CIS 530 Spring 2021, Project Report

Machine Reading Comprehension on SQUAD 2.0

Tong Fu,[*] Yuchen Ding,[†] Xiaonan Liu,[‡] Simeng Shen,[§] and Zihan Zhou[¶]
University of Pennsylvania

(Dated: December 4, 2023)

Abstract

In this project, we aim to build a model for question answering. We train the model on Stanford Question Answering Dataset 2.0 (SQuAD 2.0). We have experimented on different models including Sliding Window, Logistic Regression, DistilBERT, RoBERTa, and SqueezeBERT for question answering task. We use F1 score and exact match as two metrics to evaluate the model. The published base line Logistic Regression has 0.36 F1 and 0.46 EM on test set, and the best model SqueezeBERT has 0.67 F1 and 0.71 EM on test set.

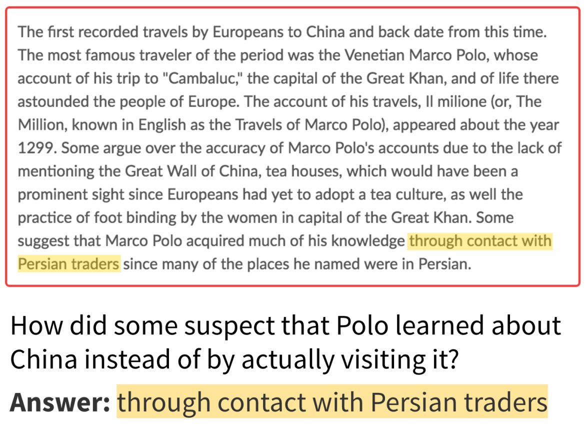
I. INTRODUCTION

A question answering implementation, usually a computer program, may construct its answers by querying a structured database of knowledge or information, usually a knowledge base. More commonly, question answering systems can pull answers from an unstructured collection of natural language documents. In this project, we aim to build a machine reading comprehension model to answer the questions from the given context.

The data we will be using is Stanford Question Answering Dataset 2.0 (SQuAD 2.0), which is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable. The task of this project is to predict the position(s) in the text corresponds to the answer, given a context paragraph and a question. An example is shown in Figure 1.

Nowadays, question answering systems on factoid questions are used in many aspects of our lives, such as search engines, online chatbots, but the ability to read a piece of text and then answer related questions is a totally different task, since it requires both the understanding of natural language and the world. As one of the key problems in the domain of Natural Language Processing, machine reading comprehension remains a challenge for scholars and researchers. This long-standing goal of teaching machines to read and comprehend human languages has been much closer to the end with the great advancements in deep neural networks and contextualized language models.

To reach the level of human performance and possibly outperform on these comprehension questions is the goal of this project. As shown on the leaderboard of the SQuAD dataset, the Exact Match for


The first recorded travels by Europeans to China and back date from this time. The most famous traveler of the period was the Venetian Marco Polo, whose account of his trip to "Cambaluc," the capital of the Great Khan, and of life there astounded the people of Europe. The account of his travels, Il milione (or, The Million, known in English as the Travels of Marco Polo), appeared about the year 1299. Some argue over the accuracy of Marco Polo's accounts due to the lack of mentioning the Great Wall of China, tea houses, which would have been a prominent sight since Europeans had yet to adopt a tea culture, as well the practice of foot binding by the women in capital of the Great Khan. Some suggest that Marco Polo acquired much of his knowledge through contact with Persian traders since many of the places he named were in Persian.

How did some suspect that Polo learned about China instead of by actually visiting it?

Answer: through contact with Persian traders

Figure 1. Illustrative example of our task, including a paragraph from an high-quality Wikipedia article, an accompanying reading comprehension question, and one or more answers that are segments of text in the passage.

[*] tongfu@seas.upenn.edu
[†] ycding@seas.upenn.edu
[‡] xiaonan6@seas.upenn.edu
[§] shensm@seas.upenn.edu
[¶] zzihan@seas.upenn.edu

human performance is 86.831%, and the F1 score is 89.452%, whereas the highest ranking FPNet has achieved a EM score of 90.871% and a F1 score of 93.183%. These outstanding results have presented us the unlimited capacity of deep neural networks, and as a group, we wanted to dive deeper into these models and learn more about computational linguistics.

II. PREVIOUS WORK ON MACHINE READING COMPREHENSION

In the paper "Teaching Machines to Read and Comprehend", the authors have developed attention based deep neural networks to solve the problem of question answering with little prior knowledge of language structure. [2] They have prepared two corpora of news stories from the CNN and Daily Mail websites, and turned them into document-query-answer tuples for training. Two simple baselines are first established: the majority baseline and the exclusive majority, and then two NLP-centric models are used for comparison: frame-semantic parsing and word distance benchmark.

Importantly, the authors propose three neural models: the Deep LSTM Reader, the Attentive Reader and the Impatient Reader. The first approach uses a two layer deep LSTM with one single encoded sequence for each document query pair so that the network can predict which token answers the query. Then, the second approach employs two bidirectional single layer LSTMs to compute attention scores for each token in the document given the query, and the last Impatient Reader is built upon the Attentive Reader with additional ability to reread as each query token is read. The results from the evaluation show clear evidence that the Attentive and the Impatient readers outperform the other models with 70.5 and 69.0 accuracies, respectively, and the Deep LSTM also gives reasonable outputs with 63.3 accuracy. In the end, the authors conclude that Attentive and Impatient Readers are able to propagate and integrate semantic information over long distances, and this attention mechanism is the key to such improvement.

Natural Language Computing Group in Microsoft Research Asia published their work "R-NET: Machine Reading Comprehension With Self-Matching Networks". It focuses on an end-to-end neural networks model, R-NET, that aims to answer questions from a given passage. [6] For reading comprehension style question answering, the task is to predict an answer A to question Q based on information found in P(passage). The paper mainly focuses on both the SQuAD that composed of 100,000+ questions and the MS-MARCO dataset which is manually created through crowdsourcing.

The R-NET model consists of four parts, which is the recurrent network, the gated matching layer, the self-matching layer, and the pointer network. First, the question and passage are processed by a bidirectional recurrent network and are matched with gated attention-based recurrent networks which accounts for the fact that words are of different importance in the passage for different reading comprehension tasks. The words are first converted to their respective word-level embeddings using pre-trained case-sensitive GloVe embeddings and character-level embeddings which is helpful to deal with out-of-vocab tokens. The authors choose to use Gated Recurrence Unit(GRU) in the experiment of using bi-directional RNN to produce new representations of all words, as it performs similarly to LSTM but with a cheaper computational cost. Next, through gated attention-based recurrent networks, question-aware passage representations is generated, with a very limited knowledge of context and some lexical divergence between the questions and passage. This problem is addressed by directly matching the question-aware passage representation against itself that extracts evidence from the entire passage. Lastly, to predict the start and end position of the answer, a pointer network is used with an additional attention-pooling layer to generate the initial hidden vector.

The paper trained and evaluated the model mainly on the SQuAD dataset. Exact Match (EM) and F1 score are two metrics that are utilized to evaluate the model. The model outperforms the baseline and other competing models proposed by former researchers, with a EM of 72.3 and F1 of 80.7 (single model) and EM 76.9 and F1 84.0 (ensemble model) on the test set.

"Retrospective Reader for Machine Reading Comprehension" propose a new method, retrospective reader, to solve machine reading comprehension problems. There are two parallel modules in the retrospective reader: a sketchy reading module and an intensive reading module to conduct a two-stage reading process. [11] The first module sketchy reading scans the document and makes a rough judgment (external front verification) about whether the question is answerable. Then, the intensive reading module jointly predicts the candidate answers and combines its answerability confidence with the sketchy judgment score to yield the final output.

Sketchy reading module consists of embedding, interaction, and external front verification. Intensive reading module consists of question-aware matching, span prediction, internal front verification and Threshold-based Answerable Verification. This state-of-the-art model Retro-Reader on ELECTRA yields a satisfactory performance with Precision 93.7, Recall 93.51, F1 93.39, and 87.60 accuracy.

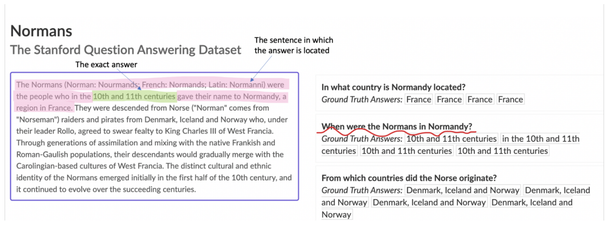
III. EXPERIMENTAL DESIGN

A. Data Description

The dataset we used is the Stanford Question Answering Dataset (SQuAD2.0) `https://rajpurkar.github.io/SQuAD-explorer/`. It consists of the context, which is the reading passage, questions posed by crowdworkers on a set of Wikipedia articles, as well as the answer to every question. Each answer is either a span (or segment) of text from the corresponding reading passage, or it could also indicate that the question might be unanswerable. We used version 2.0 of the dataset, an updated version that not only includes the 100,000 questions in the old SQuAD 1.1 dataset, but also combines with over 50,000 unanswerable questions written adversarially by crowdworkers.

An example of one of the Q&A passage selected from the training set [1] is shown in Figure 2. SQuAD 2.0 dataset downloaded is in json format. The example in json format is shown in Appendix A.

The size of the dataset is shown in Table I. We



Normans

The Stanford Question Answering Dataset

The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France. They were descended from Norse ("Norman" comes from "Norseman") raiders and pirates from Denmark, Iceland and Norway who, under their leader Rollo, agreed to swear fealty to King Charles III of West Francia. Through generations of assimilation and mixing with the native Frankish and Roman-Gaulish populations, their descendants would gradually merge with the Carolingian-based cultures of West Francia. The distinct cultural and ethnic identity of the Normans emerged initially in the first half of the 10th century, and it continued to evolve over the succeeding centuries.

In what country is Normandy located?
Ground Truth Answers: France France France France

When were the Normans in Normandy?
Ground Truth Answers: 10th and 11th centuries in the 10th and 11th centuries 10th and 11th centuries 10th and 11th centuries

From which countries did the Norse originate?
Ground Truth Answers: Denmark, Iceland and Norway Denmark, Iceland and Norway Denmark, Iceland and Norway Denmark, Iceland and Norway

Figure 2. Sample of dataset.

Table I. Size of the dataset.

Data	Contexts	Questions and Answers
Train	18,877	86,821
Test	1204	5,928

Table II. Updated size of the dataset.

Data	Contexts	Questions and Answers
Train	13,214	60,397
Dev	5,663	25,884
Test	1204	5,928

also split the train set into train/validation by 7:3, so the updated size is shown in Table II.

B. Evaluation Metrics

We follow the approaches in Rajpurkar's paper "SQuAD: 100,000+ Questions for Machine Comprehension of Text" [8]. The SQuAD dataset can be used with two metrics. The first one is Macro-averaged F1 Score. It measures the average overlap between the prediction and ground truth answer. Both should be treated as bags of tokens, and used to compute F1. However, since the dominant models already have very high F1 scores that are above 90%, further advance can be very marginal. Thus, to better measure the differences in model performances, a second metric of Exact Match is used. This metric measures the percentage of predictions that match the ground truth answers exactly. Since this is a tougher metric, it can better reflect improvement in models.

There are two things to note for. First, both metrics ignores punctuation and articles (a, an, the). Second, there may be multiple ground truth answers, thus for each question, we use the one that is closest to our prediction. Also, for the strong baseline, we also introduced accuracy (to compensate for the inaccuracy of how exact match is calculated).

The equations are shown below:

```
    pred_bag = bag of words of predictions
    true_bag = bag of words of gold
standard
    overlap = overlap between (pred_bag,
true_bag)
    precision = len(overlap) /
len(pred_bag)
    recall = len(overlap) / len(true_bag)
    F1_score = (2 * precision * recall) /
(precision + recall)
    ExactMatch = (pred_bag == true_bag)
```

IV. EXPERIMENTAL RESULTS

A. Simple Baseline: Sliding Window

The simple baseline follows the approach in Rajpurkar's paper "SQuAD: 100,000+ Questions for Machine Comprehension of Text" [8], where it selects candidate answers based on constituency parsing and excludes the answer string from its sentences. To reduce the complexity of our baseline, we used a simpler method selecting possible answers by using each sentence as possible answers. For each candidate answer, we compute the bi-gram overlap between the answer and the question. We keep all the candidates that have the maximal overlap. After so, we select the best one using the sliding-window approach proposed in Richardson et al. (2013) [9]. The algorithm computes the maximum sum of log of uni-gram overlap over the entire window. The detailed algorithm is shown in Figure 3.

A summary of our simple baseline result is shown in Table III.

Table III. Metrics of Sliding Window.

Data	F1	Exact Match
Train	0.091	0
Test	0.243	0.212

Baseline Algorithms

Require: Passage P, set of passage words PW, i^{th} word in passage P_i, set of words in question Q, set of words in hypothesized answers $A_{1..4}$, and set of stop words U,

Define: $C(w) \coloneqq \sum_i \mathbb{I}(P_i = w)$;

Define: $IC(w) \coloneqq \log\left(1 + \frac{1}{C(w)}\right)$.

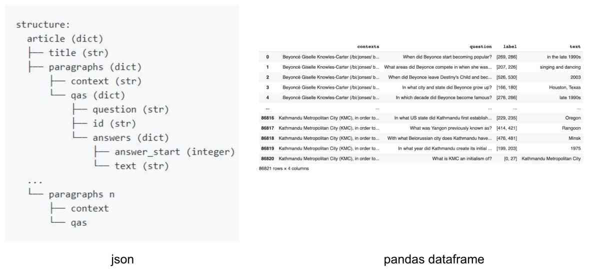
Algorithm 1 Sliding Window

```
for $i = 1$ to 4 do
    $S = A_i \cup Q$
    $sw_i = \max_{j=1..|P|} \sum_{w=1..|S|} \begin{cases} IC(P_{j+w}) & if\ P_{j+w} \in S \\ 0 & otherwise \end{cases}$
end for
return $sw_{1..4}$
```

Figure 3. Simple baseline structure.

B. Published Baseline: Logistic Regression

The structure shown in Figure 4 is the format of the input dataset. We have a training set and a dev set that contain the same structure of data. The title, context, answer, question, start position of the answer, and id are included. We first transform the json dataset to pandas dataframe so that it's more intuitive for further modeling. The label column is the start and the end position of the answer.



```
structure:
  article (dict)
  ├── title (str)
  ├── paragraphs (dict)
      ├── context (str)
      └── qas (dict)
          ├── question (str)
          ├── id (str)
          └── answers (dict)
              ├── answer_start (integer)
              └── text (str)
  ...
  └── paragraphs n
      ├── context
      └── qas
```

	contexts	question	label	text
0	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ b...	When did Beyonce start becoming popular?	[269, 286]	in the late 1990s
1	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ b...	What areas did Beyonce compete in when she was...	[207, 226]	singing and dancing
2	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ b...	When did Beyonce leave Destiny's Child and bec...	[526, 530]	2003
3	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ b...	In what city and state did Beyonce grow up?	[166, 180]	Houston, Texas
4	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ b...	In which decade did Beyonce become famous?	[276, 286]	late 1990s
...	...	...	...	...
86816	Kathmandu Metropolitan City (KMC), in order to...	In what US state did Kathmandu first establish...	[229, 235]	Oregon
86817	Kathmandu Metropolitan City (KMC), in order to...	What was Yangon previously known as?	[414, 421]	Rangoon
86818	Kathmandu Metropolitan City (KMC), in order to...	With what Belorussian city does Kathmandu have...	[476, 481]	Minsk
86819	Kathmandu Metropolitan City (KMC), in order to...	In what year did Kathmandu create its initial ...	[199, 203]	1975
86820	Kathmandu Metropolitan City (KMC), in order to...	What is KMC an initialism of?	[0, 27]	Kathmandu Metropolitan City

86821 rows × 4 columns

Figure 4. Dataset structure and pandas dataframe.

The strong baseline we reproduced is a simplified version of the Multinomial Logistic Regression model published by Rajpurkar et.al in 2016 [8]. In this paper, the author built a total of 180 million features (mostly lexicalized or dependency tree path features) and discretized each continuous feature into 10 equally sized buckets. To help the model pick the correct sentences, the paper matches word and bigram frequencies as well as the root match features. However, it is not clear in the paper how they have defined the target variable. Thus for the strong baseline model, we are only focusing on selecting the correct sentence and only implemented the feature based on cosine distance. To do so, we transformed the variables from text to sentence index having that text. We restricted the paragraph length to be only 10 sentences so that all paragraphs have the same length. If a paragraph is less than 10 sentences, the corresponding missing columns will be replaced by the maximum cosine distance which is 1.

Moreover, we also tried to include a dependency parse tree feature, which matches the root of the question to all the subroots as there is a higher possibility that the sentence is within that sentence. However, this didn't improve the accuracy and F1 score significantly, so we decided to not include this feature.

For embedding, we used the sentence2vec (InferSent [4]) that uses vectors to represent entities numerically. It creates a vocabulary from the training data and use this vocabulary to train inferSent model. Once the model is trained, InferSent provide sentence as input to the encoder function which will return a 4096-dimensional vector irrespective of the number of words in the sentence. InferSent provides semantic representations for English sentences (trained with GloVe) such that the embeddings can be used further for finding similarities between sentences. This actually took the longest time in this project as the dataset is huge. Then, we broke the paragraph/context into multiple sentences using TextBlob and created the main feature of distance that includes the cosine similarity between each sentence-question pair.

We split the train/validation into 7:3 and sent the data to train using the multinomial logistic regression model as suggested in the published paper. We evaluated the test data and got the accuracy and F1 score. For the strong baseline model, since we are only trying to locate to the correct sentence that contains the answer, there is no Exact Match score. We wrote an evaluation metrics to calculate the accuracy, instead of EM. Although it's not as accurate as expected, it's still a huge improvement from our simple baseline model. A summary of our strong baseline result is shown in Table IV.

Table IV. Metrics of Logistic Regression.

Data	F1	Accuracy
Train	0.37	0.46
Dev	0.36	0.46
Test	0.38	0.46

C. DistilBERT on SQuAD 1.0

To better examine the results of BERT-based models, we performed a experiment on SQuAD 1.0, the ancestor of SQuAD2.0, which has no impossible questions. We suspect that impossible answers may be a huge factor influencing the performance. We used the same pre-trained model of DistilBERT, keeping all things the same. It achieved 0.792 F1 score and 0.696 EM score on SQuAD 1.0. As a comparison shown in Table V, it achieved 0.565 F1 score and 0.468 EM score on SQuAD 2.0. It can be inferred that many errors occur when the model tries to decide whether a question is answerable or not. Since wrongly classify an impossible question brings down both metrics down by a large amount, it explains why DistilBERT did not perform as well in SQuAD 2.0.

Table V. Metrics of pre-trained DistilBERT.

Data	F1	Exact Match
SQuAD 1.0	0.792	0.696
SQuAD 2.0	0.565	0.468

D. DistilBERT on SQuAD 2.0

Even though transfer learning from pre-trained models becomes accessible nowadays in the area of NLP, the computational cost of those large models is still a challenging issue. Thus, DistilBERT is introduced as a smaller pre-trained language model. DistilBERT can be fine-tuned with good performances on tasks such as question answering. Lots of prior work examined the use of distillation for building task-specific models, DistilBERT leverages knowledge distillation during the pre-training phase. Knowledge distillation, also known as teacher-student learning, is a compression technique that a small model is trained to reproduce the behavior of a larger model. DistilBERT is able to reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster. To compensate for the inductive biases learned by larger models during pre-training, DistilBERT utilizes a triple loss combining language modeling, distillation and cosine-distance losses. Thus, DistilBERT is computationally cheaper to pre-train while is able to maintain a good performance. [10]

We start with the pre-trained DistilBERT model and fine-tune with our dataset. For time consideration, we only trained for two epochs. For each epoch, we feed the model with batch of size 32, resulting 4073 iterations in one epoch.

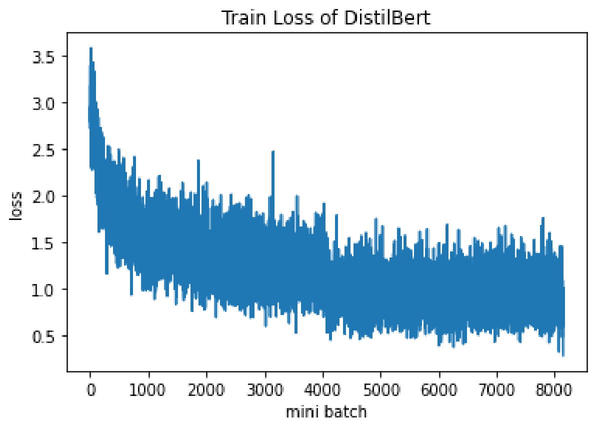


Figure 5. Train loss of DistilBERT after 2 epochs.

Figure 5 shows the training loss against the number of batches.

Table VI. Metrics of DistilBert.

Data	F1	Exact Match
Train	0.622	0.470
Test	0.565	0.468

Table VI shows the F1 score and exact match accuracy of DistilBert. We see that the loss is oscillating very strongly.

Table VII shows the F1 score and exact match accuracy of DistilBert answering different types of questions depending on whether the question is possible to answer.

Table VII. Metrics of DistilBert with questions divided if possible to answer.

Question possible to answer	F1	Exact Match
possible	0.530	0.335
impossible	0.601	0.601

E. RoBERTa

The RoBERTa model was proposed in *RoBERTa: A Robustly Optimized BERT Pretraining Approach* [5], which is based on Google's BERT model. It builds on BERT and modifies key hyperparameters, removing the next-sentence pretraining objective and training with much larger mini-batches and learning rates. This allows RoBERTa to improve on the masked language modeling objective compared with BERT and leads to better downstream task performance. The changes in training causing a substantial increase in performance leads to the believe that BERT was relatively undertrained.

We start with the pre-trained "distilroberta-base" tokenizer and model. For time consideration, we only trained for two epochs. For each epoch, we feed the model with batch of size 32, resulting 4073 iterations in one epoch.

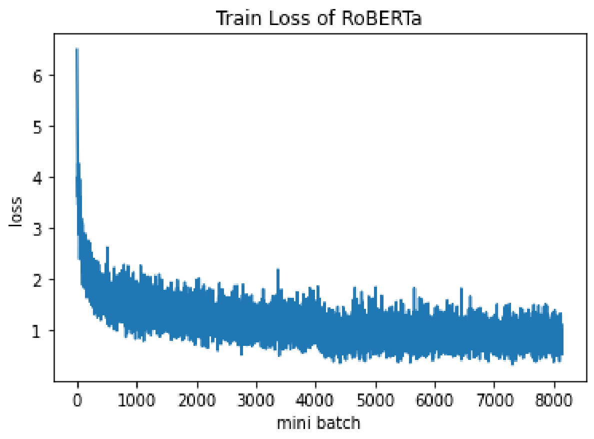


Figure 6. Train loss of RoBERTa after 2 epochs.

Figure 6 shows the training loss against the number of batches. We see that the loss is oscillating after converging to about 1, though it has improvement from DistilBERT. This may indicate that we have not use the best batch size, and the mini-batch stochastic gradient descent results the fluctuation. The convergence rate of RoBERTa is faster than that of DistilBERT.

Table VIII. Metrics of RoBERTa.

Data	F1	Exact Match
Train	0.699	0.547
Dev.	0.637	0.545

Table VIII shows the F1 score and exact match accuracy of RoBERTa. We see that the train scores do not outperform the test scores much, indicating that the model is a little underfitted. This is expected since we only trained for two epochs, and we can anticipate further improvements if time and RAM permitted. After all, the result has substantial improvement from the published baseline.

Table IX. Metrics of RoBERTa with questions divided if possible to answer.

Question possible to answer	F1	Exact Match
possible	0.522	0.337
impossible	0.752	0.752

Table IX shows the F1 score and exact match accuracy of RoBERTa answering different types of questions depending on whether the question is possible to answer. We see that the model performs much better on impossible questions.

F. SqueezeBERT

As we experimented above, although BERT and RoBERTa can achieve very high performance already, they are extremely computationally expensive. So, we experimented on another less time-consuming model, proposed in *SqueezeBERT: What can computer vision teach NLP about efficient neural networks?* [3]. They use methods such as grouped convolutions to yield significant speedups for computer vision networks, which many of these techniques have not been adopted by NLP neural network designers. The model replaces several operations in self-attention layers with grouped convolutions, and use this technique in a novel network architecture called SqueezeBERT, which runs 4.3x faster than BERT-base.

Table X. Metrics of SqueezeBert.

Data	F1	Exact Match
Train	0.903	0.893
Dev.	0.709	0.674

As shown in Table X, the testing F1 and EM score are both better than DistilBERT and RoBERTa. It is because both models we used above are distilled versions of BERT and RoBERTa. We anticipate that a full BERT-Large model would outperform these. However, due to the lack of memory space and computational power, SqueezeBERT is the best model we have. Although we cannot conclude that SqueezeBERT is superior , we think it is reasonable to say that SqueezeBERT offers a compelling speed-accuracy tradeoff for NLP tasks especially for SQuAD2.0.

V. ERROR ANALYSIS

First, we see that our deep learning models substantially improve the results from the baselines. The published baseline Logistic Regression can only predict the answer is sentence, and our deep learning models locate the answers in words.

We use the results of RoBERTa in further analysis. We see that the F1 score and exact match accuracy of RoBERTa answering different types of questions depends on whether the question is possible to answer. We see that the model performs much better on impossible questions as shown in Table IX. We illustrate a few examples where the model made errors.

The following is a part of a test context for questions and answers:

```
The Normans (Norman: Nourmands; French:
Normands; Latin: Normanni) were the people
who in the 10th and 11th centuries gave
```

```
their name to Normandy, a region in France.
They were descended from Norse ("Norman"
comes from "Norseman") raiders and pirates
from Denmark, Iceland and Norway who, under
their leader Rollo, agreed to swear fealty
to King Charles III of West Francia.
```

A question asks **When were the Normans in Normandy?** The predicted answer is **10th and 11th**, while the true answer is **10th and 11th centuries** or **in the 10th and 11th centuries**. We see that the word "centuries" is missing in the prediction, while the word is important in postulating the meaning of the answer. This kind of error should be avoided given enough training.

Another question asks **From which countries did the Norse originate?** The prediction indicates that the answer is unanswerable, while the true answer is **Denmark, Iceland and Norway**. A plausible reason is that the context uses "pirates from" before the true answer, and the model fails to identify that "pirate from" has the same meaning as "originate". This level of error is hard to avoid at this point, as the model needs to identify synonyms. We may use part of speech tagging and named entity relations in further training, but that is a different story.

The following shows another part of a test context for questions and answers:

```
The Norman dynasty had a major political,
cultural and military impact on medieval
Europe and even the Near East.
```

A question asks **What type of major impact did the Norman dynasty have on modern Europe?** The predicted answer is **political, cultural and military**, while the true answer is that the question is answerable. The question easily confused the model as it has many key words for the model to locate answer, i.e. "major impact", "the Norman dynasty", and "Europe". However, the question asks about "modern Europe", but the predicted answer is about "medieval Europe". This kind of error needs a more thorough investigation of the context, and a more robust model is needed.

Here we identify three kinds of errors, and we postulate that more training can fix the error where the predicted answer lacks important parts, but a stronger model is needed to identify synonyms in questions and contexts, and to detect the confusions built in the questions.

VI. CONCLUSION AND DISCUSSION

In this project, we have tried to implement a question answering model and we utilize Stanford Question Answering Dataset 2.0 (SQuAD 2.0). The metrics we employ are F1 score and Exact Match. We have experiments on several models to achieve the goal. First of all, we have tried several baseline models, including simple Baseline Sliding Window, a published baseline Logistic Regression. Then, we have tried more advanced deep learning neural models, including DistilBERT, RoBERTa, and SqueezeBERT.

Table XI. Comparison between different models

Model	Dev. F1	Dev. Exact Match
Sliding Window	0.24	0.21
Logistic Regression	0.36	0.46
DistilBERT	0.57	0.47
RoBERTa	0.64	0.55
SqueezeBERT	0.67	0.71

Table XI shows the F1 score and exact match accuracy of different models we have experimented on. We notice that model SqueezeBERT yields the best performance. The F1 on development set is 0.71 and the Exact Match on the development set is 0.67. We also see that the BERT based models perform better on questions impossible to answer.

For future improvements, first of all, we can have more training epochs and test different batch sizes and learning rates, even optimization methods for deep learning models. Given the lack of time in this project, we are regret that this is not achieved.

We may also add an additional binary classifier to predict whether the question is answerable, which we can then train jointly by summing the classification and span loss terms. During evaluation, we only need to predict span indices on pairs that are classified as answerable. The additional binary classifier is expected to substantially improve the results of our deep learning models.

ACKNOWLEDGMENTS

We appreciate the assistance and guidance from Prof. Mark Yatskar, and every teaching assistant throughout the semester.

Also, we want to appreciate Aryan Sakhala for his tutorial on building DistilBert for question answering.

All pre-trained models are from Huggingface.

BIBLIOGRAPHY

[1] *Explore SQuAD2.0.* `https://rajpurkar.github.io/SQuAD-explorer/`. Accessed: 2021-05-06.

[2] K. M. Hermann et al. "Teaching Machines to Read and Comprehend". In: (2015). URL: `https://arxiv.org/pdf/1506.03340.pdf`.

[3] Forrest N. Iandola et al. *SqueezeBERT: What can computer vision teach NLP about efficient neural networks?* 2020. arXiv: `2006.11316 [cs.CL]`.

[4] *InferSent.* `https://github.com/facebookresearch/InferSent`. Accessed: 2021-05-06.

[5] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: (2019). URL: `https://arxiv.org/abs/1907.11692`.

[6] Microsoft Research Asia Natural Language Computing Group. "R-NET: MACHINE READING COMPREHENSION WITH SELF-MATCHING NETWORKS". In: (2017). URL: `https://www.microsoft.com/en-us/research/wp-content/uploads/2017/05/r-net.pdf`.

[7] P. Rajpurkar. *The Stanford Question Answering Dataset.* `https://rajpurkar.github.io/mlx/qa-and-squad/`. Accessed: 2021-05-06.

[8] P. Rajpurkar et al. "Squad: 100,000+ questions for machine comprehension of text". In: (2016). URL: `https://arxiv.org/abs/1606.05250`.

[9] M. Richardson, C. J. Burges, and E. Renshaw. *Mctest: A challenge dataset for the open-domain machine comprehension of text.* Empirical Methods in Natural Language Processing (EMNLP), pages 193â 203. Accessed: 2021-05-06. 2013.

[10] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: (2020). URL: `https://arxiv.org/pdf/1910.01108.pdf`.

[11] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. "Retrospective Reader for Machine Reading Comprehension". In: (2020). URL: `https://arxiv.org/pdf/2001.09694.pdf`.

Appendix A: Example of SQuAD2.0 Dataset in JSON

```
{'context': 'The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were
the people who in the 10th and 11th centuries gave their name to Normandy, a region
in France. They were descended from Norse ("Norman" comes from "Norseman") raiders
and pirates from Denmark, Iceland and Norway who, under their leader Rollo, agreed to
swear fealty to King Charles III of West Francia. Through generations of assimilation
and mixing with the native Frankish and Roman-Gaulish populations, their descendants
would gradually merge with the Carolingian-based cultures of West Francia. The distinct
cultural and ethnic identity of the Normans emerged initially in the first half of the
10th century, and it continued to evolve over the succeeding centuries.',

 'qas': [{'answers': [{'answer_start': 159, 'text': 'France'},
    {'answer_start': 159, 'text': 'France'},
    {'answer_start': 159, 'text': 'France'},
    {'answer_start': 159, 'text': 'France'}],
   'id': '56ddde6b9a695914005b9628',
   'is_impossible': False,
   'question': 'In what country is Normandy located?'},

  {'answers': [{'answer_start': 94, 'text': '10th and 11th centuries'},
    {'answer_start': 87, 'text': 'in the 10th and 11th centuries'},
    {'answer_start': 94, 'text': '10th and 11th centuries'},
    {'answer_start': 94, 'text': '10th and 11th centuries'}],
   'id': '56ddde6b9a695914005b9629',
   'is_impossible': False,
   'question': 'When were the Normans in Normandy?'},

  {'answers': [{'answer_start': 256, 'text': 'Denmark, Iceland and Norway'},
    {'answer_start': 256, 'text': 'Denmark, Iceland and Norway'},
    {'answer_start': 256, 'text': 'Denmark, Iceland and Norway'},
    {'answer_start': 256, 'text': 'Denmark, Iceland and Norway'}],
   'id': '56ddde6b9a695914005b962a',
   'is_impossible': False,
   'question': 'From which countries did the Norse originate?'}]}
```