PROJECT REPORT

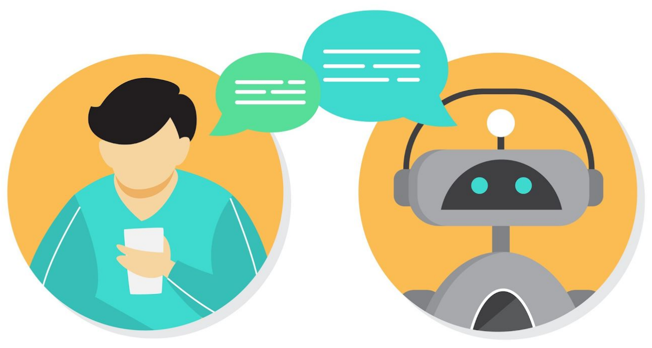
Team 19
Sai Dinesh Bijjam
Subrahamanyam Verma

Code is Available here: https://tinyurl.com/nla-rankqa-illuminate

Index

In this report we are going to go through the following:

Introduction



Question Answering system is an information retrieval system in which a direct answer is expected in response to a submitted query, rather than a set of references that may contain the answers. It is a man machine communication device. The basic idea of QA systems in Natural Language Processing is to provide correct answers to the questions for the learners. These QA systems are classified into the following types:

- Factoid QA systems
- Web based QA systems
- Restricted Domain QA systems
- Rule based QA systems.

There will be at least three core components in a Question Answering System, they are:

- **Question classification:** This plays an essential role in QA systems by classifying the submitted question according to its type. In order to correctly answer a question, it is required to understand what type of information the question asks for, because knowing the type of a question can provide constraints on what constitutes the answer, which helps other modules to correctly locate and verify an answer. The question type classification component is therefore a useful, if not essential, component in a QA system as it provides significant guidance about the nature of the required answer.

- **Information retrieval:** This is very important for question answering, because if no correct answers are present in a document, no further processing could be carried out to find an answer. Information domains, such as the web, have enormous information content. Therefore, the goal of the information retrieval system is to retrieve accurate results in response to a query submitted by the user, and to rank these results according to their relevancy. One thing to be considered is that it is not desirable in QA systems to rely on IR systems which use the cosine vector space model for measuring similarity between documents and queries. This is mainly because a QA system usually wants documents to be retrieved only when all keywords are present in the document.

- **Answer extraction:** Answer extraction aims to retrieve the answer for a question asked by the user. Researchers have presented miscellaneous heuristic measures to extract the correct answer from the answer candidates. Extraction can be based on measures of distance between keywords, numbers of keywords matched and other similar heuristic metrics. Commonly, if no match is found, QA systems would fallback to delivering the best ranked paragraph.

Literature Survey

- **DrQA**
    - **Intuition and Problem**
        - **Intuition:** We need to find similarity between a question and a paragraph of the data and return the paragraph with highest similarity.
        - **Problem:** The data is huge!! Good similarity metrics computations are intensive. Take the following question as example,
          "Who wrote the film Gigli?"
          How do we compare this question with millions of pages with lots of paragraphs in wikipedia with intensive similarity computation? (Technically not feasible)
    - **Conventional Method**
        - **Information Retrieval:** Narrow down the set to top-n relevant articles
        - **Machine Comprehension:** There will be k>>n candidate answers from top-n articles, the aim of this module is to rank these k candidate answers and select one.
    - **Information Retrieval**
        - Calculate TF-IDF score for all the words in a question. We get a list of TF-IDF scores for a question (q)
        - For each document:
            - Calculate TF-IDF score for all the words in the document
        - This will give a list of TF-IDF scores for a document(d_i)
        - Similarity(f) = q (X) d_i ; Where (X) = Dot product
        - Return the top-n documents with max f
    - **Machine Comprehension: Paragraph Encoding**
        - Let us say we have n-paragraphs from the previous module, assume that the question has l words and each paragraph has a maximum of m words.
        - We construct the following features for i^{th} word in a paragraph p:
            - Word embeddings of word p_i (Glove)
            - Exact match: 3 booleans, if p_i is in q
                - Originally
                - Lowercase
                - Lemma form
            - Token features: POS, NER, TF
        - **Machine Comprehension: Question Encoding**
            - We train an RNN on the word embeddings of question and compute the feature vector from hidden outputs from the RNN layers
            - $\mathbf{q} = \sum_j b_j \mathbf{q}_j$

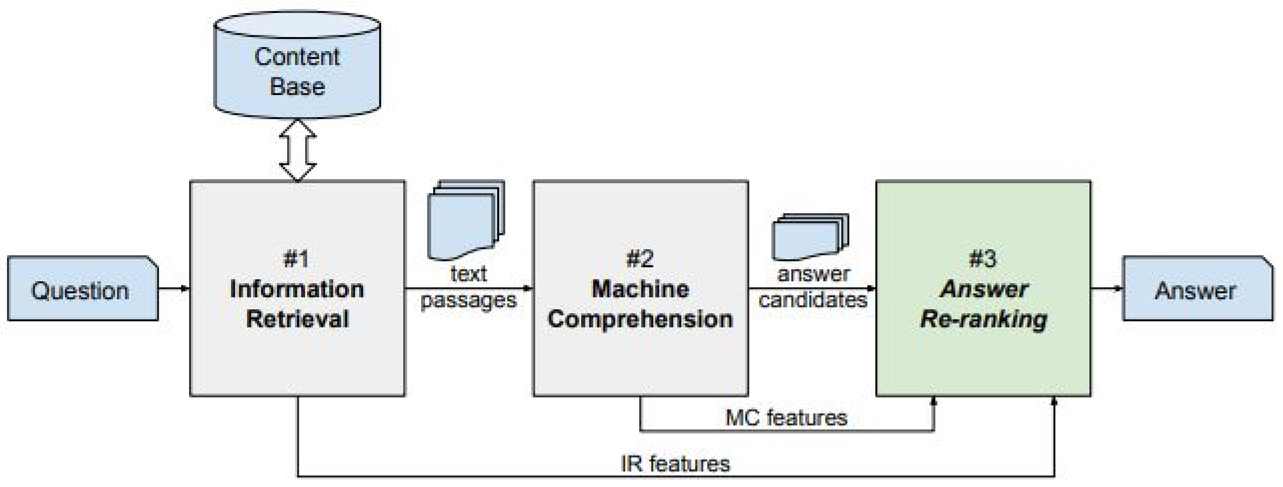
$$b_j = \frac{\exp(\mathbf{w} \cdot \mathbf{q}_j)}{\sum_{j'} \exp(\mathbf{w} \cdot \mathbf{q}_{j'})},$$

-

- **Machine Comprehension: Prediction**
  - From a given set of paragraph features we need to predict the answer from the paragraph
  - Example Paragraph:
    "Gigli is a 2003 American romantic comedy film written and directed by Martin Brest."
  - So, we need to look for the starting and ending position of the answer in a paragraph.
  - We build two classifiers, Pstart(i) and Pend(i) which scores the probability of i being start or end respectively
  - Pstart(i) X Pend(j) will be the probability of that part of paragraph(i->j) being the answer.
- **Results**

Method	Dev		Test	
	EM	F1	EM	F1
Dynamic Coattention Networks (Xiong et al., 2016)	65.4	75.6	66.2	75.9
Multi-Perspective Matching (Wang et al., 2016)†	66.1	75.8	65.5	75.1
BiDAF (Seo et al., 2016)	67.7	77.3	68.0	77.3
R-net†	n/a	n/a	71.3	79.7
DrQA (Our model, Document Reader Only)	**69.5**	**78.8**	70.0	79.0

- **RankQA**
  - **Conventional paradigm in QA(NN)**
    - **First Stage:** First stage is to narrow down the search space with information retrieval techniques
    - **Second Stage:** The next stage is to compute the features and train a Neural Network based on these features and rank the answers/paragraphs
  - **The Idea behind RankQA**
    - While computing the rank in the second stage the output or the importance of the first stage is ignored completely. To make use of this, a Re-Ranking Based architecture is proposed to completely include both first and second stage features to decide the final rank
  - **Overview of the Architecture**



  - **Re-Ranking:**
    - After getting top-k candidate answers from the MC module. We follow a three step procedure:
    - **Feature extraction:** We extract a set of information retrieval and machine comprehension features for every answer candidate directly from the individual modules of the QA pipeline.

        - **Answer aggregation:** It is frequently the case that several answer candidates are duplicates and, hence, such identical answers are aggregated
        - **Re-ranking network:** Every top-k answer candidate is re-ranked based on the features generated. We score each candidate answer with a two-layer feed-forward network.

$$f(x_i) = \mathrm{ReLU}(x_i A^T + b_1)\, B^T + b_2$$

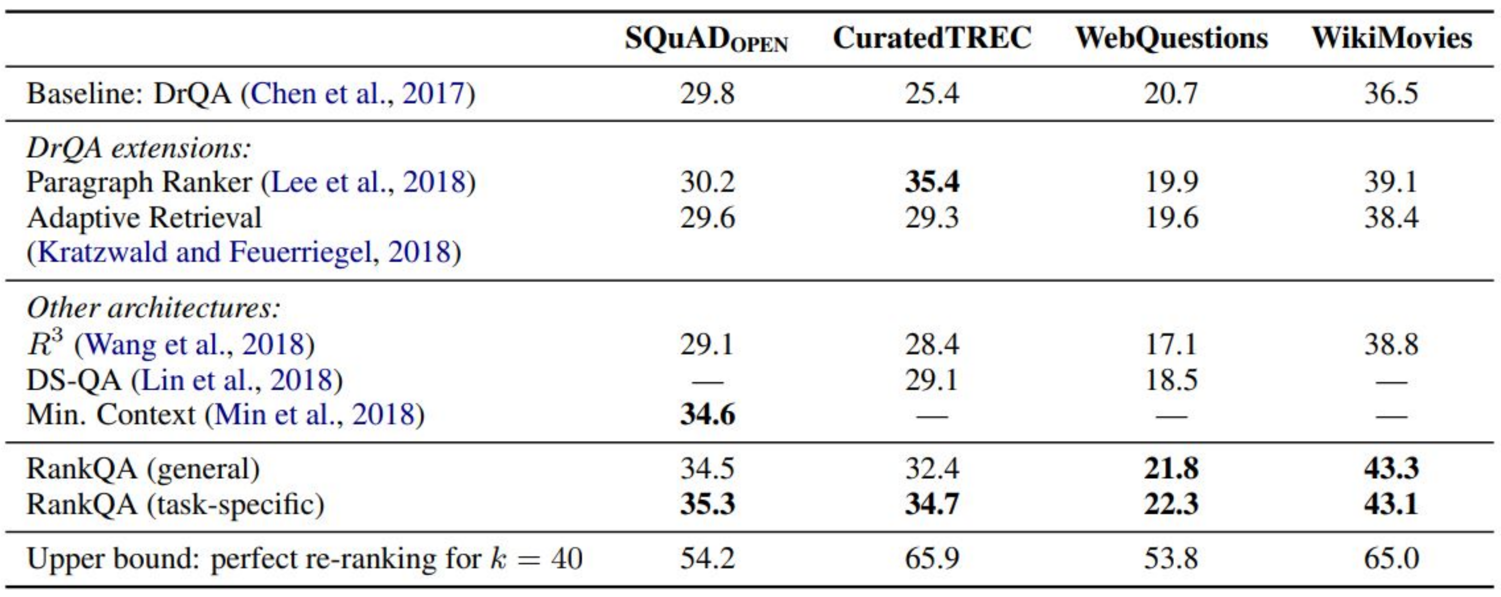
        - **Re-Ranking: Loss function:** With the following Loss function

$$\mathcal{L}_{\text{rank}}(x_i, x_j) = \Big[y_i - \sigma\left(f(x_i) - f(x_j)\right) \Big]^2$$

            - In order to deal with exploding weights, regularization term is added

$$\mathcal{L}_{\text{reg}} = \|A\|_1 + \|B\|_1 + \|b_1\|_1 + \|b_2\|_1$$

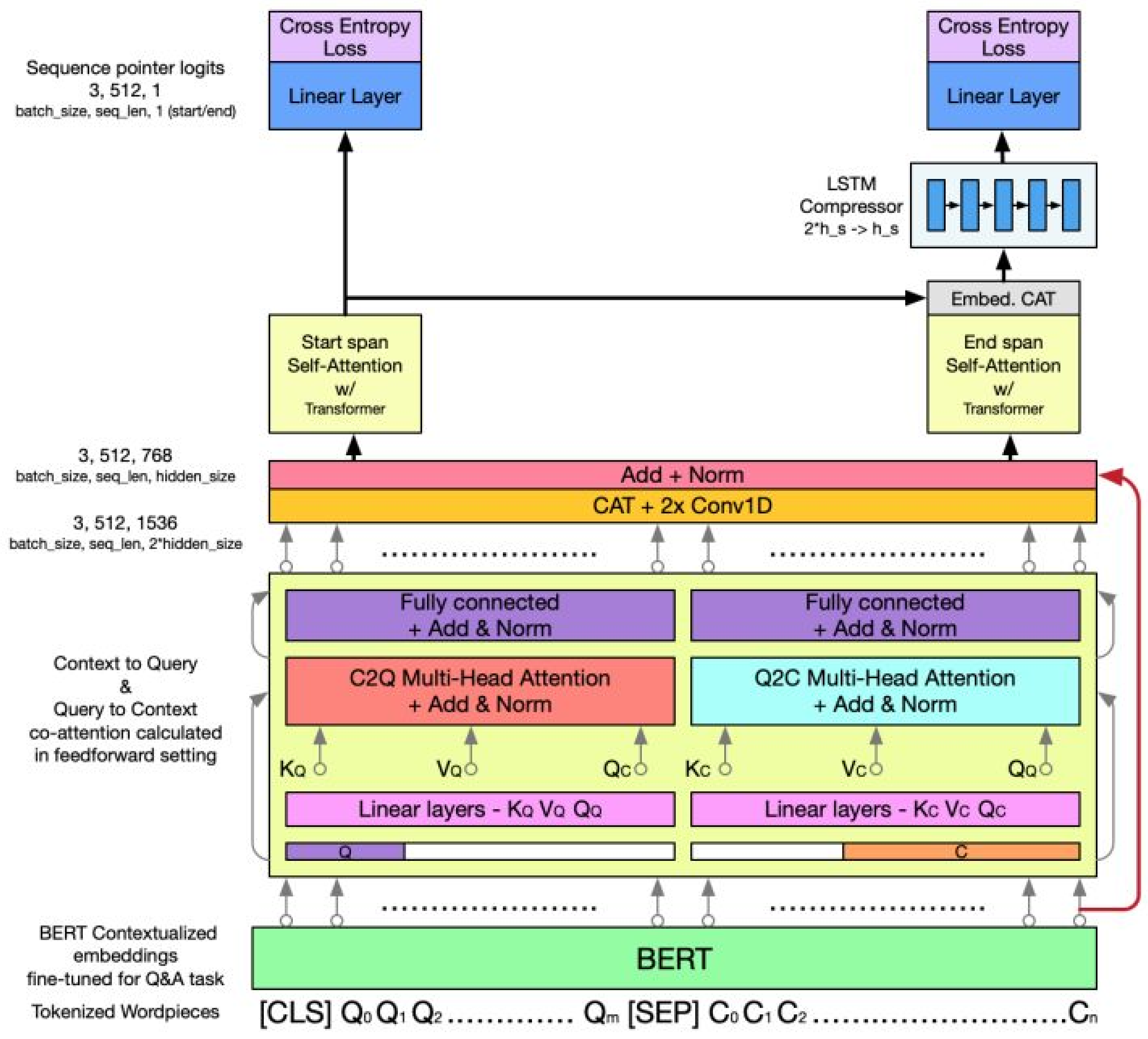
    - **Results**



	SQuAD$_{\text{OPEN}}$	CuratedTREC	WebQuestions	WikiMovies
Baseline: DrQA (Chen et al., 2017)	29.8	25.4	20.7	36.5
DrQA extensions:				
Paragraph Ranker (Lee et al., 2018)	30.2	**35.4**	19.9	39.1
Adaptive Retrieval (Kratzwald and Feuerriegel, 2018)	29.6	29.3	19.6	38.4
Other architectures:				
R^3 (Wang et al., 2018)	29.1	28.4	17.1	38.8
DS-QA (Lin et al., 2018)	—	29.1	18.5	—
Min. Context (Min et al., 2018)	**34.6**	—	—	—
RankQA (general)	34.5	32.4	**21.8**	**43.3**
RankQA (task-specific)	**35.3**	**34.7**	**22.3**	**43.1**
Upper bound: perfect re-ranking for $k = 40$	54.2	65.9	53.8	65.0

- **BERT-QA**
    - This paper is an extension to BERT aimed to increase performance on SQUAD 2.0 dataset. The Transformer architecture on which BERT is based places hierarchical global attention on the concatenation of the context and query. The major change when compared to BERT is more focused context to query (C2Q) and query to context (Q2C) attention via a set of modified Transformer encoder units.
    - **Methods:**
        - **Directed Coattention:**
            - The base BERT network, the baseline for this project, is built with 12 Transformer encoder blocks

- BERTQA uses directed coattention between the query and context, as opposed to attending to their concatenation



Cross Entropy Loss
Linear Layer
Sequence pointer logits
3, 512, 1
batch_size, seq_len, 1 (start/end)
Cross Entropy Loss
Linear Layer
LSTM Compressor
2*h_s -> h_s
Embed. CAT
Start span Self-Attention w/ Transformer
End span Self-Attention w/ Transformer
3, 512, 768
batch_size, seq_len, hidden_size
Add + Norm
CAT + 2x Conv1D
3, 512, 1536
batch_size, seq_len, 2*hidden_size
Fully connected + Add & Norm
Fully connected + Add & Norm
Context to Query & Query to Context co-attention calculated in feedforward setting
C2Q Multi-Head Attention + Add & Norm
Q2C Multi-Head Attention + Add & Norm
KQ VQ QC
KC VC QQ
Linear layers - KQ VQ QQ
Linear layers - KC VC QC
Q
C
BERT Contextualized embeddings fine-tuned for Q&A task
BERT
Tokenized Wordpieces
[CLS] Q0 Q1 Q2 .... Qm [SEP] C0 C1 C2 .... Cn

- The BERT embeddings are masked to produce separate query and context embedding vectors

$$E_q = E * m_q$$
$$E_c = E * m_c$$

- Where E is the contextualized embeddings derived from BERT, m is the mask, and c and q are the context and query respectively. E_q and E_c are then projected through linear layers to obtain key, value, and query vectors

$$P_x = W_P E_x^T + b_P \forall P \in Q, K, V, x \in q, c$$

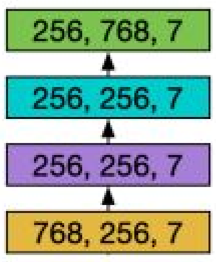
- Where Q, K, and V are the query, key and value vectors
- The Q, K, and V vectors are used in scaled dot-product attention to create the separate Context-to-Query (C2Q) and Query-to-Context (Q2C) attention vectors.

$$Coattention(Q, K, V) = softmax(\frac{Q_y K_z^T}{\sqrt{d_k}})V_z$$

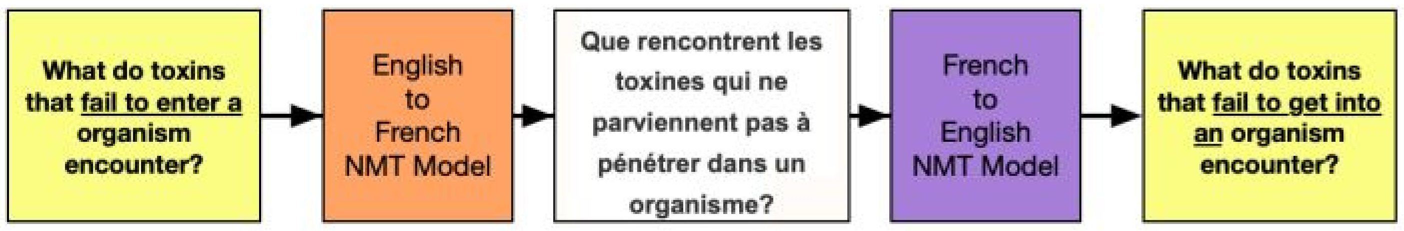
- Where y is q and z is c for Q2C and y is c and z is q for C2Q

▪ **Localized Feature Extraction**

- BERTQA added four convolutional layers within the coattention architecture. The input to these layers were the BERT embeddings and the outputs were added to the outputs of the multi-head attention layers in the coattention architecture and then layer-wise normalized. This combination of coattention and local information provides a hierarchical understanding of the question and context.



- **Skip Connections**
  - BERTQA added a skip connection from the BERT embedding layer combined with the convolved directed co-attention output
  - This allows the directed co-attention layers to learn distinct information coming from BERT embeddings via the skip and allows for efficient backpropagation to the BERT layers.
- **Data Augmentation**
  - For data augmentation Backtranslation using Google Cloud Translation API to translate the sentence from English to French and then do a back translation to English, essentially 2 translations per question


What do toxins that fail to enter a organism encounter?
English to French NMT Model
Que rencontrent les toxines qui ne parviennent pas à pénétrer dans un organisme?
French to English NMT Model
What do toxins that fail to get into an organism encounter?

- **Results:**

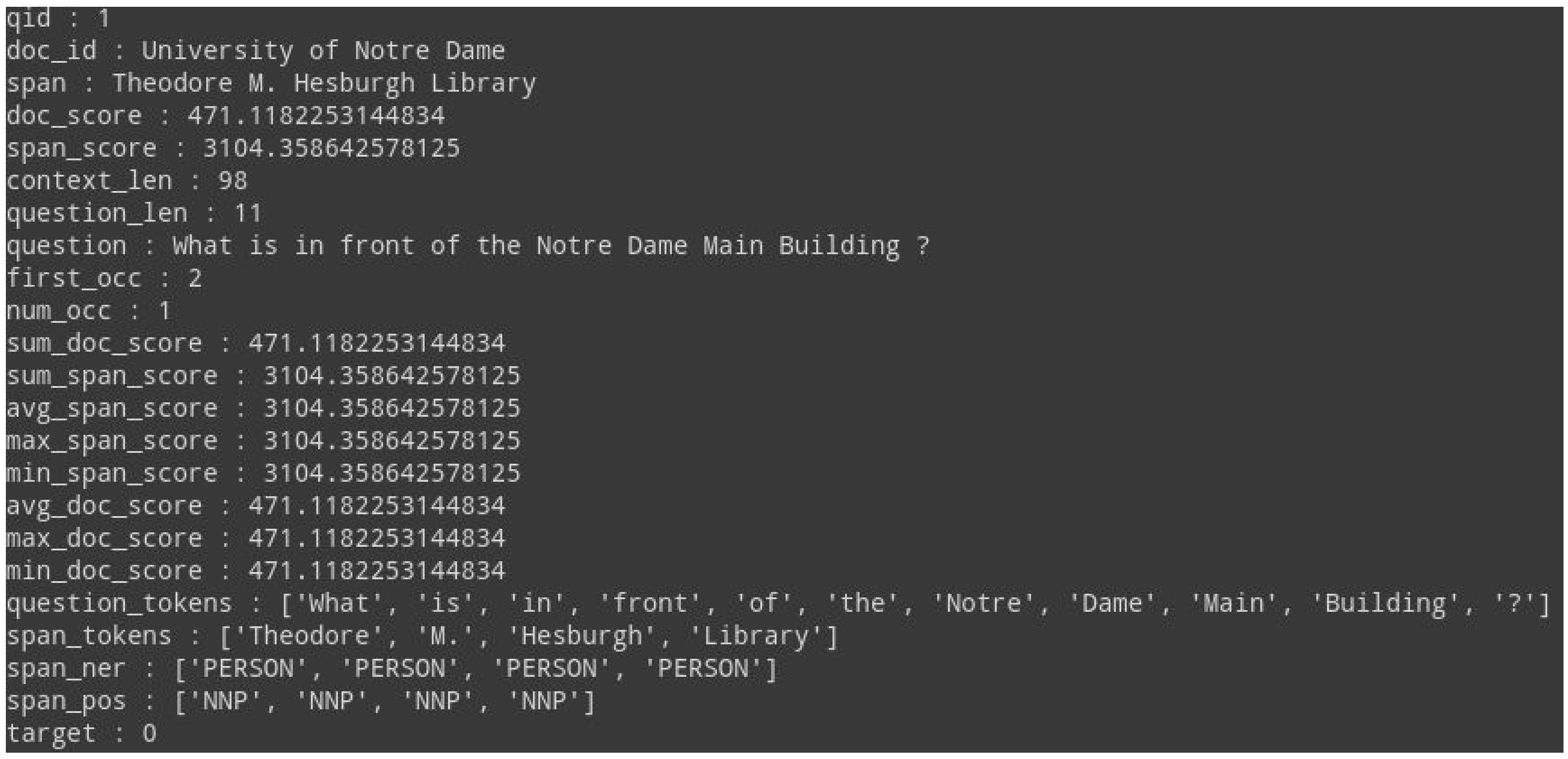
	Base	C2Q/Q2C	Simple Skip	Transformer Skip	Inside Conv
F1	74.15	74.34	74.81	74.95	**77.03**
Has Ans F1	**80.62**	76.30	76.13	76.35	78.47
No Ans F1	68.21	72.54	73.01	73.66	**73.83**
EM	71.09	71.56	72.11	72.07	**74.37**

	BERT Large	Model 1	Model 2	Model 3	Ensemble
F1	80.58	82.08	81.51	81.31	**83.42**
Has Ans F1	**84.39**	84.36	83.53	84.38	81.09
No Ans F1	77.08	79.98	79.68	78.49	**85.56**
EM	77.74	78.81	78.24	78.00	**80.53**

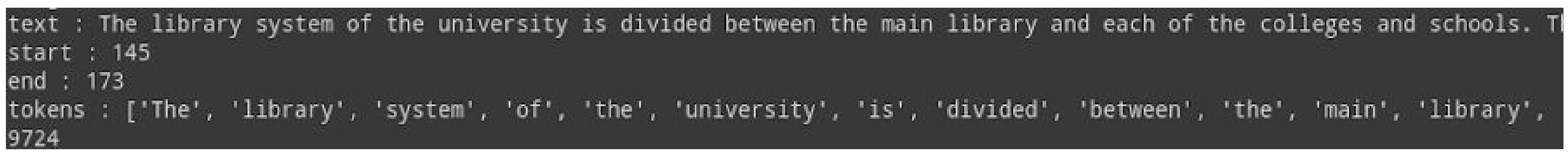
Dataset Used

We used the features extracted by the DrQA/BertQA models from the datasets

- SQUAD
- WikiMovies



```
qid : 1
doc_id : University of Notre Dame
span : Theodore M. Hesburgh Library
doc_score : 471.1182253144834
span_score : 3104.358642578125
context_len : 98
question_len : 11
question : What is in front of the Notre Dame Main Building ?
first_occ : 2
num_occ : 1
sum_doc_score : 471.1182253144834
sum_span_score : 3104.358642578125
avg_span_score : 3104.358642578125
max_span_score : 3104.358642578125
min_span_score : 3104.358642578125
avg_doc_score : 471.1182253144834
max_doc_score : 471.1182253144834
min_doc_score : 471.1182253144834
question_tokens : ['What', 'is', 'in', 'front', 'of', 'the', 'Notre', 'Dame', 'Main', 'Building', '?']
span_tokens : ['Theodore', 'M.', 'Hesburgh', 'Library']
span_ner : ['PERSON', 'PERSON', 'PERSON', 'PERSON']
span_pos : ['NNP', 'NNP', 'NNP', 'NNP']
target : 0
```



```
text : The library system of the university is divided between the main library and each of the colleges and schools. T
start : 145
end : 173
tokens : ['The', 'library', 'system', 'of', 'the', 'university', 'is', 'divided', 'between', 'the', 'main', 'library',
9724
```

This is how the dataset looks like. It contains the question, answer and the features extracted for every answer.

Our Model:

We built the model which was specified in the RankQA paper.

- We convert all the dataset answers into a feature vector.
- Then we pass this feature vectors of all the answers of each question through a trained feed forward neural network
- Then we take the all the answers and assign the correct answer to the answer which has the highest output i.e score(output of the neural network)
- For training we take some pair of answers from each question which are opposite to each other i.e one is correct and the other is wrong and we try to the minimize the objective which is given above in the paper

Experiments

- **Different Loss Function**
  - We tried using normal loss function i.e MSE loss
  - We also used the pairwise loss which was mentioned in the paper
- **Normalizing the features**
  - We built the model by normalizing the features along the axis of every features
  - We also tried without normalizing
- **Ablation Study**
  - We also made a study of which features are really affecting the model in the given features by removing some of the features and running the model.
  - Example: Removing NER, POS from the given features.

Results and Analysis

ABLATION STUDY AND NORMALIZING RESULTS:

We are checking the accuracy of the model to compare how our model works.

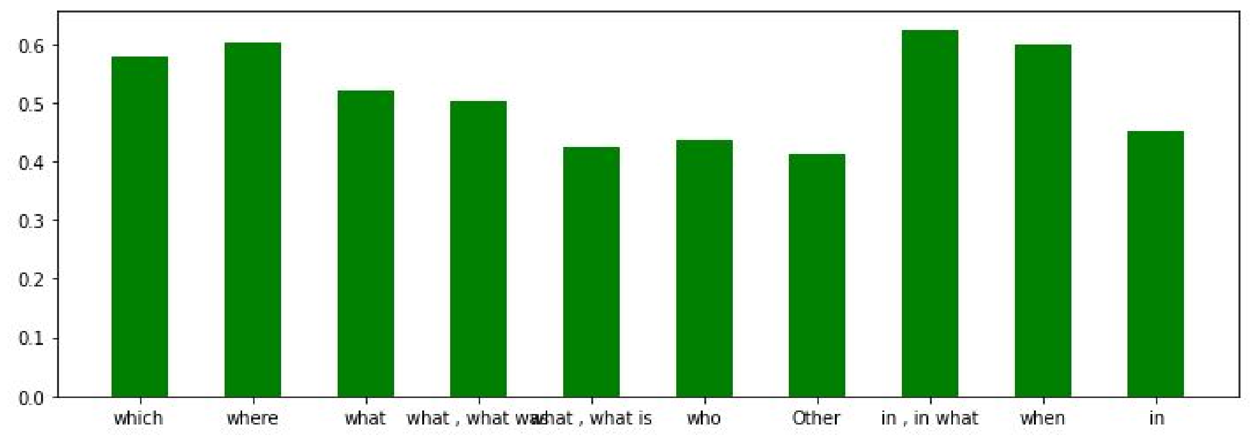
Features used	SQUAD	WIKI
Using All features	0.3466414380321665	0.43538987138263663
Without POS	0.34626300851466413	0.4320739549839228
Without NER	0.34427625354777674	0.4317725080385852
WIthout QTYPE	0.345600756859035	0.4325763665594855
Without Normalized features(scores like span score, doc score)	0.10302743614001893	0.27009646302250806
WIthout POS and NER	0.3454115421002838	0.42313102893890675
Without NER, POS, QTYPE	0.34626300851466413	0.4207194533762058
By normalizing scores	0.1512771996215705	0.21985530546623794
Without Doc Scores	0.15572374645222328	0.22086012861736334
Without Span Scores	0.15250709555345318	0.21875

- We observed that the most of the score was affected by the span score, doc score.
- There is not much change when we remove the features such as POS, NER, QTYPE
- We also observed that when we normalize the doc score, span score the accuracy of the model is falling drastically. The main reason may be that when we normalize these features the other features are dominating and the effect of the doc score and span score is totally not recognized by the NN model.

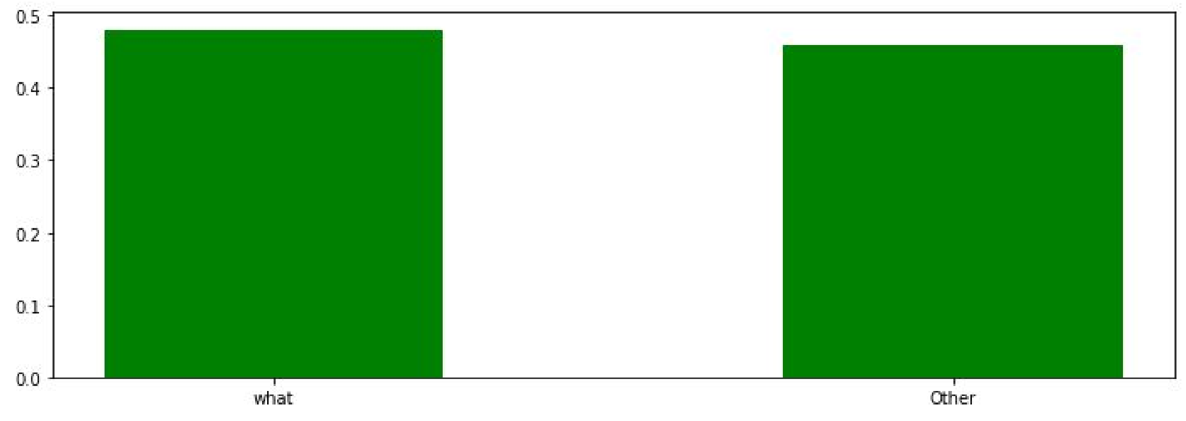
QUESTION ANALYSIS

We also tried to analyze which type of questions went wrong. But we were not able to come to any conclusions using this analysis because the model gave similar results for all the question types. There is no specific question type which was given more importance.

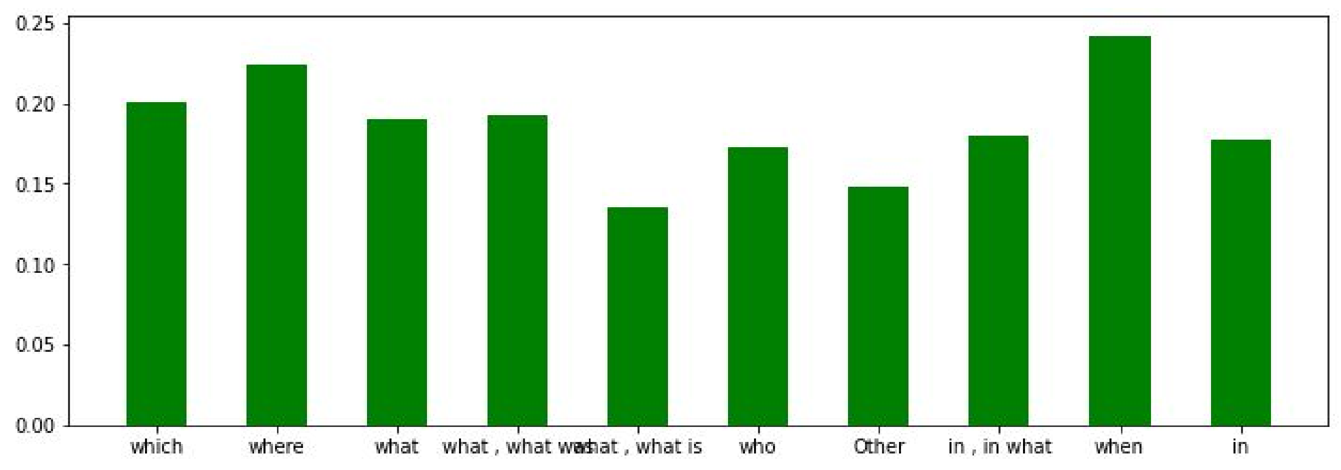
These images represent the no of wrong answer percentage of that category



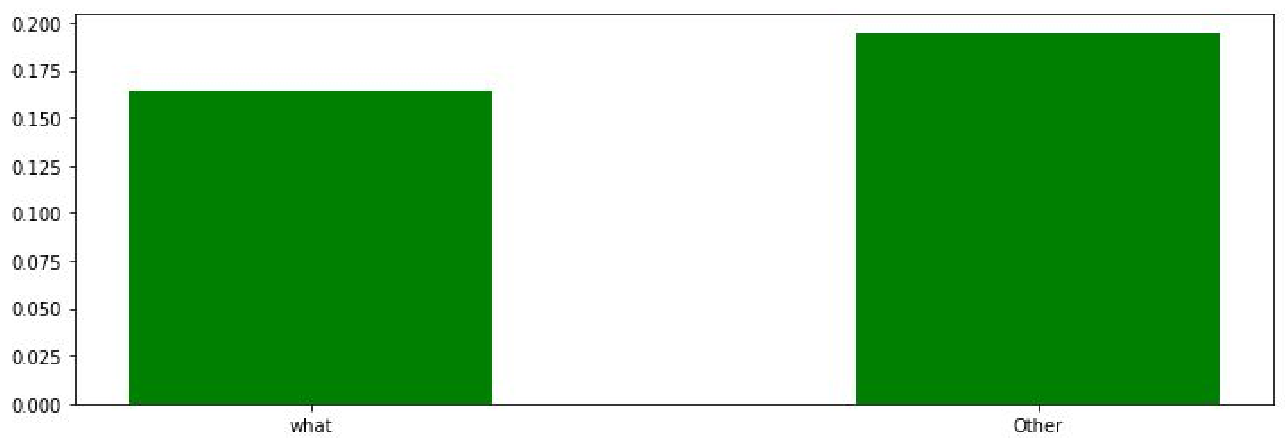
Initially for SQUAD(**before training**)



Wiki Movies(**before training**)



Squad(**After Training**)



Wiki Movies(**After Training**)

We can clearly see there is a significant drop in wrong answers percentage in each category. But it is a similar drop in each category.

OTHER LOSS FUNCTION:

- We tried a normal loss function which is MSE without comparing with answers given for the question instead of the given loss
- We observed that this does not give the best results. The reason may be that we are not comparing with the already present answers. We are just going forward without comparing.

	NORMAL LOSS	GIVEN LOSS
SQUAD	0.3089	0.3466
WIKIMOVIES	0.4113	0.4353

Limitations

- We believe this model may not work properly because some of the features are binary features and other are normal features. Due to this it may be tough for the neural model to learn the parameters.
- We require a lot of data for training the model.
- The model also takes a lot of time if we try to take all the pairs of answers(for the given loss function) to train the model (but all the things are not taken into consideration).

Future Scope

- As the model is really simple we can try different models in the last module of the model
- We can also try to observe the features really affecting the answers, extract them and try such new features.
- As the model is really simple we can results of different state of the models and try improving them using the re-ranking part

References

- DrQA https://arxiv.org/abs/1704.00051
- RankQA https://arxiv.org/pdf/1906.03008.pdf
- Bert https://arxiv.org/pdf/1810.04805.pdf
- BertQA https://arxiv.org/pdf/1912.10435.pdf