Neural Question Answering on SQuAD 2.0

Stanford CS224N {Default IID SQuAD} Project

Yiren Zhou
Department of Computer Science
Stanford University
zhouyr@stanford.edu

Abstract

This work aims to investigate innovative designs of model architectures that can help boost performance on the SQuAD 2.0 dataset, without using pre-trained language models. Since around half of the questions are unanswerable, it is important for the model to tactfully abstain from answering. The main contribution is a self-implemented QANet architecture with extensions on the embedding layer and the output layer. By using unified encoding for the context and question before feeding into context-query attention and employing threshold-based answer verification during testing, the model achieves stronger out-of-sample performance than the original QANet baseline. With a novel debiased ensemble method, the model achieves an EM score of 67.68 and F1 score of 70.53 on the test leaderboard for the IID SQuAD track.

1 Key Information to include

- Mentor: Elaine Sui
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Machine reading comprehension (MRC) and question answering (Q&A) has gained a lot of research attention over these years because of its theoretical importance and wide range of practical applications. Typically, a reading comprehension task involves letting the machine understand a given context paragraph, and outputting a span of the paragraph as the answer of an input query. This requires the machine to encode the paragraph and question into neural representations, and model the probability of every possible answer candidates. Besides, there are questions that are unanswerable based on the given context. An unanswerable example with a plausible answer is shown as follows:

Article : Endangered Species Act [1]
Context: "...Other legislation followed, including the Migratory Bird Conservation Act of 1929, a 1937 treaty prohibiting the hunting of right and gray whales, and the Bald Eagle Protection Act of 1940. These later laws had a low cost to society-the species were relatively rare-and little opposition was raised."
Question: "Which laws faced significant opposition?"
Answer: N/A
Plausible Answer: later laws

Since the advent of BERT [2], natural language research has shifted to a new paradigm that heavily utilizes pre-trained contextual embeddings trained by external data source. Among the models that do not use pre-trained contextual embeddings, QANet achieves state-of-the-art performance on SQuAD 1.1 dataset in 2018. QANet is an end-to-end Q&A model whose encoder only consists convolution and self-attention, without recurrent components [3]. On the other hand, SQuAD 2.0 was published later with around half of the questions unanswerable, which raises challenges on predicting the answer using previous Q&A models like QANet. Our empirical studies show that a base QANet model frequently gives plausible answer to unanswerable questions. These facts motivates us to make the following extensions to the QANet architecture to better adapt it to SQuAD 2.0 dataset:

1. We implemented the unified embedding layer for the concatenated context and question, with additional segment embedding to distinguish context and question segments.
2. We utilized threshold-based answer verification (TAV) to make the model tactfully abstain from answering when the predicted no-answer probability is high.
3. We tested the Unified QANet Model + TAV on the SQuAD 2.0 dataset and throughly evaluated its performance and limitations.

Models with various hyper-parameter configurations were trained using the training data and evaluated on the development set. We foundd that the default depth for encoder blocks in the original QANet paper is already close to optimal. The unified encoding layer only slightly improves the performance, yet the threshold-based verification significantly boosts the performance metrics, especially the exact match (EM) score. We attribute this improvement to successfully avoid providing plausible answers to some unanswerable questions. Finally, we proposed a novel ensemble method which can debias the predicted length distribution and further improve performance on test set.

3 Related Work

Recurrent model that processes sequential inputs was widely used for natural language processing tasks. For MRC tasks, it is usually combined with attention mechanism to deal with interactions between the context and question. The Bidirectional Attention Flow (BiDAF) model [4] utilizes two-way attention to encode the context into a question-aware neural representation, which achieved strong results on SQuAD dataset [5].

On the other hand, transformer models later dominate natural language processing tasks through its structural efficiency and strong performance. Inspired by the transformer architecture, QANet is invented as an end-to-end model whose encoder only consists depthwise-separable convolution and self-attention [3]. QANet achieves state-of-the-art performance on SQuAD 1.1 dataset in 2018.

The recent progress on the machine reading comprehension task without using pre-trained models mainly focuses on building verifiers that can successfully detect those question whose answer is not available from the given paragraph. [6] proposed a read-then-verify system, which leverages an answer verifier to decide whether the predicted answer is entailed by the input snippets. [7] introduced a retrospective reader (Retro-Reader) that integrates two stages of reading and verification strategies, ending with a threshold-based verification heuristic. Meanwhile, there are following research show that apart from the benefits of pre-training, performance can also be improved by using a unified encoding and matching network architecture similar to BERT transformer [8].

In this work, we take these ideas as inspiration to improve the architecture of QANet. The goal is to improve the QANet architecture to better adapt it to SQuAD 2.0. According to [8], we designed a unified variation of the embedding layer for QANet. Inspired by [6] and [7], we experimented two variants of the answer verification strategies to improve the model's decision-making before outputting the predicted result.

4 Approach

The reading comprehension task in SQuAD is defined as follows [5]: Given a context paragraph with n words $C = \{c_1, c_2, ..., c_n\}$ and the question sentence with m words $Q = \{q_1, q_2, ..., q_m\}$, output a span $S = \{c_i, c_{i+1}, ..., c_j\}$ from the original paragraph C using the model. For SQuAD dataset, the loss function is the averaged cross entropy between the predicted softmax distribution of the starting and ending position of the span and the reality (s, e):

$$L^{span} = -\frac{1}{N}\sum_{i=1}^{N}\left[\log(p^s_{y_i^s}) + \log(p^e_{y_i^e})\right]$$

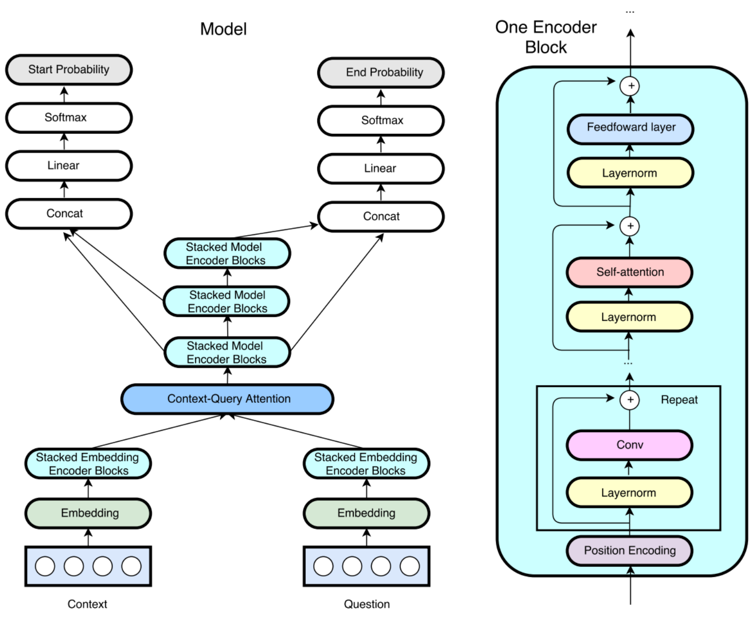

Figure 1: The QANet architecture. Source: [3]

where y_i^s and y_i^e are respectively the groundtruth starting and ending position of example i. During inference, the optimal $(\hat{y}_i^s, \hat{y}_i^e)$ which maximizes the $p^s_{\hat{y}_i^s} p^e_{\hat{y}_i^e}$ with $\hat{y}_i^s \leq \hat{y}_i^e$ and $\hat{y}_i^e - \hat{y}_i^s + 1 \leq L_{max}$ will be used as the predicted span for example i. L_{max} is set to 15 to avoid predicting long answers for stable inference.

4.1 BiDAF Baseline

BiDAF Baseline The BiDAF model provided by CS224N project starter code is used as the baseline. The architecture of BiDAF is omitted for brevity and can be find in [4]. Besides the original model, we extended BiDAF by incorporating character-level embedding using 2D-convolution and max-pooling. This character embedding is shared with the later QANet implementation and will be detailed shortly.

4.2 QANet[1]

The QANet model consists of five layers, which are introduced below. See Figure 1. We use h to indicate the hidden size variable of the model.

Input embedding layer The raw text of contexts and questions were first processed into token indexes that can query for word and character embeddings. We use 300-D pre-trained GloVe word vectors[9] for the word embedding, followed by a linear projection of $\mathbb{R}^{h\times 300}$. For the charactor embedding, each character has 64-D vector representation and every word is padded or truncated to a length of 16. A single word's character representation is of dimension $\mathbb{R}^{16\times 64}$. We used a 2D convolution with dimension $(64, h)$ of size 5 across the sequence length, and then max-pool across the characters to get a vector representation of the word. Then they were concatenated together and resized by a linear projection of dimension $\mathbb{R}^{h\times 2h}$. Finally, a two-layer highway network was applied to the hidden representation before generating the output.

Embedding & model encoder block The embedding encoder block and model encoder block share the same structure: It begins with a sinusoidal positional encoder [10]. Then there are several repeated depthwise-separable convolution layers, followed by a multi-head attention and a feed forward layer. Layer normalization was applied before each of the above components, and residual connection was

[1]The positional encoding is adapted from https://github.com/BangLiu/QANet-PyTorch. The QANet implementation was done from scratch, besides referring to online tutorial and threads for debugging purposes.

employed in every sub-block to avoid vanishing gradient problem due to the stacking depth. Similar to [3], we used 1 embedding encoder block, which contains 4 convolution layers with kernel size 7, before feeding into the context-query attention. To speed up training, we use 5 stacked model encoder layers instead of 7, with 2 convolution and a kernel size of 5. The same stacked model encoder blocks, with 5 encoder layers, are stacked 3 times with sharing weight to generate the span matching representation after the context-query attention. Due to GPU memory constraint, we set all the number of multi-head attention to 4 instead of 8.

Context-query attention layer QANet use a standard context-to-query attention constructed by similarities between each pair of context and query words [3]. They find that the additional query-to-context attention from BiDAF can provide little benefits. Since our goal is to improve QANet, we directly use the bi-directional attention from [4], hoping it can have synergy with other later modifications of the QANet architecture.

Output layer QANet adopts the strategy of [4] to predict the probability of each position being the start or end of an answer span. The hidden vectors of different stacked model encoder blocks M_0, M_1, M_2 are used as the input. With two linear projection $W_1, W_2 \in \mathbb{R}^{h \times 2h}$, the probability of starting and ending position of the answer span are predicted using softmax layer:

$$p^s = \text{softmax}\left(W_1[M_0; M_1]\right), \quad p^e = \text{softmax}\left(W_2[M_0; M_2]\right)$$

The predicted probability is then used to compute the pre-defined negative log likelihood loss.

4.3 Extensions

Unified encoding (Unified QANet) Inspired by [8], we implemented a unified embedding layer, which encodes the concatenated question and context representation. It performs the same operations as the QANet input embedding layer, with an additional trainable segment embedding to indicate whether the word belongs to context or passage segment. For a single word representation x_j, the final output is the sum of the original output of highway network and the segment embedding:

$$x_j = \text{highway}(W[x_j^w; x_j^c]) + x_{s_j}, \quad x_{s_j} \in \mathbb{R}^h, s_j \in \{0, 1\}$$

where $W \in \mathbb{R}^{h \times 2h}$ is the previous linear projection for concatenated word and character embedding. $s_j = 0$ indicates context words and $s_j = 1$ indicates passage words. Using this embedding, the self-attention weights can be automatically learned across the two segments for question-to-passage or passage-to-question matching.

Threshold-based answerable verification (TAV) From the original QANet paper, $c_0 = \text{OOV}$ (Out of Vocabulary) is inserted at the beginning of each paragraph as a start placeholder, and span $S = \{c_0\}$ will indicate a no-answer output. Among all the valid starting and ending position pairs, $(0, 0)$, which indicates no answer, is only one of these $l(l+1)/2$ possible cases (l is the sequence length of the context). This makes QANet frequently give plausible answers to unanswerable queries.

Therefore, two different verification methods were implemented:

1. (**TAV1**) We extended the final output layer of QANet with an additional binary classifier branch to predict answerability. The extended model will output a tuple $(\hat{y}_i^s, \hat{y}_i^e, \hat{p}_i^{na})$ for example i, where the last entry is the predicted no-answer probability.

$$p^s = \text{softmax}\left(W_1[M_0; M_1]\right), \quad p^e = \text{softmax}\left(W_2[M_0; M_2]\right)$$

$$p^{na} = \text{Sigmoid}\left(W_3[\text{pool}([M_0; M_1], p^s); \text{pool}([M_0; M_2], p^e)]\right)$$

   where $W_3 \in \mathbb{R}^{1 \times 2h}$, the pooling indicated the sum of the hidden vectors across context/question length weighted by the predicted probability vector p^e and p^s, respectively. The the loss function is extended with the binary cross entropy of classification:

$$L = L^{span} - \frac{1}{N} \sum_{i=1}^{N} [y_i^{na} \log(p_i^{na}) + (1 - y_i^{na}) \log(1 - p_i^{na})]$$

2. (**TAV2**) Alternatively, we can use a heuristic strategy to decide whether a question is answerable according to the predicted start and end probability [7]. Given the output

probability vector p^s and p^e, we can calculate the no-answer probability p^{na} and has-answer probability p^{has}:

$$p^{na} = p_0^s p_0^e; \;\; p^{has} = \max_{0<i\leq j\leq n} \left(p_i^s p_j^e\right), \; j - i + 1 \leq l_{max}$$

$$\text{score}^{na} = p^{na} - p^{has}$$

For both strategies, an answerable threshold δ is determined by the development set performance. If the final probability/score is above the threshold δ, the original answer will be replaced by the null string.

Model ensemble We use a majority voting approach to ensemble 7 models: 3 copies of QANet models and 4 copies of Unified QANet model with TAV2. Descending weights are assigned to models from the one with the highest development set performance to the lowest for tie breaking. After comparing the prediction answer length with the ground truth, we find that the predicted length distribution differs from the real distribution. We attribute this to the independent architecture of starting & ending position prediction. Therefore, we proposed an original weighting scheme on top of the majority voting.

Assuming there are k models $\mathcal{M}_1, ..., \mathcal{M}_k$ from a I.I.D model distribution, then the majority voting answer score is approximately proportional to predicted density. Therefore, we define the new length-debiased score by:

$$\text{score}_A = \sum_{i=1}^{k} \mathbb{I}\{\mathcal{M}_i(C, Q) = A\}$$

$$\text{score}_A^{debiased} = \text{score}_A \times \frac{P_l(l = l_A)}{P_l^{\mathcal{M}}(l = l_A)}$$

where A, C, Q are the corresponding answer, context and question. P_l and $P_l^{\mathcal{M}}$ are the probability mass functions of ground truth answer length and predicted answer length.

5 Experiments

5.1 Data

The SQuAD 2.0 dataset contains IID (context, question, answer) triples [1]. The training data consists of 129941 examples. The dev and test set both consist around 6k examples. The raw data is pre-processed to tokens and then represented by the pre-trained GloVe [9] word vectors combining with randomly initialized trainable character embeddings.

5.2 Evaluation method

The performance are evaluated using three metrics: **Exact Match** (**EM**), **F1** and **AvNA**.

- **Exact Match (EM)** is a binary measure indicating whether the output matches the ground truth answer exactly.
- **F1** is the harmonic mean of precision and recall.
- **AvNA** is the precision of the model's prediction on whether a question is answerable.

The averaged scores over the entire development/test set will be used as the reported scores.

5.3 Experimental details

BiDAF We run the BiDAF model with character embeddings using the same configuration as in the starter code for 30 epochs. Both took around 3.5 hours on a GeForce RTX 2080 GPU.

QANet The QANet is trained using AdamDelta, with a hidden size of 128, batch size of 24, EMA decay of 0.999, dropout rate of 0.1, multi-head attention heads number of 4, and a learning rate of 0.5 for 30 epochs. The layer configuration are the same as [3], except that the query-to-context attention was added to the context-query attention layer, and the number of stacked model encoder blocks is

Table 1: Model performance comparison on the SQuAD 2.0 development set

Model	EM	F1	AvNA
BiDAF (Baseline)	58.01	61.26	68.27
BiDAF + Character Embeddings	60.36	63.41	69.37
QANet	66.27	69.45	75.32
UnifiedQANet	66.32	69.72	**76.02**
UnifiedQANet (1 more embedding encoder)	66.09	69.49	74.79
UnifiedQANet (1 more model encoder)	65.94	69.38	75.47
UnifiedQANet + TAV1	65.48	68.65	74.09
UnifiedQANet + TAV2	**67.13**	**69.97**	75.52
Ensemble	69.50	72.35	N/A
Ensemble + Length debiasing	**70.32**	**73.10**	N/A

decreased from 7 to 5 for training speedup. We train the model for 30 epochs on a GeForce RTX 2080 GPU in around 14 hours.

UnifiedQANet The Unified QANet variants are trained using the same optimization configuration as QANet. We trained the Unified QANet with one more embedding encoder layer (2 in total) and with one more model encoder layer (6 in total). After observing they achieves lower development performance metrics, we decide not to explore layer depth configurations further. We then trained the model with TAV1 with the additional branch for classification. After tuning on the development set, the optimized no-answer probability threshold for TAV1 is set to be $\delta_1 = 0.4$. We then applied TAV2 to the original Unified QANet since it only requires changes in model inference phase. The optimized no-answer score threshold for TAV2 is $\delta_2 = 0.1$.

5.4 Results

The experiment results are shown in Table 1. For the BiDAF baseline, incorporating character embedding improves the performance. We also observe that our implementation of QANet greatly outperforms both the BiDAF baseline and the one with char embeddings in terms of both EM, F1 and AvNA scores.

The Unified QANet performed slightly better than the QANet, but the difference is not significant enough. Since our QANet only consists of 5 model encoder layers, instead of 7 in the original paper, we tried two upsized variants, one with 1 more embedding encoder and the other with 1 more model encoder. However, they did not achieve performance gain, which suggests that for our implementation and dropout configuration, the model complexity is already around the sweet spot.

The Unified QANet with TAV2 achieved the highest performance. The result is expected since a perfect score for a specific example can be achieved by successfully predicting unanswerability. By slightly shrinking towards not giving an answer, the model can benefits from the strategic trade-off between answering and giving up. On the other hand, the Unified QANet with TAV1 did not perform well. There are two possible reasons: 1. The classification branch may not be well designed. 2. The original cost function is extended with an additional binary cross entropy term. This can hurt the model's learning towards the ground truth span, since now only half of the gradient is focusing on the span loss function during backpropagation.

Finally, we see that the two ensemble of the seven models both achieve significant improvement in EM and F1 score on the development set. The best ensemble achieves an EM score of 67.68 and F1 score of 70.53 on the test leaderboard. The performance difference in development and test sets indicates there are some minor overfitting during our hyperparameter optimization, which is based on development score. It is also possible that there exists distribution shifts between the dev and test set.

6 Analysis

Predicted answer length distribution One of the deficiency of QANet span matching is that the starting position and ending position are predicted independently. Even with architecture modifica-

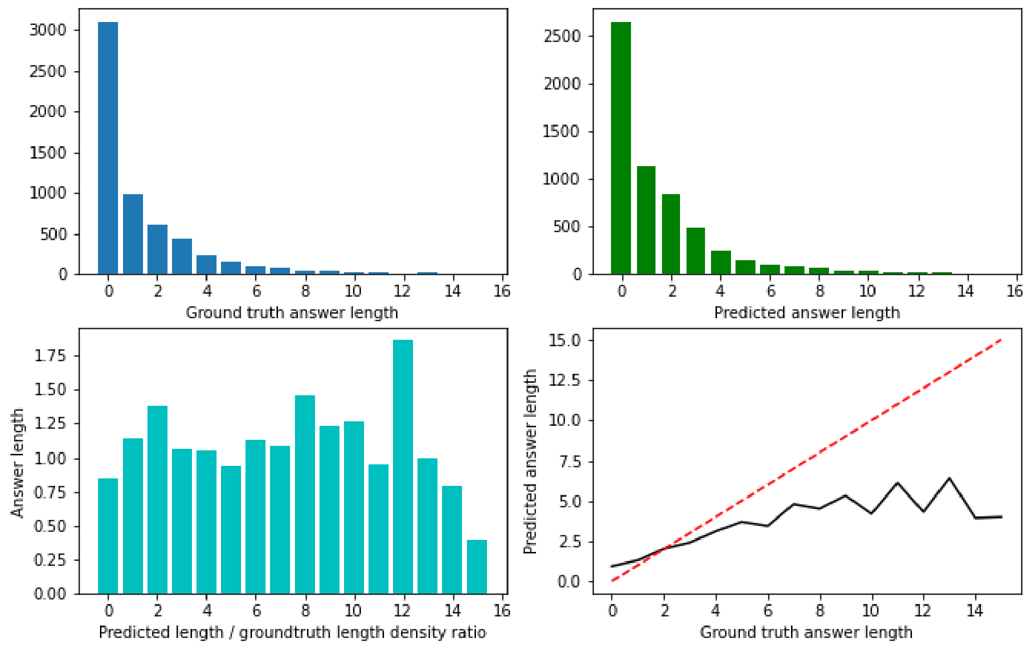

Figure 2: The QANet architecture. Source: [3]

Table 2: Performance breakdown of the UnifedQANet on different types of questions (dev)

	What	Who	How	When	Where	Which	Why	Other
EM	66.32	67.88	64.95	75.00	62.07	68.49	59.52	63.67
F1	69.86	70.32	69.32	75.85	66.60	74.41	67.60	67.61
AvNA	75.96	74.50	74.85	81.36	75.00	82.87	77.38	75.74

tions, it is still not guaranteed that the model can fully take the answer length into account when choosing the optimal one among all the solution candidates. The upper two plots in figure 2 show the histogram of ground truth answer length (only contains those length within l_{max}) and predicted answer length. The lower left is the density ratio of the predicted length and ground truth length. We can see that the model is predicting less NULL answer and promoting answers with length from 2 to 12. The bottom right shows that the model tends to over estimate the answer length when the answer is less than or equal to 1 word. Therefore, we use the weight from bottom left to debias the answer length distribution during ensemble. This provides 0.82 increase of EM and 0.75 increase of F1 on the SQuAD development set, besides the majority voting ensemble.

Performance breakdown & Error analysis Table 2 shows the performance breakdown for different type of questions. To better understand when the model makes mistakes, we took 3 examples from the development set for further analysis. See Table 3 for the concrete examples.

In example 1, the model fails to understand the difference between "modern" and "medieval", and outputs a plausible answer. This is sensible since in GloVE these two words should be quite close in terms of distance, though they carry different meanings. A possible remedy is to use a higher-dimension word vector or pre-trained contextual embedding which understand the intricate difference between these two words. Example 2 is a "why" question, which the model is not good at answering. The answer hides in the previous sentence but the subject is implicit, so the model fails to deduce it from later context. For example 3, the model fails to match the question when the correct answer. The prepositional phrase "Immediately after Decision Time" is the obvious answer. However, the model does not represent and understand this phrase well enough, and output the "45 min" since it is more related with time sensitive queries.

Table 3: Error examples

Context paragraph	Question	Ground truth	Predicted
1. ...The Norman dynasty had a major political, cultural and military impact on medieval Europe and even the Near East. The Normans were famed for their martial spirit and eventually for their Christian piety, becoming exponents...	What type of major impact did the Norman dynasty have on modern Europe ?	N/A	political, cultural and military impact
2. ...The earlier they surrendered to the Mongols, the higher they were placed, the more the held out, the lower they were ranked. The Northern Chinese were ranked higher and Southern Chinese were ranked lower because southern China withstood and fought to the last before caving in...	Why were Northern Chinese ranked higher ?	they surrendered	N/A
3. Immediately after Decision Time a " Members Debate " is held , which lasts for 45 minutes. Members Business is a debate on a motion proposed by an –OOV– who is not a Scottish minister...	When is the Members Debate held ?	Immediately after Decision Time	45 minutes

Table 4: Confusion matrix and rates on detecting unanswerability (no answer is Positive)

Model	#TP	#TN	#FP	#FN	TPR	TNR	AvNA
UnifiedQANet	2162	2362	486	941	82.65%	**71.51%**	**76.02**
UnifiedQANet + (TAV2)	2257	2237	611	846	**78.70%**	72.56%	75.52

The three examples shows two deficiencies of the model: 1. The embedding from the encoder layer is not comprehensive enough to carry complicated information for context-query matching. 2. The model is unsure about answerablity when there are required logical deduction for the question.

AvNA model comparison Table 4 shows the answerability prediction comparison between two models. From Table 1 we know the second one performs better. We can see that thresholding is basically a trade-off between True Positive Rate (TPR) and False Positive Rate (TNR). It is inituitive why it can boost performance: A perfect score for this example can be achieved by successfully predicting that a question is unanswerable, but nothing is guaranteed for the opposite. For real world application, it is also beneficial for the NLP model to abstain from answering and call for expert intervention instead of providing plausible answers by doing this trade-off.

7 Conclusion

In this project, we deal with the machine reading comprehension task. By implementing and extending QANet, we achieved strong out-of-sample results on the SQuAD 2.0 dataset. Using a unified encoder layer and tuning the layer depth parameters does not seem to improve the model significantly. On the other hand, we are able to significantly improve performance by using threshold-based verification and length-debiased ensemble, which are both extensions related with answerability. We find that our model usually make mistakes when the ground truth answer is long or when plausible answer exists. Therefore, it remains promising to see whether pre-trained language models with deeper representation ability can further boost the performance. Finally, it will be interesting to explore more advanced answer verification techniques. For example, combining sketchy reading and intensive reading, which uses additional neural architectures to learn the interaction between question and the truncated context that contains the potential answer obtained by sketchy reading.

References

[1] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*,

2018.

[3] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. In *International Conference on Learning Representations*, 2018.

[4] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[6] Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Dongsheng Li. Read+ verify: Machine reading comprehension with unanswerable questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6529–6537, 2019.

[7] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. Retrospective reader for machine reading comprehension. *arXiv preprint arXiv:2001.09694*, 1:1–9, 2020.

[8] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Lei Cui, Songhao Piao, and Ming Zhou. Inspecting unification of encoding and matching with transformer: A case study of machine reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 14–18, 2019.

[9] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.