Question Answering on SQuAD 2.0 Dataset

Yuanjun Li
Department of Energy Resources Engineering
Stanford University
yuanjun@stanford.edu

Yuzhu Zhang
Department of Electrical Engineering
Stanford University
arielzyz@stanford.edu

Abstract

The complex interactions between context and question makes it difficult for machines to perform good on question answering(QA) tasks. In this study, we explore the performance difference of the task-specific Bi-Directional Attention Flow model(BiDAF) and the pretrained BERT (Bidirectional Encoder Representations from Transformers) QA model on SQuAD 2.0. We propose three self-designed model structures built on the BERT embeddings. The baseline BiDAF model achieves 60.5% F1 and 57.4% EM on the validation set. Our modified fine-tuned model on BERT achieves F1 and EM scores up to 76.6% and 73.6%. The best performed BERT QA + Classifier ensemble model further improves the F1 and EM scores to 78.1% and 75.3%.

1 Introduction

From online searching to information retrieval, question answering is becoming ubiquitous and being extensively applied in our daily life. Reading comprehension is a popular instance of Question Answering tasks, where the system tries to provide the correct answer to the query with a given context paragraph. In 2016, Rajpurkar et al.[1] released the the Stanford Question Answering Dataset(SQuAD 1.0) which consists of 100K question-answer pairs each with a given context paragraph and it soon becomes a standard test for the reading comprehension task with public leaderboard available. In 2018, the team further released SQuAD 2.0, which contains over 50,000 unanswerable questions that post a much harder requirement on model development. At the same year, large-scale pre-trained language modes such as OpenAI GPT [2] and BERT [3] have achieved great performance on multiple language tasks using generic model architectures. For BERT, it is pre-trained with two auxiliary tasks(the Mask Language Model task and the Next Sentence Prediction task) with large corpus to encourage the bi-directional prediction on text as well as sentence-level understanding, Since many important down-stream tasks such as Question answering (QA) and Natural Language Inference (NLI) are based on understanding the relationship between pair of sentences, BERT can perform well when fine-tuned on these downstream specific tasks without customized network architectures.

2 Related work

There has been a lot of work on building deep learning systems for the SQuAD Dataset, as can be seen in the leaderboard (`https://rajpurkar.github.io/SQuAD-explorer/`) . On the SQuAD 2.0 leaderboard, all the top 10 model performers are developed based on BERT and utilize extra components such as attention mechanism [4] or synthetic self-training. Its promising performance motivates us to apply BERT as the basis of our developed models. Prior to the existence of BERT, Bi-Directional Attention Flow [5] introduces a novel concept of combining both character and word level representations using a highway network [6] to improve performance of Reading Comprehension Systems. To address the non-answerable questions, a Read + Verify scheme is proposed by [7], which utilizes a neural reader to extract candidate answers and introduces an answer verifier to decide whether the predicted answer is entailed by the input snippets. We also take inspiration from their work in our model.

3 Approach

3.1 Baseline Model (BiDAF)

Bi-Directional Attention Flow (BiDAF) network is composed of six hierarchical stages, which perform the context at different levels of complexity. Besides, this bi-directional attention flow technique generates a query-aware context representation without summarizing in early stage [5].

In this work, BiDAF is used as baseline to compare with BERT+X structures. This implementation in our study does not include a character-level embedding layer compared with the original work. The existing five layers are in following sequences: Embedding Layer, Encoder Layer, Attention Layer, Modeling Layer and Output Layer. (Fig.1).

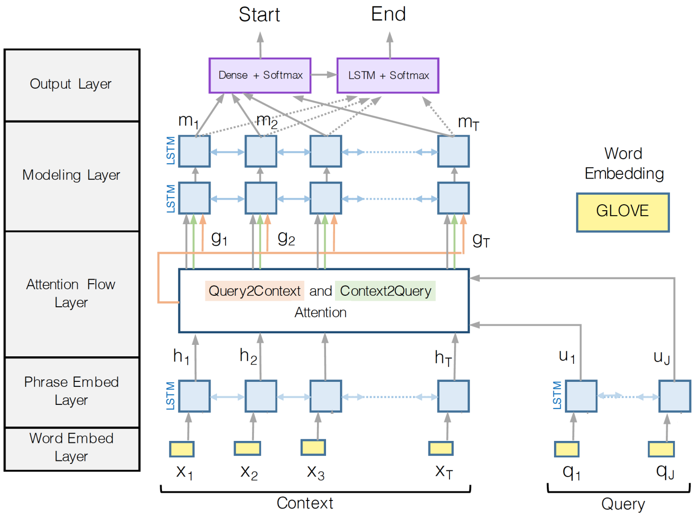

Figure 1: BiDAF (Figure adapted from [5])

3.2 BERT Fine-tuned Model

In this section, we proposed three extension models based on the BERT fine-tuned QA model. The input representation of BERT applies the WordPiece embeddings and positional embeddings with a multi-layer bidirectional Transformer encoder.

3.2.1 Fine-tuned BERT QA Model on SQuAD 2.0

The first experimented method is the BERT fine-tuned QA model for SQuAD dataset in (`https://github.com/huggingface/pytorch-pretrained-BERT/`). The input question and paragraph pairs are represented as a single packed sequence, with the question using the A embedding and the paragraph using the B embedding, as in Figure 5. The pairs are separated by a special token ([SEP]). The predicted answer span are learned with a start vector $S \in R^h$ and an end vector $E \in R^h$. In the original approach, a fully connected linear layer $W \in R^{2\times h}$ followed by a standard Softmax procedure is applied on the final hidden vector $T_i \in R^h$ to compute the probability of word i being the start and end of the answer span. Specifically for SQuAD 2.0, The non-answerable solving is using [CLS] token as the ground truth. As a result, the predicted answer start position and end position are equal to -1.

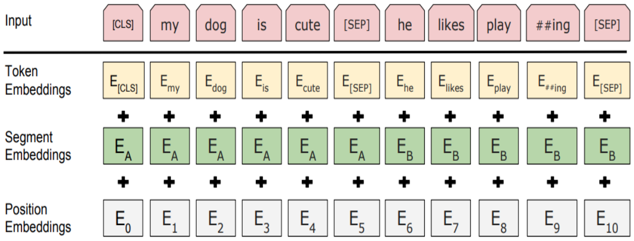
Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2: BERT input representation

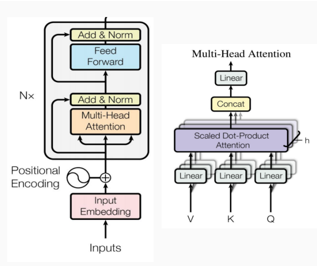

Figure 3: Transformer Structure

3.2.2 Fine-tuned BERT QA Model with modified output layer

The proposed method is a modified version of the BERT fine-tuned QA model in 3.2.1. In this phase, we aim to improve the model performance by altering the output layers on BERT. As shown in Figure 4, our final best-performed architecture consists of three main layers that are described below in detail.

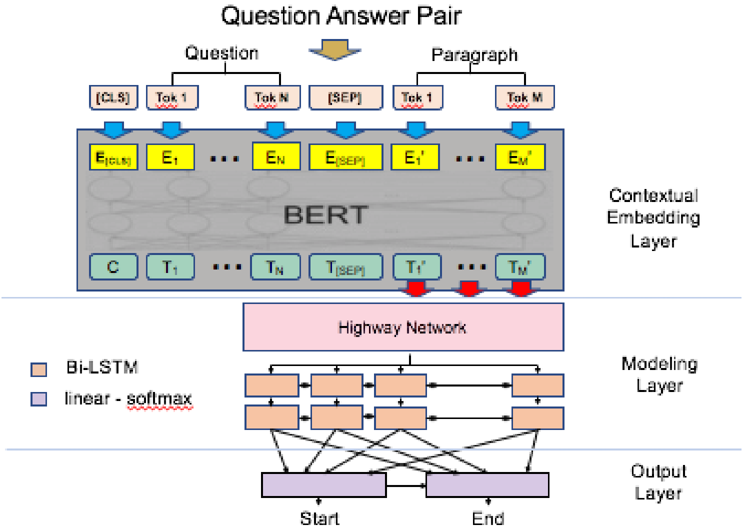

Figure 4: Modified BERT QA model

Embedding Layer
After the pre-training process of BERT, the transformer outputs the full sequence of hidden-states corresponding to the last attention block. The hidden states are the concatenated vector representation of the the context and questions $T_1, T_2 \cdots T_N \in R^h$. Therefore BERT can be viewed as the embedding layer which is fixed and are not updated during training in the model.

Highway Network
We use a two-layer Highway Network[6] to transform each hidden vector T_i. The transformation is applied twice and each time with distinct learnable parameters. ,which means we apply the above transformation twice, each time using distinct learnable parameters. By introducing the Highway Network we hope to refine the BERT embedded representation with the gating mechanism and enable to the LSTM structure with bigger step transition depth to optimize the training procedures. [8]

$$\begin{aligned} x_p &= \text{ReLU}(W_p T_i + b_p) \\ x_{\text{gate}} &= \sigma(W_{\text{gate}} T_i + b_{gate}) \\ x_{\text{highway}} &= x_{\text{gate}} \odot x_p + (1 - x_{\text{gate}}) \odot T_i \in \mathbb{R}^h \end{aligned}$$

where $W_{\text{gate}}, W_p \in R^{h \times h}$ and $b_p, b_{gate} \in R^h$

Modeling layer
Modeling layer consist of two layers of bi-directional LSTMs with hidden size h to scan the vector space representations after the highway layer. It is designed to better capture the relation between context and question. The output of the modelling layer is $M \in R^{2h \times L}$. After the modelling layer, we also perform dropout operation to avoid overfitting.

$$\{g_1, \cdots, g_L\} = \text{biLSTM}(\{x_1, \cdots, x_L\})$$
$$\{m_1, \cdots, m_L\} = \text{biLSTM}(\{g_1, \cdots, g_L\})$$

where $m_i = [\overrightarrow{m_i}; \overleftarrow{m_i}] \in R^{2h}$, for $i = 1, \cdots L$, where L is the maximum sequence length

Output Layer For the output layer, we predict the answer start and end probability distributions independently with two heads, each performing a linear down-projection followed by softmax.

$$P_{i,\text{start}} = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}, P_{i,\text{end}} = \frac{e^{E \cdot T_i}}{\sum_j e^{E \cdot T_j}}$$

3.2.3 Fine-tuned BERT QA Model with Modified Loss

Pooled representation of the firsts token
The third model is aimed at improving the model performance on non-answerable questions. We observe that the first token [CLS] is an indicator to emit logit for "no answer". Therefore, as shown in Figure **??**, we take the final hidden state for the first token [CLS] in the input, which by construction corresponds to the the special [CLS] embeddings. The vector is denoted as $Q \in R^h$. Then the predicted classC_p could be constructed with a sigmoid regression layer:

$$C_p = \text{sigmoid}(W * Q + b)$$

where $W \in R^{K \times h}, K = 2, k_i = \{0, 1\} \in R^2$, as 0 is an indicator of non-answerable questions and 1 is an indicator of answerable questions.

Modified Training Loss Function In the original scheme, the loss is computed as the sum of the cross entropy loss for start and end positions. The proposed model considers the total loss to be a weighted average of the start and end position losses and the cross entropy loss of the predicted labelsC_p compared to true labels C_r. Let s_i, e_i denote the start and end logit outputs, s_r, e_r be the true start and end positions, and CE be the cross entropy loss. Then the modified loss becomes:

$$Loss = (\alpha CE(s_i, s_r) + \alpha CE(e_i, e_r) + \beta CE(C_p, C_r))/(2 * \alpha + \beta)$$

3.2.4 Ensemble BERT QA + Classification Model

Instead of combining the QA and classification tasks as 3.2.3, in the third model, we separately fine-tuned the QA and classification model and ensemble the two models to jointly produce the answer span. In Figure 6., for the BERT classifier on the right, it takes the final hidden state for the first [CLS] token, which by construction corresponds to the special [CLS] word embedding. We denote this vector as $C \in R^H$. The vecor is then input into a classification layer $W \in R^{K \times H}$ with K being the number of labels. As the same with 3.2.3, $K = 2$.

Then the label probabilities can be computed as

$$P = \text{softmax}(CW^T)$$

If the predicted lable is 0 as impossible answers, the output answer span from the QA model would be rejected.

4 Experiments

4.1 Data

Stanford Question Answering Dataset (SQuAD) is one of the most widely-used reading comprehension benchmarks. In this project, SQuAD 2.0 is applied. Compared with SQuAD 1.1, SQuAD 2.0

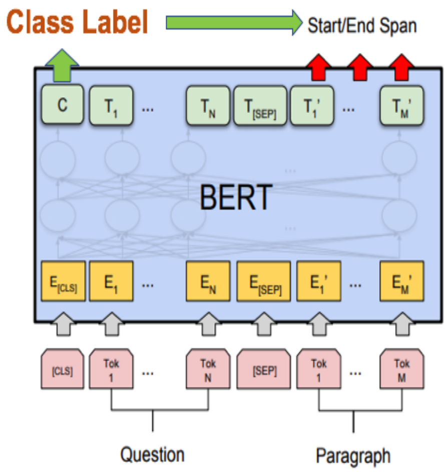

Figure 5: Single BERT QA + Classifier

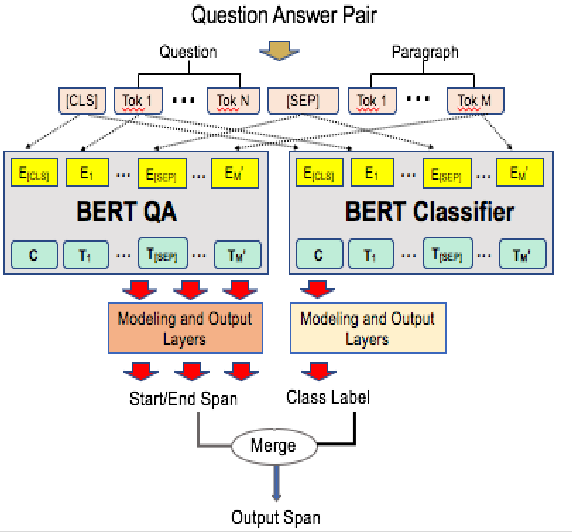

Figure 6: Ensemble BERT QA + Classification

introduces over 50,000 unanswerable questions written by crowdworkers similar to the answerable ones.[9] This requires the system not only to answer the question, but also can differentiate when there exists no answer to the question. The dataset is partitioned into training, development and test sets as provided. Among the 127803 answerable questions in the training set, we plotted the distribution of paragraph context length, question length and answer length in the below figures. It shows that small answer lengths are dominating but also with a long tail.

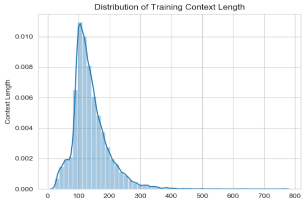

Figure 7: Context length

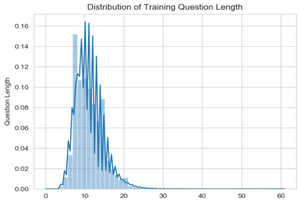

Figure 8: Question Length

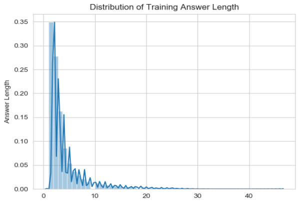

Figure 9: Answers Length

4.2 Evaluation method

Two evaluation metrics are employed: Exact Match (EM) score and F1 score. EM is a binary measurement of whether the percentage of output from a system exactly matches the ground truth answer(the proportion of questions that are answered in exact same words as the ground truth). F1 score is a harmonic mean of precision an recall. For each question, precision is calculated as the number of correctly predicted words divided by the total words in the predicted answer. Recall is the number of correctly predicted words divided by the number of words in the ground truth answer. The F1 score is averaged among questions. For detailed analysis, the scores are broken down to the answerable questions and non-answerables in our models.

4.3 Experimental details

Model configurations are listed as below, the highlighted parameters are the final model parameter for best performance:

Models	Tunable Parameters				Training Config	
	Batch Size	Learning Rate	Max Seq. Length	Number of epochs	Time per epoch	Machine
Baseline(BiDAF)	64	0.5	-	30	0.3 hours	NV6
Fine-tuned BERT QA Model	6	3e-5	384	**2**,3	4 hours	NV6
BERT QA with Modified Output Layer	6	3e-5	384,**512**	2,**3**	4 hours	NV6
BERT QA + Classifier(single)	6	3e-5	384,**512**	2	4 hours	NV12
BERT QA + Classifier(ensemble)	**6**	**2e-5,** 3e-5	256,**384**,512	**2**,3	6 hours	NV12

Due to the constriants we have, for all the BERT-based models, BERT base uncased pretrained model is applied: it contains 12 transformer blocks, 768 hidden layers, 12 self-attention layers and in total of 110M parameters.

4.4 Results

4.4.1 Final Model Results

The experimental results on the validation set for our current findings are listed in the below table: The scores are broken down to the total data, the data with answers and the data that has no answer.

Models	Total Questions		Answerable Ques.		Non-answerables	
	F1	EM	F1	EM	F1	EM
Baseline(BiDAF)	60.5%	57.4%	-	-	-	-
Fine-tuned BERT QA Model	74.5%	72.2%	70.0%	65.2%	78.6%	78.6%
BERT QA with Modified Output Layer	76.6%	73.5%	81.3%	74.6%	72.4%	72.4%
BERT QA + Classifier(single)	75.3%	73.1%	74.7%	70.0%	76.0%	76.0%
BERT QA + Classifier(ensemble)	**78.1%**	**75.3%**	**77.1%**	**71.2%**	**79.0%**	**79.0%**

Our submission on the test PCE leaderboard achieves(**EM: 74.505%, F1: 77.816%**)

4.4.2 Other Performance comparisons

For the fine-tuned BERT QA model with modifications on the output layers, we have tried a lot of different layer components and parameter combinations to achieve final performance in the above table. The attempted components are listed as below:

Model	F1	EM
BERT + linear	73.8%	71.8%
BERT + GRU	76.4%	73.8%
BERT + LSTM 1-layer	76.0%	72.8%
BERT + LSTM 2-layer	76.8%	73.7%
BERT + bi-LSTM 2-layer	77.1%	74.4%
BERT + bi-LSTM 2-layer + Highway	77.8%	74.9%

Compared with single linear output layer, GRU, LSTM and bi-LSTM can both model the correlation in the embedding matrix output by BERT and boost the QA model performance. It is found out that, the bi-directional LSTM performs the best among all. It is as expected since it allows two-way interaction of the words representations. It also coincides with the bi-directional encoder inside of BERT, which aims to let words can see themselves. The highway network is also beneficial in the sense that it can compute features at different level of granularity, thereby increasing the training efficiency.

5 Analysis

5.1 Answerable vs non-answerable questions

The first interesting thing we have noticed is as we increase the number of epochs in training, the performance of the answerable questions is improved while the performance for the non-answerable questions drop hugely. By further analysis, we don't think this is an overfitting problem since the scores of answerable questions keep increasing. Besides the similar results are obtained for all dropout parameters tried. One possible solution is that, the no-answer indicator is only from the first [CLS] token, the value of attentions to the [CLS] token may be much weaker than the word-word attention. Hence the Transformer may focus less on the attention associated with the [CLS] token. .

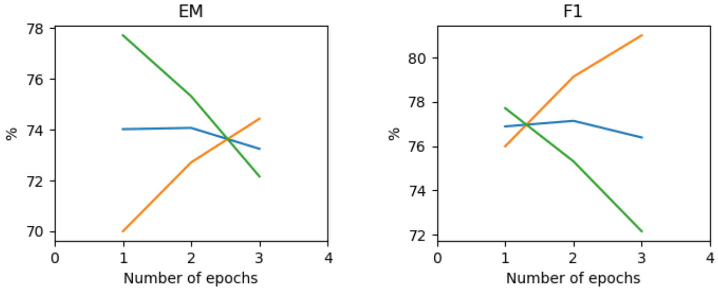

Figure 10: EM and F1 for total questions(Blue), answerables(Orange) and non-answerables(Green) change with number of epoch

5.2 Error Analysis

The error analysis is based on examining specific samples where model fails to choose the exact answer on the validation set. Three main errors are found: false positives, Improper focus on text and Subtle semantic ambiguities.

5.2.1 False positives

The model is confused about when to decide it has no answer. Rather than having confidence to output impossible label, the model tends to find a rather long span in context as the predicted answer. We have tried the two classifier models to alleviate the problem, by increasing the penalty on misclassification of no-answer labels. However, it is found that the problem still exists and is the main source of the drop in the performance of non-anwerable questions as mentioned before.

Example:

- Context: To classify the computation time (or similar resources, such as space consumption), one is interested in proving upper and lower bounds on the minimum amount of time required by the most efficient algorithm solving a given problem. The complexity of an algorithm is usually taken to be its worst-case complexity, unless specified otherwise.
- Question:How does one note classify the computation time(or similar resources)?
- Answer: []
- Predicted: "proving upper and lower bounds on the minimum amount of time required by the most efficient algorithm solving a given problem"

5.2.2 Improper focus on text

The model is focusing on the wrong region of the paragraph and hard to differentiate the antonyms in the context and questions("successor VS former") thus missing the true answer span.

Example:

- Context: The uprising occurred a decade following the death of Henry IV, a Huguenot before converting to Catholicism, who had protected Protestants through the Edict of Nantes. His successor Louis XIII, under the regency...
- Question: What King and former Huguenot looked out for the welfare of the group?
- Answer: Henry IV
- Predicted: Louis XIII

5.2.3 Subtle semantic ambiguities

Aside from the first two categories give large errors, we have also found a few cases where the model points to the correct position yet has a very minor difference with the ground truth answer. In this case, the model is doing as expected without any mistakes.

Example:

- Context: Virtually all nuclear power plants generate electricity by heating water to provide steam that drives a turbine connected to an electrical generator.
- Question: In a nuclear power plant, what is the steam turbine connected to?
- Answer: electrical generator
- Predicted: an electrical generator

5.3 Attention visualization

By doing this visualization [10], we find the attention mechanism of BERT is quite effective. For example in the fifth layer, the combinations seem to be more focused compared with layer 3. i.e. (we, have), (if, we), (keep, up) (get, angry).

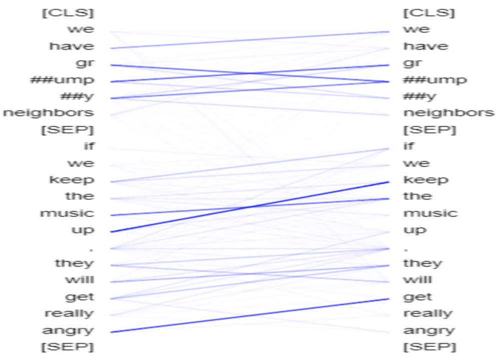

Figure 11: Layer 3 Attention

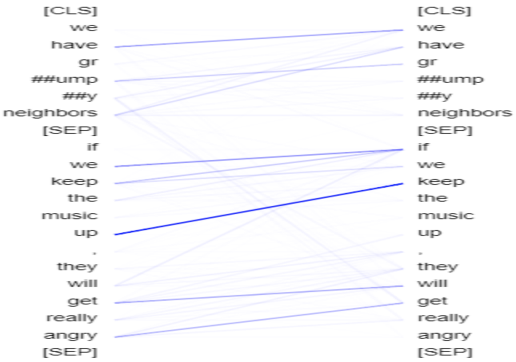

Figure 12: Layer 5 Attention

6 Conclusion

In this paper, we implement and design three variants on three different types of question-answer modes: single BERT fine-tuned model, single BERT model classified model. ensemble BERT question answer with classifier model. The experimental evaluations show that our model achieves competitive results in SQuAD 2.0. The model can perform well on predicting the correct answer locations for answerable questions and also detecting whether the question is answerable. For future development, we hope to incorporate more additional features to the BERT embedding output, such as name entities, POS tag and the question-context matching features, to further boost its performance.

7 Additional Information

We would like to thank TAs for insightful discussions and helpful suggestions provided to our project.

References

[1] Zhang Rajpurkar and et al Lopyrev. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250, 2016*, 2016.

[2] Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI, 2018.

[3] 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.

[4] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. *arXiv preprint arXiv:1607.04423*, 2016.

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

[6] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[7] Minghao Hu, Yuxing Peng, Zhen Huang, Nan Yang, Ming Zhou, et al. Read+ verify: Machine reading comprehension with unanswerable questions. *arXiv preprint arXiv:1808.05759*, 2018.

[8] Julian G. Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. Recurrent highway networks. *CoRR*, abs/1607.03474, 2016.

[9] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[10] Visualize bert attention. *https://medium.com/synapse-dev/understanding-bert-transformer-attention-isnt-all-you-need-5839ebd396db*.