1. Research paper summary

Paper: "QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension" (ICLR 2018)
Authors: Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, Quoc V. Le
URL: https://arxiv.org/abs/1804.09541

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[2], BERT[3] 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 beginners to explore different techniques and develop intuitions behind them. As such, in the summary below, one of the recent papers, "QANet…" is discussed. The paper proposes a Q&A architecture called "QANet" which improves upon the performance of the then SoTA. The key goal of the architecture is efficiency / speed which also results in performance gain as a nice by-product. The paper demonstrates the efficiency and performance improvements over SQuAD1.1 dataset [4]. The system, heavily inspired by the Transformer[5] architecture, builds upon several earlier techniques, such as character-level embeddings, positional embeddings, depth-wise separable convolutions, multi-headed attention, residual blocks, layer normalization, two-layer highway network etc. Our main motivation behind analyzing this paper and using it as a basis for the default final project is to study these techniques carefully and to adapt the architecture suitably for the SQuAD2.0 dataset. The summary only covers aspects of the paper that are relevant to our objective.

Introduction

Prior to QANet, the SoTA Q&A systems, such as BiDAF[6] were composed of two key components (1) Recurrent models 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 paper borrows a neat idea from NMT, proposed in the Transformer[5] 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. On the decoding side, the paper uses the attention mechanism similar to BiDAF.

Model

Recall that the Q&A problem can succinctly be formulated as following: Given

a. A context / passage / document of n words, $C = \{c_1, c_2 \dots, c_n\}$
b. A question / query sentence of m words, $Q = \{q_1, q_2, \dots, q_m\}$

We want to output a span $S = \{c_i, c_{i+1}, \dots c_{i+j}\}$ from the original paragraph C if the question is answerable.

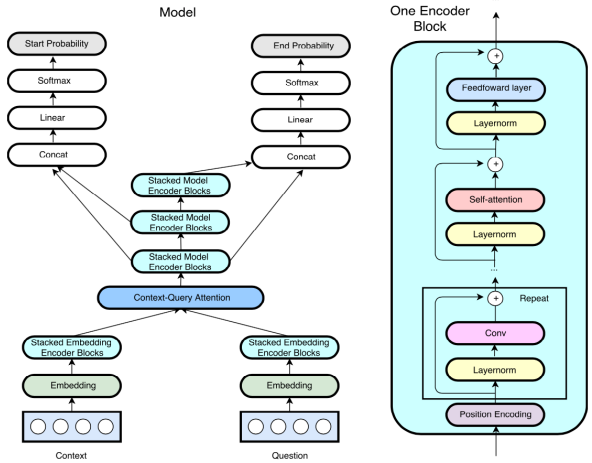

Figure 1: QANet Architecture

Encoder block

The QANet architecture is shown in figure 1. A key component that is used in several places, is the encoder block. It is shown on the right. The encoder block draws upon many ideas from the Transformer [5]. It is stacked several times and the stack is used in three different places: to encode the question, to encode the context and finally on the output side 3 times.

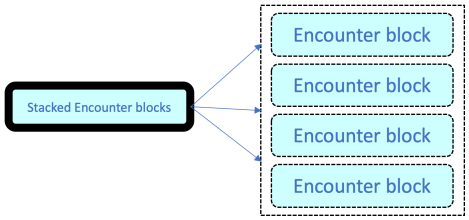

Figure 2: Staked encounter blocks

The encoder block consists of four key layers.

1. Positional embedding layer: The position of a word within a sentence carries useful information for the model. In an RNN, this information is trivially encoded as the inputs are presented to the model one step at time. But for non-RNN model, a "positional encoding" is needed to be given explicitly. This layer adds the positional encoding to the input at the beginning of each encoder layer as defined in Transformer [5]
2. Convolution layer (repeated multiple times and the number varies based on the location of usage of the encounter block): the implementation uses depth-wise separable convolutions [11,12] which require an order of magnitude less computation to achieve nearly the same result as the normal convolution. This is a key ingredient of the speedup. (Details: kernel / filter size: 7, number of filters: 128)
3. Self-attention layer: the multi-head attention mechanism, again from Transformer [5] is used with number of attention heads as 8. Attention key depth is 128 and depth per head is 16.
4. Feed-Forward layer

Key techniques used: three different regularization methods are employed to stabilize the network and consequently to improve the training speed: Layer normalization [13], L2 Regularization [14], Residual connections [15]

The overall model is shown on the left in figure 1. The model consists of 5 layers, described below.

1. **Input embedding layer:** the purpose is to obtain a rich low dimensional representation of the query Q and context C. This is done by
    a. Looking up the 300 ($p1$) dimensional pre-trained GloVe [8] embeddings $[x_w]$
    b. Learning a 200 ($p2$) dimensional embedding for each character in the word using convolution $[x_c]$. The character embeddings are passed through a convolutional layer and a max-pooling layer, as described in [17], to produce 200-dimensional character-level word embeddings. The intuition behind using a character-level embedding is to be able to handle out of vocabulary words (OOV), rare words, misspellings and morphology during test time.
    c. Concatenating the two vectors into $[x_w;\ x_c]$ into a vector of size $(p1 + p2)$

    Key techniques used: A two-layer highway network [10] is also used as optimization to avoid the vanishing gradient problem.
2. **Input embedding layer:** embedding layer output is fed to the encoder layer to generate corresponding context and question representation. This layer uses a single encoder block (with 4 convolutions). The output of this layer is of 128 dimensions (d) as the input is immediately converted to this dimension by a one-dimensional convolution.
3. **Context-Query Attention layer:** this is similar to the previous reading comprehension models, such as [12] and [16]. The purpose of this layer is to calculate two quantities.
    a. Context-to-query attention (A): First calculate pair wise similarities between words in the encoded context (C) and encoded query (Q). The similarity measure is a trilinear function [16]

    $$f(q, c) = W0[q, c, q \odot c]$$

    This would yield a matrix S of dimensions nxm. $W0$ is a trainable variable. After normalizing each row using $softmax$, we get a different matrix $\bar{S}$. The context-to-query attention is calculated as

    $$A = \bar{S} \cdot Q^T \text{ (of dimensions } nxd)$$

b. Query-to-context attention (B): Column-normalizing S yields $\bar{\bar{S}}$. The query-to-context attention is calculated as $B = S \cdot \bar{\bar{S}}^T \cdot C^T$

4. **Model encoder layer**: The input to this layer at each position is $c, a, c \odot a, c \odot b$ where a and b are rows from attention matrices A and B calculated above. Layer details: number of stacks: 3, number of encoder blocks within each stack: 7, number of convolutions within each: 2. All the three stacks share weights.
5. **Output layer**: The output layer is task-specific. Similar to method described in [9], the optimization function is formulated as follows. Let $p^1$and p^2 be the probabilities of each position in the context being start and end of the answer span, respectively

$$p^1 = softmax\ (W_1[M_0; M_1]), \quad p^2 = softmax\ (W_2[M_0; M_2])$$

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 object 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.

$$L(\theta) = -\frac{1}{N}\sum_{i}^{N}\left[\log\left(\mathrm{p}^1_{\mathrm{y}^1_\mathrm{i}}\right) + \log\left(\mathrm{p}^2_{\mathrm{y}^2_\mathrm{i}}\right)\right]$$

Where y_i^1 and y_i^2 are start end positions of the i^{th} training instance from ground truth. The model outputs probabilities for all pairs. Through an efficient method (such as using dynamic programming), the best pair of spans (s, e) that maximize $p_s^1\ p_e^2$ have to be chosen as the prediction. The overall model can be pictorially depicted as in figure 3.

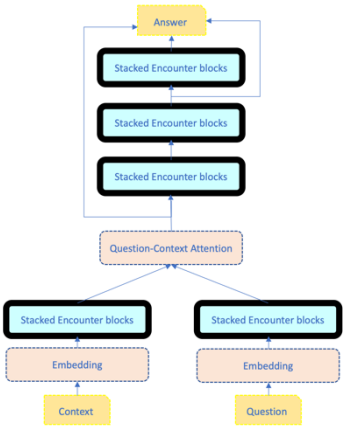

Figure 3: Overall model - logical view

Conclusion

This is one of the deepest neural architectures for NLP. The paper reports 16x and 12x improvements in speed for training and test respectively. As a consequence, it was possible to train the model longer and beat the then SoTA performance. As for the task-specific adaptation, in addition to several data preprocessing steps, the authors perform data augmentation by document paraphrasing using back-translation which seem to have yielded non-trivial improvements. This is based on the intuition that the authors also report performance on Wikipedia sub-dataset and hypothesize about techniques to improve performance on datasets containing multi-paragraph context, such TriviaQA. The authors also report several experiments, including ablation studies to measure the impact of individual components. Overall a well-rounded report of solid technical work.

2. Project Description

Goal

One of the successful Non-PCE approaches to machine comprehension is the QANet architecture, described in the paper summarized in the previous section. The architecture proposed by the paper was evaluated against the SQuAD 1.1 dataset [4] with impressive results (EM: 82.471 and F1: 89.306). The SQuAD [2.0] dataset[18], 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.

1. The main objective of our project is to evaluate and possibly, adapt the QANet system for the SQuAD2.0 dataset.
2. An adaptation stretch goal is to exploit the ideas from Transformer-XL[19] in QANet architecture.
3. Also, as stated earlier in the paper summary, our main motivation behind choosing this project is to carefully study the key ideas (such as, character-level embeddings, positional embeddings, depth-wise separable convolutions, multi-headed attention, residual blocks, layer normalization, two-layer highway network) from the recent evolution of neural architectures that the QANet is built upon and to develop intuitions behind them. Also, as an out-of-the-box QANet implementation is not readily available in PyTorch, we hope to gain hands-on experience by implementing the architecture "from the scratch".

Baseline and evaluation

We will be using the usual evaluation metrics for SQuAD. The baseline for our implementation will be the previously published score of QANet (single) by Google Brain & CMU on the SQuAD1.1 Leader board - EM: 82.471 and F1: 89.306. We may choose not to re-implement the data-augmentation as it is not in our learning goals; instead, if time permits, we will be evaluating against domain-specific datasets such as CliCR (Clinical Case Reports for Machine Comprehension [20]) to further explore the different comprehension skills (co-reference, tracking, logical etc.) of the system [21].

References

[1] Towards the Machine Comprehension of Text: An Essay, https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/mct.pdf
[2] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Ken- ton Lee, and Luke Zettlemoyer. Deep contextualized word representations. arXiv preprint arXiv:1802.05365, 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] 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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, pp. 2383–2392, 2016.
[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin: Attention Is All You Need
[6] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. CoRR, abs/1611.01603, 2016. URL http://arxiv.org/ abs/1611.01603
[7] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. CoRR, abs/1611.01604, 2016. URL http://arxiv.org/abs/1611.01604
[8] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word//w representation. In Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.
[9] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603, 2016.
[10] Rupesh Kumar Srivastava, Klaus Greff, and Ju ̈rgen Schmidhuber. Highway networks. CoRR, abs/1505.00387, 2015. URL http://arxiv.org/abs/1505.00387.
[11] Franc ̧ois Chollet. Xception: Deep learning with depthwise separable convolutions. CoRR, abs/1610.02357, 2016. URL http://arxiv.org/abs/1610.02357.
[12] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open- domain questions. In Proceedings of the 55th Annual Meeting of the Association for Computa- tional Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers, pp. 1870–1879, 2017.
[13] Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. CoRR, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.
[14] Andrew Ng. Feature selection,L_1 vs.L_2 regularization, and rotational invariance. ICML 04
[15] Identity Mappings in Deep Residual Networks. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, abs/1603.05027v3
[16] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Making neural QA as simple as possible but not simpler. In Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017, pp. 271–280, 2017.
[17] Yoon Kim. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25- 29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1746– 1751, 2014.
[18] Pranav Rajpurkar, Robin Jia, Percy Liang, Know What You Don't Know: Unanswerable Questions for SQuAD, arXiv:1806.03822
[19] 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. 2018.
[20] Simon Šuster, Walter Daelemans CliCR: A Dataset of Clinical Case Reports for Machine Reading Comprehension, arXiv:1803.09720
[21] Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability, http://aclweb.org/anthology/P17-1075,