# Adapting QANet for SQuAD 2.0

**Parthasarathy Suryanarayanan**
psuryan@stanford.edu

**Yash Mangilal Jain**
yashjain@stanford.edu

## Abstract

The QANet architecture [13] was evaluated against the SQuAD 1.1 dataset [8] with impressive results (EM: 82.47 and F1: 89.30). The SQuAD [2.0] dataset [7], the latest revision of SQuAD, tests the ability of a system to not only answer the question whenever possible but also determine when no answer is supported by the paragraph and abstain from answering. Thus, it poses a much more challenging machine comprehension task. The main objective of this work is to evaluate the QANet system for the SQuAD2.0 dataset, by implementing the architecture from the ground up and understand the challenges posed by the non-answerability. We explore the ways to adapt the system based on the findings. An adaptation stretch goal is to exploit the ideas from Transformer-XL[3] in QANet architecture. In this update, we attempt to summarize the progress so far and discuss further directions.

## 1 Introduction

We can define machine comprehension in terms of Question Answering in its most general form [1]. Recently a variety of neural architectures have achieved near-human performance in open domain question answering tasks. These recent advances are broadly of two types – (1) Pre-trained Contextual Embeddings (PCE) based methods such as ELMo[6], Bert[4] etc. and (2) Non-PCE methods. The former offers more of an "off-the-shelf" module that could be employed for specific tasks such as question answering, the latter, even though not being the state of the art (SoTA) any longer (as of Feb 2019), offer more scope for creativity and opportunities for deep learning practitioners to explore different techniques and develop intuitions behind them. One of the successful non-PCE approaches to machine comprehension is the QANet architecture.

The main goal of the QANet architecture is efficiency / speed which also results in performance gain as a nice by-product. The authors of the system demonstrate the efficiency and performance improvements on SQuAD [1.1] dataset. However, suitability of the architecture for the non-answerability setting has not been systematically evaluated to our knowledge. Our major focus is understanding the limitations of QANet against SQuAD [2.0] dataset and investigating methods to augment the architecture, to predict if the question can be answered or not based on the information from the comprehension. We describe our implementation of the QANet architecture in the following section.

## 2 Architecture

Prior to QANet, the SoTA question answering systems, such as BiDAF [9] were composed of two key components (1) Recurrent units such as LSTM for capturing sequential input and (2) exploiting attention mechanism for capturing long-term interactions. Due to the sequential nature, (1) can't be parallelized to exploit the hardware and hence is slow. To achieve the speedup, the QANet borrows a neat idea from NMT, proposed in the Transformer[12] architecture: encode the question and context separately using a non-recurrent, therefore faster, encoder. This encoder uses convolution (which models the local features) and self-attention (which models global interaction) as building blocks. The parallel nature of CNN architectures leads to a significant speed boost, especially given the fact that context passages in SQuAD are really long.

### 2.1 Representation

Before we dive deep into the architecture, it is useful to define the terminology and representation. Recall that the QA problem can succinctly be formulated as following: Given a context / passage of n words, $C = \{c_1, c_2..., c_n\}$ and a question / query sentence of m words, $Q = \{q_1, q_2..., q_m\}$, we want to output a span $S = \{c_i, c_{i+1}, ...c_{i+j}\}$ from the original paragraph C if the question is answerable. QANet uses fixed size contexts and questions. In the default implementation, only the first 400 tokens in the given paragraphs are used as context and 50 tokens in the given question are used as query. For each word, first 16 characters are only used to compute the character embeddings. If the context/query is shorter than the length threshold, zero-padding is applied. QANet uses pre-trained GLoVE vectors as word and character embeddings for representing both query and context.

### 2.2 Model

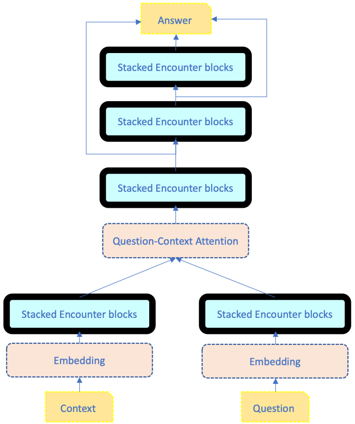


Figure 1: Overall Model - logical view

QANet architecture consists of 5 key modules: an embedding layer, an encoder layer, a context-query attention layer, a modeling layer and an output layer. Embedding layer combines the word and character embeddings into a singular representation for each word. After that, the context and the question are passed through their respective encoder layers. The output from the encoders are fed to the Context-Query attention layer which combines context and query and produces a representation for each word in context. The key layer for prediction is the output layer. Similar to method described in [9], the optimization function is formulated as follows. Let

$$P^1 = \mathit{softmax}(W_1[M_0; M_1]) \qquad P^2 = \mathit{softmax}(W_2[M_0; M_2])$$

be the probabilities of each position in the context being start and end of the answer span, respectively.

Where $W_1$ and $W_2$ are trainable weights and $M_0, M_1, M_2$ are outputs from the three model encoders. Now the objective function could be defined as the sum of negative log probabilities of predicted distributions indexed by true start and end indices, averaged over the training instances.

$$Log(\theta) = -1/N \sum_i^N [log(P^1_{y^1_i}) + (P^2_{y^2_i})]$$

Where $y^1_i$ and $y^2_i$ are start end positions of the $i^{th}$ training instance from ground truth. The model outputs probabilities for all pairs similar to other pointer networks for reading comprehension. Through an efficient method (such as using dynamic programming), the best pair of spans $(s, e)$ that maximize $p^1_s p^1_e$ have to be chosen as the prediction. Ultimately these probabilities are turned into a span consisting of a start index and an end index, indicating the location of the answer in the context paragraphs. Since, the answer is essentially a contiguous phrase in the context, predicting start and end index is sufficient. The overall model is shown below.

## 3 Approach

Our motivation behind choosing this project is to carefully study the key ideas (such as, attention mechanism, pointer networks etc.) from the recent innovations of neural architectures that the QANet is built upon. We hope to gain hands-on experience by implementing the architecture “from the scratch”. So the first step essentially is implementing the architecture. The second step is to perform an error analysis on the development set and come up with a few ideas to implement. At the time of writing this, we are still in the process of training our first implementation and hence we have not explored any ideas fully. At the outset, we want to explore ideas along two different dimensions: 1. Answerability prediction improvements, 2. General accuracy improvements for answerable questions.

### 3.1 Answerability Improvements

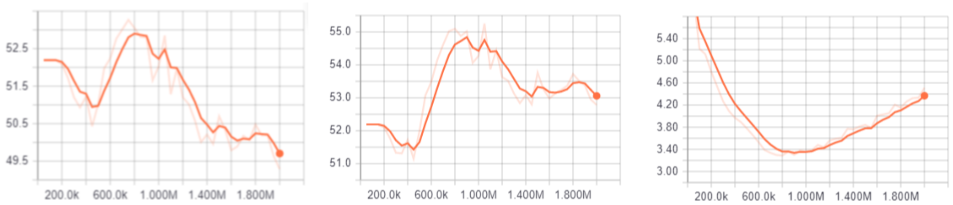
To deal with unanswerable cases, systems must learn to identify a wide range of linguistic phenomena such as negation, antonymy and entity changes between the passage and the question. Therefore we expect the performance of the base QANet over SQuAD [2.0] to be worse. Previous works, including the framework supplied as part of the baseline implementation for default final project, usually predict an additional “no-answer” probability to detect unanswerable cases. Given a context of N tokens, this model predicts start/end probabilities for N+1 positions. The start/end probabilities in the extra position are used for answerability prediction. Other approaches include adding auxiliary losses to the system [5] or encoding the no-answer option within the model [2].

### 3.2 Accuracy Improvements for Answerable Questions

After the error analysis we would explore ways of borrowing some ideas from Transformer-XL architecture [3] to improve accuracy of QANet, especially for handling long-range dependencies.

## 4 Experiments

Currently we are in the process of training our initial implementation. Due to GPU memory constraints we have configured the experiments to run with the following settings: The hidden size of the model is 96 (in contrast to the 128 from the paper). The number of attention heads is set to 1 (instead of 8). Further, batch size is limited to 32 (instead of 64). The running time of the experiments seem to be well in excess of 24 hours and the model learns slowly and we reach EM of 53.2 and F1 score of 55.2. We observe that well into the experiment (after 8 epochs) the NLL (native log-likelihood) loss suddenly starts climbing sharply, indicating some serious issues with our implementation which we are investigating.



(a) EM

(b) F1

(c) NLL

Figure 2: Training run

### 4.1 Data and Evaluation

We use the usual evaluation metrics for SQuAD - EM (Exact Match) and F1 measure. The baseline for our implementation will the Bi-DAF model supplied as part of the default final project. As a departure from the QANet paper, we have chosen not to re-implement the data-augmentation due to time-constraints; instead, if time permits, we will be evaluating against domain-specific datasets such as CliCR [11] to further explore the different comprehension skills (co-reference, tracking, logical etc.) of the system as described in [10].

### 4.2 Results

Currently we are in the process of training our initial implementation, so no results are available yet. This section will be updated later.

### 4.3 Future Work

The first order of business must be be to make sure that the QANet model learns best to the current specification of the loss function, which is not the case currently. Once we are able to train the model successfully we will update this section.

## References

[1] Christopher JC Burges. Towards the machine comprehension of text: An essay. *TechReport: MSR-TR-2013-125*, 2013.

[2] Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*, 2017.

[3] Zihang Dai*, Zhilin Yang*, Yiming Yang, William W. Cohen, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Language modeling with longer-term dependency, 2019.

[4] 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.

[5] 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.

[6] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

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

[8] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

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

[10] Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 806–817, 2017.

[11] Simon Šuster and Walter Daelemans. Clicr: a dataset of clinical case reports for machine reading comprehension. *arXiv preprint arXiv:1803.09720*, 2018.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[13] 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. *arXiv preprint arXiv:1804.09541*, 2018.