Expert Answers in a Flash: Improving Domain-Specific QA

InterIIT Tech Meet 11.0

Team ID: 43

Midterm Report

January 2023

Introduction

There is a widespread effort to reduce the volume of assistance requests that are handled by support agents by letting the system handle them initially and responding automatically with some suggestions. In our situation, we want to create a system that, given a user query, returns a Knowledge Base article (KB article) in order to reduce human effort. In this report, we present a summary of the methodologies and approaches used to address the problem statement, as well as the models and outcomes attained for both Tasks 1 and 2. The justifications for our model selections are also included in the methodology section. We provide references to relevant research works and sources throughout the process.

Literature Review

We went through various publications in this research area and shortlisted a few that we thought were relevant to our task. The following section contains brief descriptions of the publications.

Transformer Architecture

Described in the research work Attention is All You Need [1] the transformer architecture comprises of encoder-decoder stacks. Transformer-based models like BERT [2] and its variants are extensively used for extractive Question-Answering.

Key Value Memory Networks

Most extractive QA models are based on well-structured knowledge bases, but they have their own drawbacks. Knowledge bases have limitations such as their inevitable incompleteness and fixed schema. Complete documents can potentially answer more questions, but retrieving answers directly from complete documents is much harder. KeyVM-network [3] is a method based on memory networks that extracts facts from a document and stores them in a key-value database. As a result of the unstructured paragraphs and seeming viability of storing data, this was pertinent to the work at hand. We discarded KeyVMNN in favour of transformer-based models because KeyVMNN was not readily extensible to tasks with possibly unanswerable questions.

Seq2Seq AI Chatbot with Attention Mechanism

Seq2Seq AI Chatbot with Attention Mechanism [4] is developed using an encoder-decoder attention mechanism architecture. This encoder-decoder uses RNN with LSTM cells. The attention mechanism introduced in this research work, Neural Machine Translation by Jointly Learning to Align and Translate, allows the decoder to look at the input sequence while decoding selectively. This takes the pressure off the encoder to encode every useful information present in the input. Although, the paper mentions that the approach involves a long training process that demands high processing power and configured machine.

SQuAD 2.0 Based on ALBERT and Ensemble

A comparison between a Pre-trained Contextual Embedding based Model (ALBERT), a Non-Pre-trained Contextual Embedding based Model (BiDAF with character-level embedding), and an ensemble of a classifier with the aforementioned SQuAD 2.0 solver is conducted in the report - [5]. Results show that using the ensemble-based technique increased accuracy for both the F1 and EM scores. The ensemble-based methods' best accuracy was provided by ALBERT-large (EM: 79.74 and F1: 82.53). Additionally, this research highlights how often hyperparameters (such as learning rate, etc.) affect the models' output correctness and calls for alternatives to iterative search.

Ensemble-Learning for Sustainable NLP

The major objective of this research [6] is to develop a QA model that uses less resource-intensive operations while maintaining accuracy levels that are on par with conventional BERT-based models. Similar to our objective, the authors focus especially on how fast the inference can be performed using lesser energy. They conduct a thorough statistical analysis of the models' F1 accuracies and suggest an ensemble-based strategy that combines a small model, such as BiDAF, and a large model, such as a pre-trained BERT model, with parameters that are roughly 50 times greater than those of the small model.

The problem of selecting the right model for a question has been evaluated using 5 distinct categorization techniques. A Random Forest Classifier was used to find rule-based classifiers with feature engineering. Although the accuracy did not increase much, the number of calls to the BERT model significantly decreased by 58 percent. It also revealed several intriguing characteristics, such as the longest-common subsequence (LCS) and the proportion of words that overlap other words, such as question words. Better accuracy and a 30% reduction in BERT calls were produced by neural classifiers (such as 1D CNNs and ANNs), but they were also quite susceptible to overfitting.

Retrospective Reader for Machine Reading Comprehension

The primary motive of the paper [7] is to create a QA model that uses a retrospective model that integrates two stages of reading and verification strategies parallelly to identify unanswerable questions : sketchy reading and intensive reading. Here, their main focus was the decoder side of the MRC models and span-based MRC tasks. It encodes the context using PrLMs and the query using pre-trained language models like BERT, ELECTRA and ALBERTA and then passes them into the two independent modules. We also investigate two alternative question-answer matching mechanisms as an extra layer:
1) Transformer-style multi-head cross attention (CA)
2) Traditional matching attention (MA)
This model gave quite promising results and F1 score but at the cost of a very high inference time as it is working with DPR for passage retrieval.

Hybrid BiLSTM-Siamese Network for FAQ Assistance

This approach [8] combines a BiLSTM-based classifier with a BiLSTM-based Siamese network. All the questions for which the classifier makes an error during training are used to generate a set of misclassified question-paragraph pairs. Along with correct pairs, the misclassified question-paragraph pairs are used to train the Siamese network to drive apart the latent representations of the misclassified pairs. A Siamese network can model the underlying semantic similarity of a pair of input sentences. This research discusses better results of the hybrid model than just a classifier network or a Siamese network. A new algorithm, HSCM-IT, which is a hybrid of BiLSTM based Siamese, Classification models and a new loss function SQRT-KLD usable within the softmax layer of a neural network, is introduced in this research work. The paper points out that the Siamese model became effective only after we choose pairs with high Jaccard similarity, not if such pairs are chosen at random. This ensures that our training time is very less as we are sampling on a small subset of the training set at a given time.

Methodology

We hope to solve the problem by dividing it into smaller pieces and solving them seperately. The method is as illustrated in the figure 1. Our initial goal is to make a solid representation of the textual input that machine learning models can comprehend by using embeddings. After that, a method for identifying relevant text passages that could contain the response to the current query is provided. We finally determine the answer's position from the text passages we had previously picked.

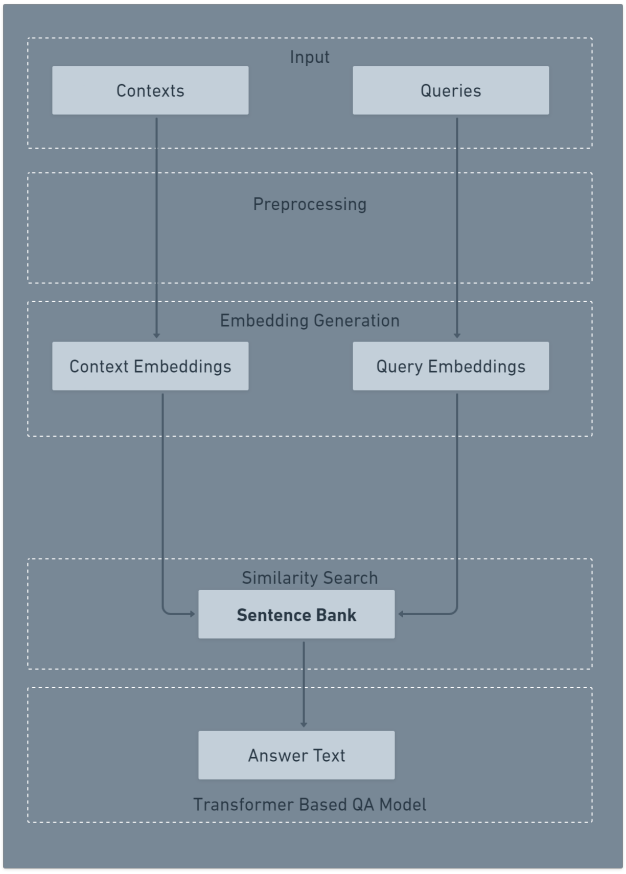

Figure 1 – General Pipeline for Question Answering

Task 1. Passage Retrieval

Method 1: Applying Nearest Neighbour Search on Sentence Embeddings

Creating Sentence Embeddings

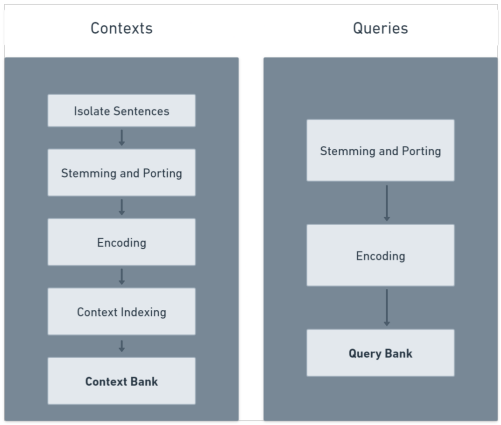

Figure 2 – Preparation of Knowledge Base

Creating embeddings is the foremost step for passing textual input to a computer. A sentence embedding assigns a vector to each sentence. Similar sentences are *nearby* in the higher dimension vector space. All transformers employ an encoding step where sentences are converted to sentence embeddings.

We explored different embedding tools like **InferSent** and **BERT** (to obtain embeddings, we tokenized sentences using BERT's tokenizer and then considered the 768-dimensional vector associated with the [CLS] token). There was a trade-off between dimensions (and hence the accuracy) and the query time.

DiffCSE is an unsupervised contrastive learning framework for learning sentence embeddings. It tries to learn sentence embeddings from a sentence, based on differences with an edited sentence, which is obtained by stochastic masking followed by sampling a masked language model. The research [9] claims that DiffCSE beats out **SimCSE** by 2.3 absolute points on semantic textual similarity tasks. However, during our experimentation, we found that the embeddings learned by DiffCSE on a small set of query sentences were inferior for sentence similarity search as compared to SimCSE, and captured a more simpler understanding of similarity which was not as useful for our task. Hence, we discarded this approach.

We are currently also experimenting with **Google's universal sentence encoder model** which specifically targets transfer learning tasks like text classification, semantic similarity, and clustering. It converts a variable length English text into a 512-dimensional vector. Instead of averaging word embeddings, it uses a unique approach which helps derive true semantic meaning of the sentence. Two variants of the TensorFlow model allow for trade-offs between accuracy and computing resources: Transformer and Deep Averaging Network (DAN). The transformer technique performs better at the cost of computation time as the network is trained on multiple tasks, while the DAN based approach has lesser training time and is almost linear in terms of the length of the input sentence. We found the Deep Averaging Network more useful in our use case and it works well with FAISS.

Finding the Nearest Neighbours

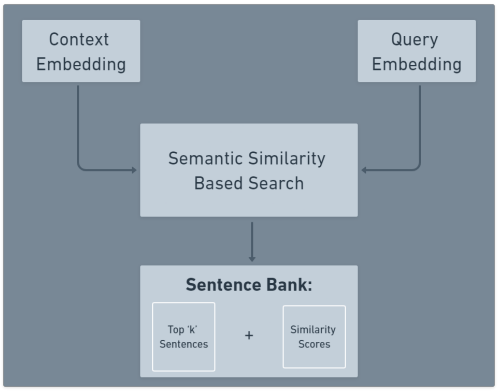

Figure 3 – Shortlisting Top *k* Sentences

Once embeddings are created, the next step is to store them in indices to facilitate fast approximate nearest-neighbour lookups. For approximate nearest neighbours the most prominent options were FAISS, ANNOY and ScANN. ScANN attempts to solve the bottleneck of Maximum Inner Product Search by using an anisotropic quantization technique illustrated here [10]. However, it performed quite poorly as compared to FAISS in our testing, even after tuning the number-of-leaves parameter. Both **FAISS** [11] and **ANNOY** had their own merits: FAISS offered constant index creation times while lookups in ANNOY were approximately 40 times faster. However, we settled on FAISS

because ANNOY's performance is known to suffer with high dimensional vectors like the sentence embeddings used by us, and the fast lookup times do not justify the drop in accuracy.

Method 2: ML Models for Passage Retrieval

RocketQA

This research work proposes a strategy to use a single training process for both passage retrieval and re-ranking. Typically, the retriever uses a list of possibilities for learning(hence called a listwise approach), while the re-ranker is learned in a pointwise manner. Instead, RocketQA uses a dynamic listwise distillation approach, where the relevance scores are computed according to a list of positive and negative passages, this lets the retriever and re-ranker adaptively improve with the information provided by the other. A hybrid data augmentation strategy is proposed to aid the same, accumulating a more diverse candidate passage for training use. Using a base ERNIE PrLM, they report an MRR@10 result of 38.8 on Passage Retrieval and 41.9 on Passage Re-ranking. However, on running this model, we obtained suboptimal results.

Task 2. Question Answering

With methods for selecting context relevant to the given query in place, for task 2, we explored pretrained QA models based on variants of BERT and tools like DensePhrases and ReAtt.

Using Pre-trained QA Models

RoBERTa

Robustly Optimized BERT Pre-training Approach, also known as RoBERTa, was trained on 160 GB of data, including the BookCorpus, Wikipedia, and some other data.
Since next sentence prediction is not used in the RoBERTa model, which is just a BERT familiarised with dynamic masking, the NSP is not removed as it is when a BERT is pretrained using masked language modelling and next sentence prediction. Due to the model's many parameters, this model had an F1 score of 0.914 on the train data but an average inference time of 700 ms.

BioBERT

The Bio-BERT model tests a domain-specific BERT-based model to gauge its performance on the SQuADv2 dataset. The model is resource intensive, so we had to train it on 1 epoch on GPU, and its results were not as promising as we expected, with an F1 score of 44.37 after fine-tuning the development set. The training time with a GPU was too high, and the average inference time per query using CPU also comes at 400 ms, which is too high per our requirements.

DistilBERT

DistilBERT was created using knowledge distillation method and is able to retain most of BERT's language understanding abilities while being smaller and faster. It was evaluated using SQuAD metrics, which showed that it was able to give correct answers to queries but sometimes included unnecessary information. It took about 3.5 hours to train for 3 epochs on a Google Colab GPU and was evaluated using SQuAD metrics. It gave correct answers to queries but often included unnecessary information, which reduced its accuracy. It also had a good accuracy for predicting null answers. Inference time on a CPU was around 300 ms, making it less useful for the intended purpose.

XLNET

XLNet solves the typical problems of BERT-like models - (1) corruption of input data by masking and (2) neglecting dependency between masked positions. XLNet uses Permutation Language Modelling instead of masking to solve these problems while retaining the generalized (bi-directional) property. We used XLNet because it outperformed BERT of various tasks, such as reading comprehension on SQUAD by 7 points. However, XLNet is RAM-intensive and RAM crashes under the given training constraint.

Retro Reader

Retrospective Reader for MRC is a transformer-based architecture with a special focus on identifying unanswerable questions. It encodes the context and the query using pre-trained language models like BERT and ELECTRA and then passes them into two independent modules. We tried training the model on

only 100 random queries and although it gave promising results with an F1 score of about 97%, it was very slow, with an average inference time of around 7-8 sec which doesn't meet our requirement.

Other Approaches

Retrieval As Attention (ReAtt)

Typically readers and retrievers for open-domain question-answering tasks are modelled separately. This research work, however, proposes leaving that in favour of a single transformer that performs ReAtt and end-to-end training solely based on supervision from the end QA task. Based on the T5 model, it uses self-attention between queries and documents as retrieval scores. It achieved impressive performance, beating many competitors with separately trained retrievers and readers on the Natural Questions dataset. However, it proved to be RAM intensive in our experimentation, leading to RAM-crash.

DensePhrases

Phrase retrieval models usually use sparse document representations, which are less effective than retriever-reader models, and hence uncommon in comprehension tasks. DensePhrases attempts to improve this by instead learning dense phrase representations from the task with negative sampling methods. It also introduces query-side fine-tuning for transfer learning. While it is claimed to be competitive with retriever-reader models on many datasets, the learned representations are often too large for resource-constrained tasks, leading to the use of other techniques.

Experimentation

For paragraph retrieval, we run our pipeline on all answerable questions from training data (i.e. 50126 queries). For question-answering, we created a validation dataset for answer prediction by sampling 5000 random queries from the training dataset provided. We then transformed our pipeline to work with the sample eval notebook supplied to us. Here is the code.

Task 1 : Paragraph Retrieval From Context Bank					
S.No.	**Embeddings Model**	**Embeddings Dimension**	**Relevant Paragraph in Top 10 Results(%)**	**Inference Time**	**Mean Rank**
1.	Univ. Sentence Encoder for QA	512	94.8%	12.41 ms	1.55
2.	Univ. Sentence Encoder Large	512	93.7%	12.66 ms	1.6
3.	SimCSE (Sentence Encoder)	768	85.8%	9.54 ms	1.8
4.	RocketQA (Context Encoder)	768	80%	14.32 ms	2.36
5.	DPR Reader (Context Encoder)	768	76%	15.34 ms	2.62

Task 2 : Answer Text Retrieval From Filtered Context					
S.No.	**QA Model**	**Model Size**	**Average Inference Time**	**F1 Score**	**Final F1 Score***
1.	Albert-Base**	46.7 MB	667 ms	0.0289	0.0085
2.	Albert-xLarge	235 MB	7530 ms	0.8005	0.0155
3.	Distilbert-Base	261 MB	321 ms	0.5958	0.3703
4.	Electra-Base	436 MB	610 ms	0.6339	0.2078
5.	Roberta-Base	497 MB	621 ms	0.5756	0.1853
6.	TinyRoberta	326 MB	315 ms	0.5867	0.3726
7.	Tinybert	17.6 MB	9 ms	0.2484	0.2484

* *Final F1 Score* $=$ *F1 Score* $\times\ 200 \div$ (*Average Inference Time*)

** *We are presently facing issues in incorporating ALBERT in our pipeline*

Future Work

We plan to create a basic pipeline for retrieving the passages and doing question-answering. There are many hyperparameters associated with it (e.g. k, the number of nearest neighbour to find) which need tuning. We are currently working on the basic blocks of this pipeline and trying to refine each of these blocks further to get the best possible results.

Context for Question-Answering

Currently, we are finding the top-k nearest neighbour (sentences) of the query and directly using that as a context in QA models. It can be better to use a sentence before and after for each of the k nearest neighbours found. This will expand the context and only improve the chance of capturing the answer. This comes with a limitation. As the size of the context increases, so does the inference time. Therefore, there is a tradeoff between the context size to use and the inference time obtained. We are currently experimenting with different contexts to use for the best performance.

Which Context To Use?

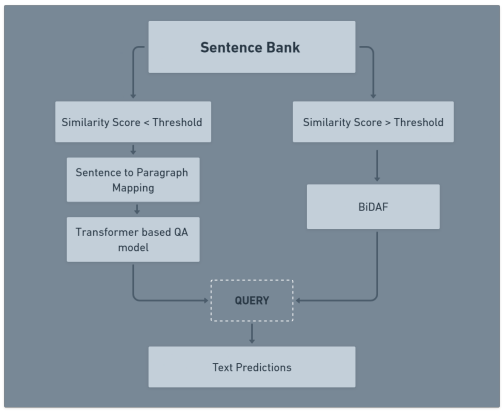

Figure 4 – Ensemble Learning Approach

We conduct a hard thresholding-based classification using the sentences that made the shortlist and their accompanying similarity scores to choose the best next step. The processed sentences are fed to the BiDAF model if the similarity score is greater than a threshold that was determined empirically because we are confident in the accuracy of the processed sentences. Although we are unclear about the sentence that contains the solution if the score is below the threshold, we nevertheless read the full paragraph. The paragraphs are processed using a QA model based on transformers. If no solution can be identified, we don't return any text.

Pre-Trained QA model

There are a lot of pre-trained QA models available that are fine-tuned on Squad 2.0 dataset. Some of these models give very high F1 Scores but are very bulky, giving very high inference time. We are looking for a model that still has the descent performance of Squad 2.0 and infers within 200 ms for any query. If we do not get such a model, we may choose to go with small models (like BiDAF) and try to fine-tune them further to improve performance.

Fine -Tuning Existing QAmodels To Improve Performance On Specific Domains

The question type and answer type are influenced by the style of the text from which answers are extracted. Training a transformer language model on the SQuAD dataset enables the model to provide short answers to general knowledge questions. These models perform well on similar question-answer types in the text similar in vocabulary, grammar, and style to Wikipedia or the web. However, the accuracy of a SQuAD-trained QA model decreases when either the question-answer type or the text domain changes. To tackle this situation, we plan to fine-tune the pre-trained QA models (already fine-tuned on Squad 2.0) on question-answers from specific themes if present, otherwise, through question-answer generation from the given paragraphs of that theme.

Appendix

- LSTM - Long Short-Term Memory
- RNN - Recurrent Neural Network
- KeyVMN - Key-Value Memory Network
- BERT - Bidirectional Encoder Representations from Transformers
- ReAtt - Retrieval as Attention
- QA - Question Answering
- ALBERT - A Light Bidirectional Encoder Representations from Transformers
- BiDAF - Bi-Directional Attention Flow
- LCS - Longest Common Subsequence
- CNN - Convolutional Neural Network
- ANN - Artificial Neural Network
- MRC - Machine Reading Comprehension
- DPR - Dense Passage Retrieval
- HSCM-IT - Hybrid of Siamese and Classification Models - Iterative Training
- Sqrt-KLD - Squareroot-Kullback-Leibler Divergence
- FAISS - Facebook AI Similarity Search
- ANNOY - Approximate Nearest Neighbors Oh Yeah
- DiffCSE - Difference-based Contrastive Learning for Sentence Embeddings
- SimCSE - Simple Contrastive Learning of Sentence Embeddings
- ERNIE PLM - Enhanced Language RepresentatioN with Informative Entities Pretrained Language Models
- PrLM - Pre-trained Language Models
- MRR - Mean Reciprocal Rank
- ScaNN - Scalable Nearest Neighbours
- GPU - Graphics Processing Unit
- CPU - Central Processing Unit
- ELECTRA - Efficiently Learning an Encoder that Classifies Token Replacements Accurately
- CLS token - Special Classification Token
- EM Metric - Exact Match Metric
- ML - Machine Learning
- TP - Number of True Positives
- TN - Number of True Negatives
- FP - Number of False Positives
- FN - Number of False Negatives
- AIT - Average Inference Time
- NSP - Next Sentence Prediction
- AI - Artificial Intelligence
- DAN - Deep Attention Neural Network

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[3] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents, 2016.

[4] Abonia Sojasingarayar. Seq2seq ai chatbot with attention mechanism, 2020.

[5] Yanpei Tian. Squad 2.0 based on albert and ensemble. 2020.

[6] Elena S. F. Berman and Surya N Hari. Ensemble-learning for sustainable nlp. 2020.

[7] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. Retrospective reader for machine reading comprehension, 2020.

[8] Prerna Khurana, Puneet Agarwal, Gautam Shroff, Lovekesh Vig, and Ashwin Srinivasan. Hybrid bilstm-siamese network for faq assistance. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 537–545, New York, NY, USA, 2017. Association for Computing Machinery.

[9] Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. Diffcse: Difference-based contrastive learning for sentence embeddings, 2022.

[10] Ruiqi Guo, Quan Geng, David Simcha, Felix Chern, Sanjiv Kumar, and Xiang Wu. New loss functions for fast maximum inner product search. *CoRR*, abs/1908.10396, 2019.

[11] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.