Extended QANet and Application on SQuAD 2.0

Larry Jin
Stanford University
zjin@stanford.edu

Yimin Liu
Stanford University
yiminliu@stanford.edu

Su Jiang
Stanford University
sujiang@stanford.edu

Abstract

The recent purposed QANet, which constructs stacks of transformer, consisting of mainly self-attention layer and convoultional layer, to replace RNN, has drawn increasing attentions in the area of machine reading and question answering (Q&A). QANet results in significant speed-up of machine comprehension tasks without much loss in correctness. In this project, we first implement a QANet framework, then modify the framework with fewer stacking layers, also replace the 1D ConvNet with GRU. With the full implementation of QANet model, we achieved EM and F1 score of 63.15 and 66.72 on the dev set of SQuAD 2.0, respectively. An ensemble of 3 models achieved F1 of 67.60 and EM of 64.26 on the dev set and F1 of 65.41 and EM of 62.10 on the test set.

1 Introduction

The growing interest in the task of machine reading comprehension after the release of SQuAD dataset [6] has facilitated significant progresses in this research area. The current leading models often advance on two key ingredients: (1) a pre-trained contextual embedding model, or, (2) an advanced encoding model. The pre-trained contextual embedding (PCE) such as ELMo [4] and BERT [2] is currently the best performing solution on the SQuAD 2.0 leaderboard.

However, due to the sophistication of the BERT framework, achieving original improvements on BERT given the time constraint seems infeasible. Therefore, in this project, we focus on the improvement of encoding model with advanced attention components. Our basic model combines a number of tending frameworks such as Bi-directional Attention Flow [7] and QANet [11], and is tested on SQuAD 2.0 dataset.

To further understand and improve the model, several modifications have been applied to the basic model. We first add a RNN based contextual embedding layer in addition to the word-level and character-level embeddings. We then simplify the encoder blocks for both encoding and modeling layers with less stacks of ConvNet. In the end, we replace the 1D ConvNet with GRU in the encoder block.

This paper proceeds as follows: Section 2 offers the overview of the research landscape this project is in, Section 3 introduces the layout of the model architecture, Section 4 describes the implementation details and shows the results of the model, Section 5 gives thorough analysis of the model, both quantitatively and qualitatively, and Section 6 concludes the paper.

2 Related work

Comparing with improving PCE models, improvements made on encoder layers with advanced attention model allows quick perturbations of the model framework within a short period of time. Therefore, our team decides to adapt and develop upon one of such models. Our baseline model is the direct implementation of Bi-directional Attention Flow model [7], which utilizes character-level,

Preprint. Work in progress.

word-level, and contextual embeddings, and builds a query-aware context representation without early summarization.

Models that are directly relevant with this project include R-Net, QANet, and Transformer-XL. R-Net [9] combines gated attention-based recurrent networks and self-matching attention mechanism to obtain the and refine the question-aware passage representation. QANet [11] replaces the RNN-based encoder in both embedding encoder and model encoder with a Transformer [8]-based encoder built upon convolutoinal neural-networks (CNN) and self-attention, which significantly improves the speed of the model without accuracy loss. Transformer-XL [1] improves based on Transformer by introducing the concept of recurrence by reusing the representations from the history, which improves the accuracy of the model with long term dependency.

Other models intending to improve the attention mechanism are also of interest, which includes DCN and Fusion-net. Dynamic Coattention Network (DCN) [10] attends over attention outputs in addition to the two-way attention between the context and question, which enables the model to recovery from initial local maxima with incorrect solutions. Fusion-net [3] identifies an efficient attention mechanism corresponding to the new concept of "history of word", and improves the overall prediction accuracy.

3 Approach

3.1 Model overview

The baseline model we used in this project is built upon the starter code with BiDAF [7], and follows the framework of QANet proposed by Yu et al. [11], which is a convolution and self-attention based question answering (Q&A) model. The structure of QANet is composed of five layers, including an embedding layer, an embedding encoder layer, an attention layer, a model encoder layer and an output layer. The architecture is shown in Fig. 1. The over fitting problem happens when we train the standard QANet method. The possible reason is the complexity of the QANet, so we make a series of modifications to simplify the QANet method. And we also want to add some RNN layers to keep the correlation of long term.

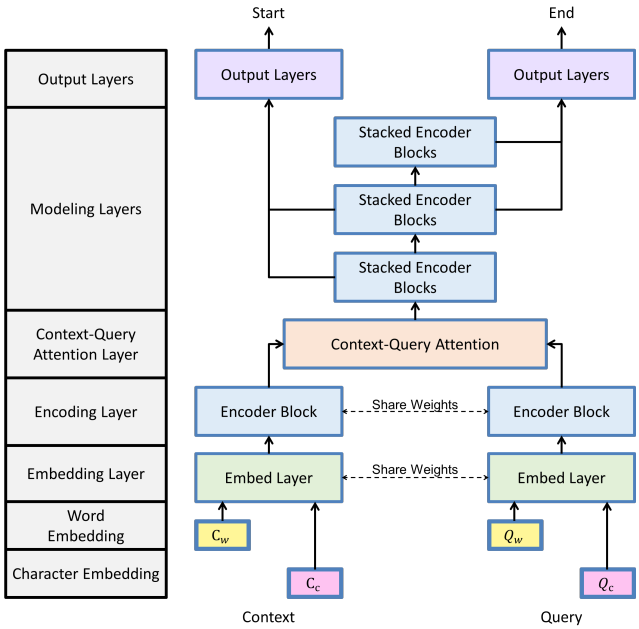


Figure 1: Baseline model architecture (closely resemble QANet).

3.2 Embedding layer

Word embedding and character embedding are used for the input embedding layer. The dimension of the word and character embedding vectors are denoted by e_{word} and e_{char}, respectively. In our case, e_{word} equals to 300 and e_{char} equals to 64. Note that in the original QANet paper, the character embedding vector dimension is $e_{\text{char}} = 200$. This is the main difference of our implementation to the original paper. The word embedding and character embedding vectors are denoted by C_w and C_c respectively for one context word, and Q_w and Q_c respectively for one query word. Then the word embedding vectors and character embedding vectors are concatenated and fed into an embedding layer, a two-layer highway network, which outputs the final embedding vector, denote by $C \in \mathbb{R}^{e_{\text{embed}}}$ for context word and $Q \in \mathbb{R}^{e_{\text{embed}}}$ for query word. Note that the embedding layer for context and query words share the same weights. In the original QANet paper, the dimension of the embedding vector is $e_{\text{embed}} = 128$. We evaluated two QANet configurations, one with $e_{\text{embed}} = 128$ and the other with $e_{\text{embed}} = 96$, as will be described more in the Results section.

3.3 Embedding encoder layer

The embedding vectors are then fed into an encoding layer. In our current implementation, the encoding layer uses one encoder block, a stack of convolution layer, self-attention layer and feed forward layer, each of these layers is proceeded by a layer-normalization layer. For the self-attention-layer, the multi-head attention mechanism defined in [8] is adopted. In the original QANet paper, the number of attention heads is 8. Besides this setting, we also considered another setting with single head attention. More details of the encoder block is described in [11]. The output from the encoding layer is denoted by $C_e \in \mathbb{R}^{e_{\text{embed}}}$ for one context word and $Q_e \in \mathbb{R}^{e_{\text{embed}}}$ for one query word. Again, the encoding layer for context words and query words share the same weights.

And in our methods, we have also tried to add RNN (LSTM) layer after word and character embedding to take account of the long-term dependencies. Further discussion will be shown in the result part.

3.4 Attention layer

The context-query attention layer of QANet uses standard attention treatments, which is referred to as dynamic coattention networks (DCN) and is described with more detail in [10]. In our project, We replace the attentions layer by the BiDAF attention.

3.5 Model encoder layer

We apply three stacked encoder blocks for modeling layer. They share the same weights. Compared with the encoder block in embedding layer, the stacked encoder block has one more convolution layer. In our methods, we simplify the encoder blocks with less stacks of convolutional layer. We also tried to replace 1D convolutional layer with GRU. The results and discussion will be shown later.

3.6 Output layer

In output layer, we predict the probabilities of the starting position and ending position. The detail of probability calculations are provided in [11].

4 Experiments

4.1 Data

The dataset being used is the Stanford Question Answering Dataset 2.0 (SQuAD 2.0). Comparing to the SQuAD 1.1 dataset, the SQuAD 2.0 dataset adds unanswerable questions that are purposely made similar to answerable ones [5]. Therefore, the SQuAD 2.0 dataset is expected to be more challenging than the SQuAD 1.1 dataset.

4.2 Evaluation method

We use two scores, F1 score and Exact Match (EM) to evaluate the performance of our models on the SQuAD 2.0 dataset.

4.3 Experimental details

The first step of our project is to implement the QANet model as specified in [11]. QANet was originally developed by replacing the recurrent layers in models such as Bidirectional Attention Flow (BiDAF) with one-dimensional convolutional layers and self-attention layers. The building blocks of the QANet is the so-called encoder block consisting of mutiple one-dimensional depth-separable convolutional layers followed by a self-attention layer and then two fully connected layers. Layer normalization and residual connection are used for each of these layers. In addition, there are several detail treatments such as the extensive use of dropout and stochastic depth (layer dropout) to regularize the model and mitigate overfitting. In our implementation, we tried to respect all the aforementioned specifications in the original QANet model. As for the specific implementation, the three key-components, multi-head attention, context-query attention and postional encoding are based on the implementation from the baseline BiDAF code, the original Transformer-XL implementation for NMT and the original Transformer implementation for NMT. In addition, we applied Xavier initialization for all the convolutional and fully connected layers which helps to prevent training divergence especially in the early steps.

For the standard QANet, we tested two set of model configurations. The first set of model configuration, referred to as 'Standard QANet 8 head' is the same as in the original QANet paper [11], except for the dimension of the character embedding vector e_{char} that is 64 comparing to 200 in the original paper. The second set of model configuration, 'Standard QANet 1 head', is relatively light-weighted that uses a smaller embedding dimension ($e_{\text{embed}} = 96$) and single head attention.

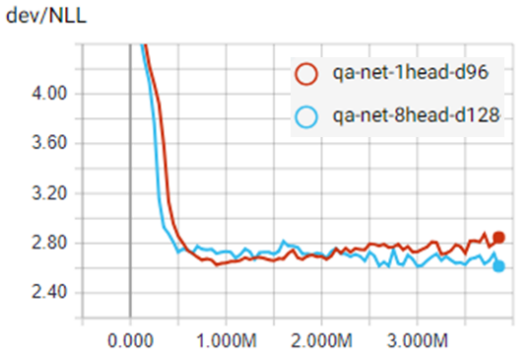


Figure 2: Evolution of the evaluation set NLL loss for the two standard QANet configurations.

Figure 2 shows the evolution of the evaluation set NLL (negative log-likelihood) loss. We observe that the loss increases slightly towards the end of training, especially for the single head configuration, which indicates slight overfitting. In addition, despite that the QANet contains no recurrent layers, the training process still takes significantly longer time than the baseline BiDAF model. This is likely due to the relatively large number of encoder blocks and relatively large number of convolutional layers within each block. Based on these two observation, in the next step we tested a light-weighted QANet by reducing the number of encoder blocks and the number convolutional layer within each encoder block to accelerate the training process and investigate if reducing the model complexity would mitigate overfitting.

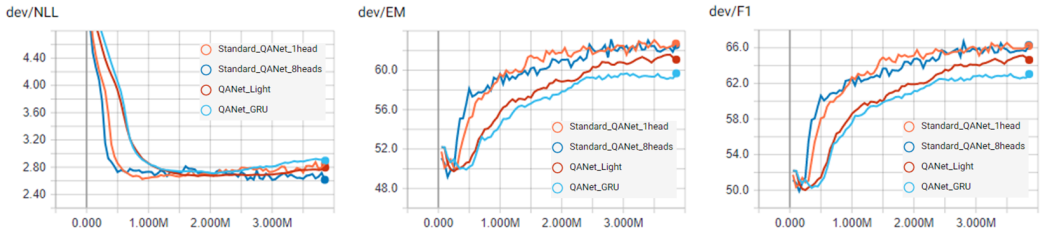
For the light-weighted QANet, we reduce the number of convolutional layers from 4 to 2 for the embedding encoder layer. In addition, we reduce to number of encoder blocks from 7 to 3 for the model encoder layer. This model is referred to 'QANet Light'.

We also experimented with another direction of simplifying the model architecture, that is reducing the number of encoder blocks from 7 to 1 for the model encoder layer, and replacing the multiple convolutional layers in all encoder blocks with one single gated recurrent unit (GRU). The model is referred as 'QANet GRU'.

All models are trained for 30 epochs with a learning rate of $l_r = 0.5$. Due to the limit on time and computing resources, different models are trained on different numbers of different GPUs and with varying batch size, therefore the training time comparison is not fair and we will present results for the training time.

4.4 Results

Figure 3 displays the training process for models implemented. The GRU model has an increasing NLL loss after 2 million iteration, which is an indication of overfitting and explains the under-performance on EM/F1 score.



(a) NLL (b) EM (c) F1

Figure 3: Training process for different models.

Table 1 summarizes the results for both dev-set and test-set of the models implemented. Our lighted version of QANet (e.g. Light, GRU) slightly under performs the basic implementation of QANet, but the EM/F1 scores are still quite close.

We take the best three models, the Standard QANet 8 head model, the Standard QANet 1head model and QANet Light model to create an ensemble model. The ensemble model achieved an EM score of 64.258 and F1 score of 67.598 on the dev set and an EM score of 62.096 and F1 score of 65.413 on the test set.

Model	Dev EM	Dev F1	Test EM	Test F1
Standard QANet 8 heads	63.07	66.60	-	-
Standard QANet 1 head	63.15	66.72	-	-
QANet Light	60.56	64.30	-	-
QANet GRU	59.57	62.99	-	-
Ensemble model	**64.258**	**67.598**	**62.096**	**65.413**

Table 1: Dev & test set results for models implemented

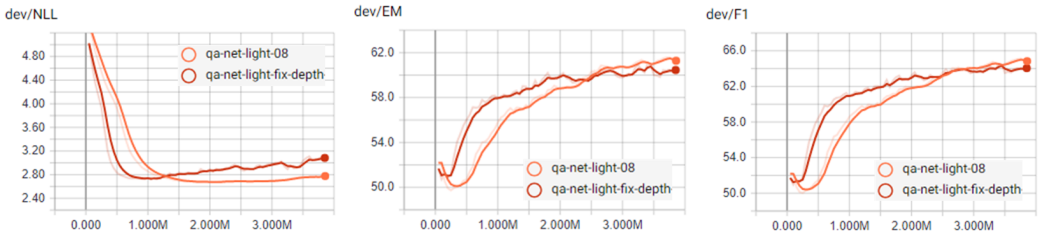
5 Analysis

5.1 Ablation analysis

In the ablation analysis, we focus on the stochastic depth (layer dropout) mechanism. We take the QANet Light model as the reference model, which contains layer dropout as described in the original QANet paper [11]. We remove the layer dropout from the QANet Light model, the resulting model is referred as the 'QANet Light Fixed Depth' model. The dev-set scores of the two models is shown in Table 2. The results from the two models are comparable. However, if we look at the training process in Fig. 4, it is clear that the QANet Light Fixed Depth model converges faster in the early steps but experiences significantly more overfitting in the later steps. This indiciates that the layer dropout layer does help mitigate overfitting.

5.2 Attention visualization and analysis

The context-to-query attention and query-to-context attention give us more intuition on the middle process of how the model finds the correct answer. In this section, we show a example where the model correctly predicts the answer.



(a) NLL (b) EM (c) F1

Figure 4: Training process for the two models in the dilation study.

Model	Dev EM	Dev F1
QANet Light	60.56	64.30
QANet Light Fix Depth	60.75	64.27

Table 2: Dev & test set results for ablation study

Context: The English name "Normans" comes from the French words Normans/Normanz, plural of Normant, modern French normand, which is itself borrowed from Old Low Franconian Nortmann "Northman" or directly from Old Norse Norðmaðr, Latinized variously as Nortmannus, Normannus, or Nordmannus (recorded in Medieval Latin, 9th century) to mean "Norseman, Viking".

Query: What is the original meaning of the word Norman?

Answer: Norseman, Viking

Prediction: Norseman, Viking

The Query-to-context (Q2C) attention shows which context words have the closest similarity to the query words. We show the Q2C attention in Figure 5. Excluding the punctuation, the words that have highest attention with respect to the query are 'comes' and 'itself', which, in some sense, reasonably represent the key meaning of the query.

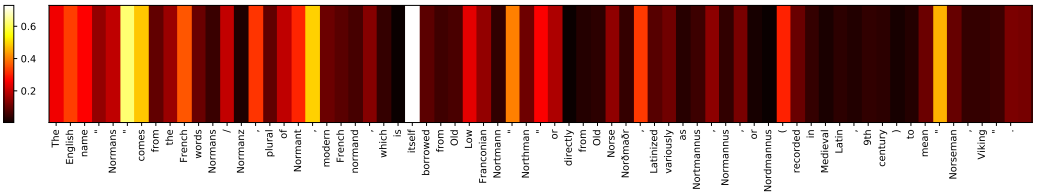


Figure 5: Query-to-context attention (max-pooled over query)

The Context-to-query (C2Q) attention is shown in Figure 6. Table 3 lists the three most relevant context words for each of the query word in this example. For the query word 'What', the most relevant words are 'Normans', 'Norðmaðr', and 'Normant', which are clearly closely correlates to what the word 'What' is referring to. The relevant context words for query word 'original' are: 'Normans', 'English', 'Old', which, are not necessarily exact, but offers some reference to the meaning of the query word.

5.3 Error analysis

In addition to compare the EM and F1 score for model evaluation, we also conduct the experiments on the relationship between the performance and the answer length, and question types. Figure 7 shows the average score regarding to answers of different length and questions of different types for our QANet light and Standard QANet 1 head method.

The left figure shows that the average F1 score decreases as the increase of the length of correct answer. The results for both methods are almost the same. Since the answers for most questions

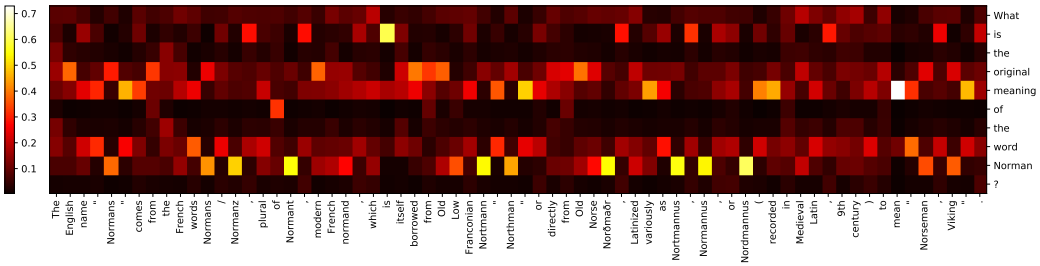


Figure 6: Context-to-query attention

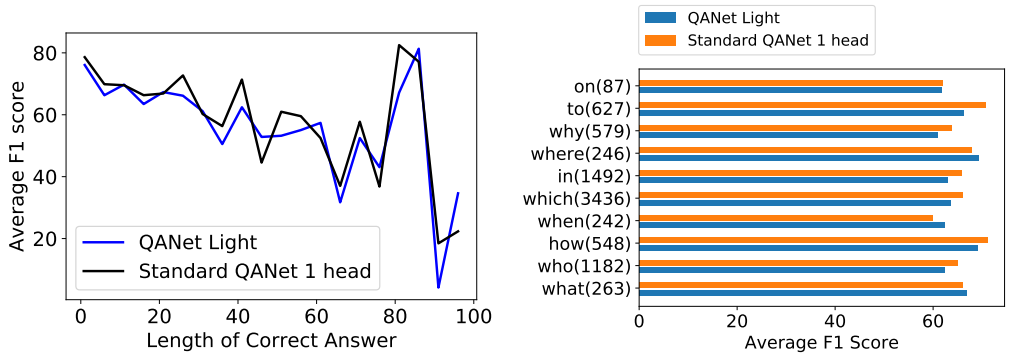
Word in query	Most relevant words in context
What	Normans, Norðmaðr, Normant
is	is, name, plural
the	the, The, Normans
original	Normans, English, Old
meaning	", mean, "
of	of, Normans, Nortmann
the	the, The, Normans
word	", ", plural
Norman	Normant, Normans, Nortmann
?	Normant, Normanz, Nortmann

Table 3: Three most relevant context word for each query word

are short, the influence of answer length on the F1/EM score is not obvious. It indicated that both methods are not good at generating long-length answers.

The F1 score for different question types are shown on the right figure in Fig. 7. We divided all of the questions into 10 types, including 'on', 'to', 'why', 'where', 'in', 'which', 'when', 'how', 'who', 'what'. The number of data are shown on the corresponding y labels. Both methods get a high F1 score on 'to', 'how', 'where' problem. And The Standard QANet 1 head result has a higher F1 score for most question types. But for question type of 'when', both methods perform bad.

Based on the observations of relationship between performance and questions or answer, both of our methods perform well on find the relevant words and answering most of questions. But if the length of correct answer is long or the question is specified in some types, these models provide a low F1 score.



(a) F1 score for answers of different lengths

(b) F1 score for different question types

Figure 7: Average F1 score for answers of different length and questions of different types.

6 Conclusion

In this project, We implemented QANet architecture to perform machine comprehension task on SQuAD 2.0 dataset. For the basic implementation (single model), we achieved EM and F1 score of 63.15 and 66.72, respectively. An ensemble of 3 models achieved EM of 64.26 and F1 of 67.60, which is near SoTA accuracy. An ablation study was conducted to analyze the functionality of important layers. We also drew some interesting insights form visualization of Q2C and C2Q attentions.In addition, an accuracy study was conducted over question types and length of answer. We noticed that the accuracy drops as the length of correct answer increases. We want to look further into the issue in the future work, and adapt frameworks suitable for processing long text words (e.g. Transformer-XL) to handle the problem.

Additional information

Acknowledgements

The experiments are conducted on the cluster of Stanford Center for Computational Earth & Environmental Siciences (CEES). The provided computational resources are greatly appreciated.

Honor code clarification

In the milestone report, we noted that for the implementation of the baseline model, we referred to the publicly available code `https://github.com/andy840314/QANet-pytorch-`. As we realized it was a violation of honor code from the feedback of the grader, we discarded the entire code immediately and re-implemented the QANet code based on a Transformer NMT repo `https://github.com/jadore801120/attention-is-all-you-need-pytorch`, which was explictly approved by the TA on Piazza and a Transformer-XL repo `https://github.com/kimiyoung/transformer-xl`. Please refer to our submitted code and our Piazza post for more information.

References

[1] Z. Dai, Z. Yang, Y. Yang, W. W. Cohen, J. Carbonell, Q. V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] H.-Y. Huang, C. Zhu, Y. Shen, and W. Chen. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. *arXiv preprint arXiv:1711.07341*, 2017.

[4] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[5] P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[6] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[7] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[9] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198, 2017.

[10] C. Xiong, V. Zhong, and R. Socher. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*, 2016.

[11] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018.