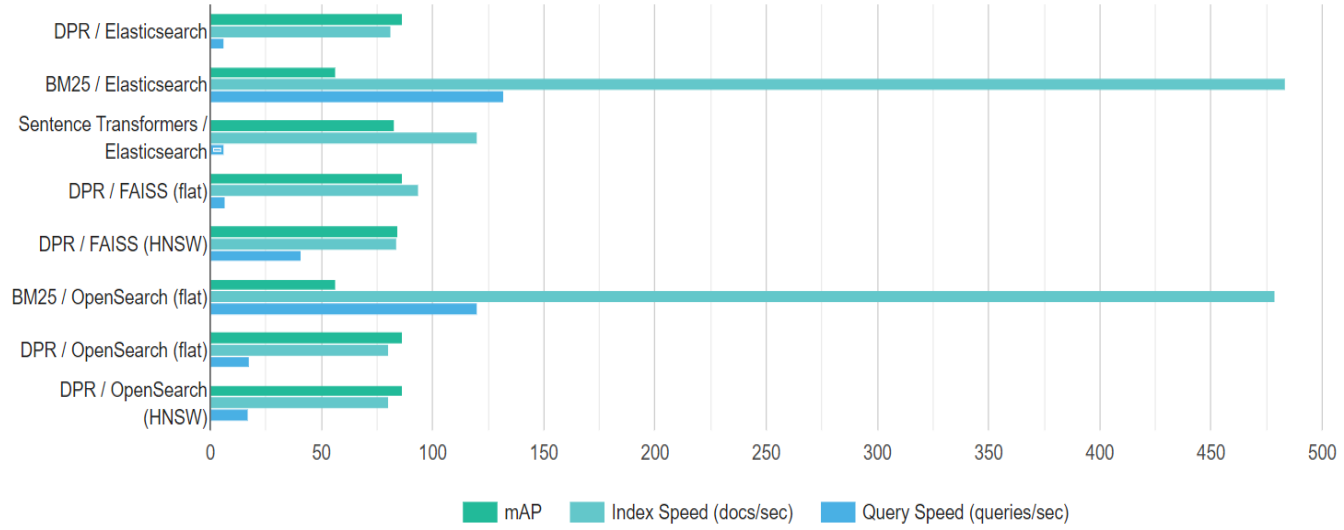


Figure 20: Performance of retrieval model. Adapted from [22]

In general DPR is outperformed by BM25 in case of retrieval in Question Answering Systems. Although the gains in mean average precision and query speed are low, index speed is very high in case of BM25 model. This speeds up the training of the model by a large amount therefore reducing the runtime.

When it comes to Initial Results achieved using the LSTM model on TensorFlow QA (Figure 25) mentions key statistics such as Validation and Training loss on Long Answer Retrieval Model.

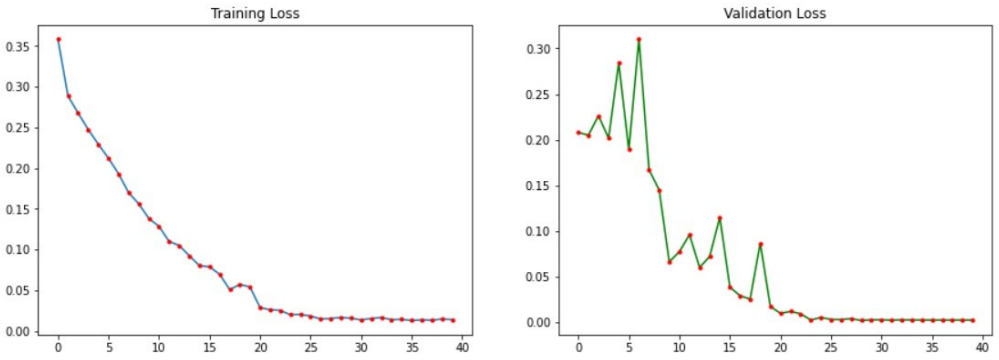


Figure 21: Statistics on Long Answer Retrieval Model of CORA

Figure 22 shows Accuracy, Precision and Recall for the 40 epochs which are run on Long Answer Retrieval Model of CORA.

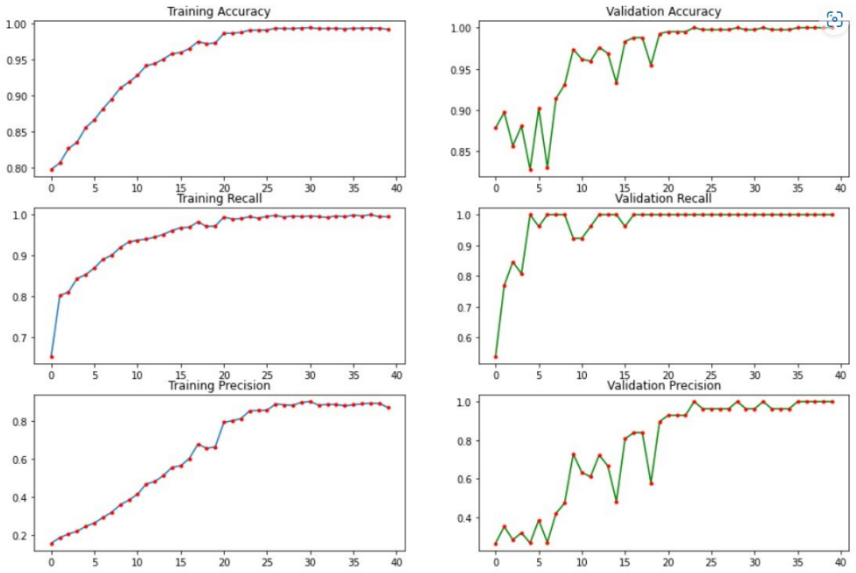


Figure 22: Statistics on Long Answer Retrieval Model of CORA

Figure 23 and Figure 24 show Evaluation metrics of Short Answer Retrieval Model for Training and Validation Sets.

```
Accuracies based on positions for training are
-----------------------------------------------
Start position accuracy: 0.8757225275039673
Start position recall: 0.8688362836837769
Start position precision: 0.8953251838684082
Start position F1 score: 0.8819
-----------------------------------------------
End position accuracy: 0.8641618490219116
End position recall: 0.8451676368713379
End position precision: 0.8789743781089783
End position F1 score: 0.8617
```

Figure 23: Evaluation metrics of Train Data Short Answer Retrieval Model of CORA

```
Start position accuracy: 0.9615384340286255
Start position recall: 0.9615384340286255
Start position precision: 1.0
Start position F1 score: 0.9804
-----------------------------------------------
End position accuracy: 0.9230769276618958
End position recall: 0.9230769276618958
End position precision: 0.9599999785423279
End position F1 score: 0.9412
```

Figure 24: Evaluation metrics of Validation Data on Short Answer Retrieval Model of CORA

Figure 24 and 25 mention key statistics such as Validation and Training loss on Short Answer Retrieval Model.

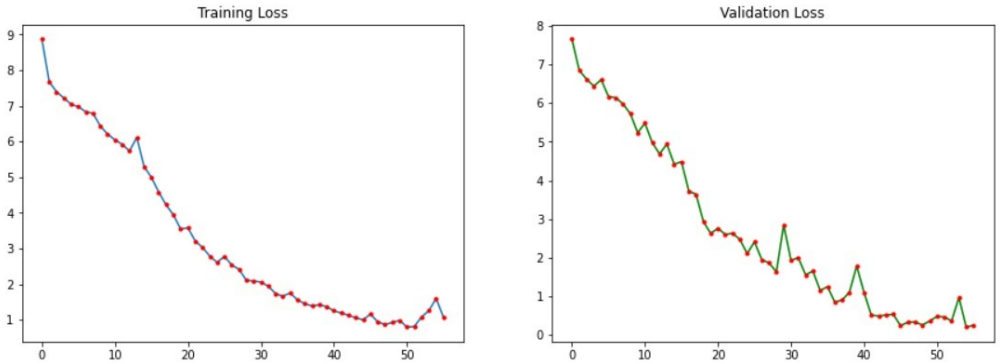


Figure 25: Evaluation metrics (Loss) of Validation Data on Short Answer Retrieval Model of CORA

Limitations

- The model is currently hosted on NGROK, this can be made live through a website and backend which has model cached instead of running it through colab
- Dataset can be more diverse with inclusion of different types of structure in questions. The Telugu dataset isn't openly available; it must be curated so that the BERT model can be trained more accurately.
- We are currently limited by the existing translation models. This affects the model training as there is no clear distinction between subject and predicate in some cases.

Conclusion and Future Work

We wish to see the following changes/improvements made to the model in the near future -

1. The first step to improve our model would be to implement a better translation model which is context based and can understand the structure of telugu language. Current openML model is weak in this aspect.
2. Dataset which is fine tuned to telugu language is necessary as questions that are in similar structure are not good enough to train the model.
3. Better UI/UX can be developed for the cross lingual system. Search results can be simultaneously shown as the question is being asked. Dataset can be dynamically built from several websites to increase accuracy.
4. Based user feedback the model can be trained automatically i.e it dynamically changes it's weights based on feedback from users

In this project, we were successful in implementing a cross lingual open retrieval system that can answer questions with good accuracy. We were able to implement a novel method which involves searching in the target language dataset as well. This required us to adapt and modify existing works and also come up with new methods and metrics. As this issue becomes more mainstream, the architecture for this model will get better in time.

Bibliography

[1] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018.

[2] Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In Proceedings of AAAI, pages 5012–5019.

[3] Mikel Artetxe and Holger Schwenk. 2018. Massively multilingual sentence embeddings for zeroshot cross-lingual transfer and beyond. ArXiv preprint arXiv:1812.10464.

[4] Jian-Yun Nie. Cross-Language Information Retrieval. Graeme Hirst, University of Toronto. Morgan & Claypool Publishers series ISBN: 9781598298642

[5] Pothula, Sujatha & Dhavachelvan, P... (2011). A Review on the Cross and Multilingual Information Retrieval. International Journal of Web & Semantic Technology. 2. 115-124.

[6] Danilo Giampiccolo, Pamela Forner, Jesús Herrera, Anselmo Peñas, Christelle Ayache, Corina Forascu, Valentin Jijkoun, Petya Osenova, Paulo Rocha, Bogdan Sacaleanu, et al. 2007. Overview of the CLEF 2007 multilingual question answering track. In CLEF.

[7] Momchil Hardalov, Todor Mihaylov, Dimitrina Zlatkova, Yoan Dinkov, Ivan Koychev, and Preslav Nakov. 2020. EXAMS: A multi-subject high school examinations dataset for cross-lingual and multilingual question answering. In EMNLP.

[8] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In ACL.

[9] Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian ˇ Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In ACL.

[10] Holger Schwenk and Matthijs Douze. 2017. Learning joint multilingual sentence representations with neural machine translation. In Proceedings of the 2nd Workshop on Representation Learning for NLP, pages 157–167, Vancouver, Canada. Association for Computational Linguistics.

[11] Holger Schwenk and Xian Li. 2018. A corpus for multilingual document classification in eight languages. Proceedings of LREC, Miyazaki, Japan.

[12] Lingam, K. Mallikharjuna et al. "English to Telugu Rule based Machine Translation System: A Hybrid Approach." International Journal of Computer Applications 101 (2014).

[13] Rudi Seitz's Article on tfidf and BM25 Available at: https://kmwllc.com/index.php/2020/03/20/understanding-tf-idf-and-bm-25/

[14] Rudi Seitz's Article on DeepPavlov https://towardsdatascience.com/bert-based-cross-lingual-question-answering-with-deeppavlov-704242c2ac6f

[15] XOR QA: Cross-lingual Open-Retrieval Question Answering (Akari Asai et al., 2021)

[16] XQA: A Cross-lingual Open-domain Question Answering Dataset (Jiahua Liu et al., 2020)

[17] Article by Rani Horev on BERT model and its architecture available on: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

[18] Article by Shipra Saxena on LSTM model and its architecture available on: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

[19] Article by Rayhan Tito Kharishma on Bi-LSTM model available on: https://medium.com/@rayhantithokharisma/sentiment-classification-using-bidirectional-lstm-model-with-twitter-us-airline-sentiment-dataset-85b601200f66

[20] Milios, Aristides & Pralat, Pawel & Soares, Amilcar & Theberge, F.. (2021). Survey of Generative Methods for Social Media Analysis.

[21] Klemen, Matej & Krsnik, Luka & Robnik-Sikonja, Marko. (2020). Enhancing deep neural networks with morphological information.

[22] Testing results of haystack models and their performances compared Available on https://haystack.deepset.ai/benchmarks

[23] Ezen-Can, Aysu. "A Comparison of LSTM and BERT for Small Corpus." *ArXiv* abs/2009.05451 (2020): n. pag.