Critical Review of QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension

Daniel Campos
University of Washington
dacampos@uw.edu

Abstract

Currently, most Deep-Learning solutions for Question Answering(Q&A) are based on Recurrent Neural Networks(RNN). Yu et al. introduce a new system, QANET, that replaces RNNs with Convolutions Neural Networks(CNN) which delivers 13x faster training, 4x-9x faster inference and a 2.1 gain on F1 on the SQUAD Dataset. This use of CNN along with implementation of augmented data should inspire much of the NLP community to rethink what neural structures they use and explore how to augment and expand their existing data sources.

1 Summary

The technological explosion of phones and headless devices like Amazon Echo (Amazon, 2018b) and recent advances in speech recognition technology (Hinton et al., 2012) have fueled the mainstream adoption of agents such as Siri (Computing, 2018) and Alexa (Amazon, 2018a). Users of these devices interact with agents through open-domain QA which has created a clear need for systems that can read though unstructured text and product an answer extremely quickly and accurately. Hoping to replicate the effects that large publicly available datasets like ImageNet's (Deng et al., 2009) had on computer vision, language researchers have released large scale QA datasets such as SQUAD (Rajpurkar et al., 2016) and MS-MARCO (Nguyen et al., 2016). Since released, these datasets have helped inspire hundreds of models which have uncovered scalable effective ways for QA.

Yu et al. use these catalysts and datasets to introduce their QANET (Yu et al., 2018) system and in doing so both set the record for performance on the SQUAD v1.1 dataset(now closed) and call into question some of the core architectural decisions the QA community had settled on. The QANET model is built on many of the standard QA cutting edge systems but by swapping a standard RNN for a CNN the QANET team is able to decrease the time overhead training most models have. QANET boasts a 13x increase in training time which allows the researchers not only to speed up their iteration cycle but allows them to explore the effects of artificial data can have on system performance and stability. The authors clearly indicate how much value they are able to gain from increased training speed by demonstrating how various gains are achieved throughout their paper. Since retraining a model can now happen quickly, researchers are able to fine tune every component of the model and understand what effect this tuning has.

1.1 Problem Addressed and Prior Work

Yu et al. formalize the QA problem into a simple solvable problem: given a query and a context passage produce a sub span of the context passages that answers the query. This definition is broad enough that it can address most potential QA problems but specific enough to remove any tasks like context passage selection and ranking. This simplification also removes the hardest constrain required for real world usage, context passages may not contain the correct answer. The reduction of the problem area into SQUAD's dataset allows the problem to be simplified into two classifiers: prediction of the start character and prediction of the end character. Separate to problem definition, the authors do a good job of dealing with prior work by calling out major advances with specific advances but they also generalize the types of advances which allows the reader to focus on the effects instead of the system. Despite the relative

depth their evaluation of prior work, the work can be lacking since it really only focuses on how systems perform on the SQUAD dataset and on TriviaQA. All other RC systems for the many other popular datasets are ignored.

1.2 The Approach

The approach the authors took can be broken down into three areas: model architecture, model optimization and data augmentation. In order to prove how useful CNNs can be, the QANET model is based on the industry standard Reading Comprehension(RC) model which consists of an word embedding (Pennington et al., 2014) and self attention (Vaswani et al., 2017) to produce a context-query representation. Next, the authors implement CNNs in the embedding encoder layer and the model encoder layer with a focus on memory efficiency. The authors do not dive into depth about their hyper parameter choice but they likely optimized their system thoroughly. As mentioned previously, usage of CNNs allows the model to train substantially faster since each neuron no longer has as many dependencies. After the authors have built out their system with CNNs they focus on the effects that each component has on the final F1 score. This fine grained investigation of systems performance allows clear evaluation of the effects minor changes have. Each decision in depth and breath of CNN causes a trade off in training time and performance and by showcasing these numbers Yu et al. prove why their architecture is optimal. To take advantage of the 13x speedup in throughput, the authors proceed to use a simple process to produce 3x the original data. This augmentation is created by taking both the query and the context and translate them to another language and then back into English. The output of this process is a new training example that may contain slightly different phrasing or word usage to the original and have a similarly perturbed answer. To make their work more compelling, the authors could have dove a little deeper to focus on why previous literature has focuses so deepley on using RNNs. By never providing insight into why the field has made RNNs the de-facto standard it can be challenging to understand the trade offs caused by choosing CNNs. Like most systems in computation, there is never a structure that is always better than another, each tends to have a job it excels in. CNNs produce a training gain but what is lost?

1.3 Experimental Results

The authors do a great job of providing detailed results on their system. Using the SQUAD evaluation framework and its well defined EM and F1 scores, the team provides a detailed breakdown of model improvement gained from each of their optimization techniques. They provide a detailed analysis comparing their models performance to other top systems on the SQUAD data set along with benchmarks for the effects that various architecture decisions have on both training and inference time and EM and F1 performance. By doing so they not only show how impressive their model is but highlight the impact various parts of their system have and showcase how they reached their final model. Additionally, they do not only focus on the v1.1 SQUAD dataset but they also research how they model performs with the Adversarial SQUAD dataset (Jia and Liang, 2017). In this new, more difficult dataset, QANET performs quite well which is likely attributed to their training on artificial data, helping the system generalize. It would have been more useful if the authors had broken down performance on this adversarial dataset in similar fashion to their regular performance on SQUAD. What is the optimal data augmentation for performance on the adversarial set?

1.4 Applicability

Much of the work Yu et al. introduce is impactful and exciting because it can be generalized to many other popular NLP tasks. By demonstrating how CNNs can replace RNNs QANET opens the door for application of successful techniques in computer vision to text and language based tasks. Additionally, Wei demonstrates how to generate additional artificial data from a human annotated corpus in a way that is scalable to many other datasets. The most interesting effect is how QANET is able to perform better on the adversarial SQUAD dataset since the perturbed data makes the model more robust. Hopefully this model robustness can continue into SQUAD 2.0. I look forward to reading how other research areas in NLP use CNNs and generate artificial data as a source of model optimization.

1.5 Limitations

This paper, much like most others in the QA space, are quite limited when viewed in potential global

use cases. The dataset the team researched is completely English based and they have not tested their model's performance on QA datasets in other languages, such as Dureader(He et al., 2017). Further more, by focusing on squad, performance on documents that are not as well structured or do not match the high content bar the wikipedia-based SQUAD tends to provide. Similar to other QA systems, the reliance on neural architecture limits the size of context paragraphs that can get consumed. Simply put the model cannot scale to larger documents because text documents are much larger than the 400 characters their model supports. Finally, and potentially most importantly, the focus on SQUAD and TriviaQA means their system is blind to a very common problem: the question is not actually answerable by the current contextual passage. For a system to perform well in real world usage it is imperative that a system can learn when it shouldn't answer a question because arguably a bad answer is worse than no answer.

1.6 Applicability

As mentioned previously, almost everything this paper is doing can be broadly explored in many other NLP tasks. CNNs and Capsule-nets (Sabour et al., 2017) may potentially be the source of huge improvements across POS-taggers, Named-Entity-Resolvers, and countless other tasks. Additionally, since most NLP problems currently rely on large datasets, the data augmentation approach the authors use could be generalized and explored

1.7 Future Work

The easier way to continue this rewarding vein of research is to expand the variety of dataset and QA tasks the system is trained and tested on. The model they build is deliberately built with a task specific layer so training a system to work on Dureader, or TriviaQA would be very straightforward. Comparing performance on these benchmarks would do the QA community a great job of understanding how various techniques generalize data.

References

Amazon. 2018a. Amazon alexa. http://alexa.amazon.com/.

Amazon. 2018b. Amazon echo. https://en.wikipedia.org/wiki/Amazon_Echo.

Apple Computing. 2018. Siri personal assistant. http://www.apple.com/ios/siri/.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fe. 2009. Imagenet: Alarge-scalehierarchicalimagedatabas. *CVPR* http://www.image-net.org/papers/imagenet$_c vpr09.pdf$.

Wei He, Kai Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *CoRR* abs/1711.05073.

G. Hinton, L. Deng, D. Yu, G. Dalh, and A. Mohamed. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29(6):82–97.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *EMNLP*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *CoRR* abs/1611.09268.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text https://arxiv.org/abs/1606.05250.

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. In *NIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR* abs/1804.09541.