SQuAD 2.0 (QANet, Character Embeddings and Token Features)

Bernardo Casares, Eric Nielsen, Eric Redondo
{bcasares, nielsene, eredondo}@stanford.edu

Problem

Training computers to understand text has gained significant popularity over the past several years due to the many applications it enables. Imagine being able to ask a computer to better understand a piece of text. Significant research has been done to design models to compete in this difficult challenge.

In general, submissions fall into one of two categories: those that use Pre-trained Contextual Embeddings (PCE) such as ELMo and BERT, and those that do not. PCE models tend to have much higher performance than non-PCE models, however come with an added level of complexity. **We chose to focus our project on enhancing existing non-PCE models.**

Data / Task

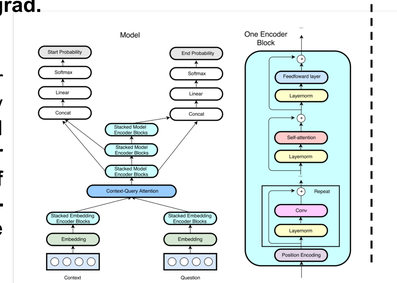
For the project we used the SQuAD 2.0 dataset, one of the most popular reading comprehension benchmarks. **The dataset comprises many questions, each associated with a context paragraph that may or may not include the answer.**

The goal of the project is to train a computer to answer the questions as correctly as possible – providing a measure for how well the computer can `understand' text.

Approach

Architecture Overview

1. **Baseline:**
   - The baseline model is inspired by the **BiDAF** model developed by Seo et al.; used 300 dim. Glove word embeddings
2. **Additional Embeddings (BiDAF)**
   - **Character-Level-Embeddings (+ 64 dim.)**
     - To reason about words not included in the vocabulary, we concatenate the word-level embeddings.
   - **Additional Word Features (+ 12 dim.)**
     - Added three embedding features using different attributes of each token: **Part-of-speech, named entity recognition tags, and normalized term frequency vectors.**
   - **Hyperparameter Tuning and Additional Updates**
     - In an effort to make the model more expressive, prevent overfitting, and improve the model's convergence, the **hidden size and the drop probability was increased, an L2 weight decay was added, and the Adam optimizer was substituted for Adagrad.**
3. **QANet**
   - Addresses one of the major issues with BiDAF by **replacing the the RNN encoder with an encoder consisting only of convolution and self-attention** – making the code parallelizable.

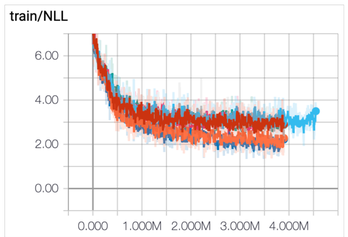


Experiments and Results

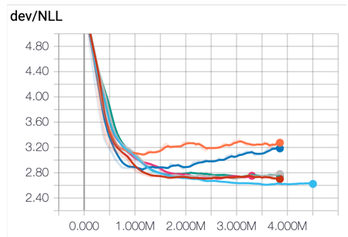
BiDAF Models

The model using full embeddings (word, character, POS, NER tag, and TF) with tuned hyperparameters performed the best out of the 7 models tested. Compared to the baseline model, this is an **EM score increase of 3.66 and an F1 score increase of 3.30**.

Model	EM Score	F1 Score
Dev		
Baseline BiDAF	56.14	59.89
Word + Character Embeddings	58.91	62.06
Word + Character + POS Embeddings	59.74	62.80
Word + Character + NER Embeddings	58.34	61.78
Word + Character + TF Embeddings	59.13	62.38
Word + Character + POS + NER + TF (All) Embeddings	59.50	62.82
All Embeddings, Hyperparameters Tuned (Best)	**60.07**	**63.19**
Test		
All Embeddings, Hyperparameters Tuned (Best)	**58.63**	**61.88**

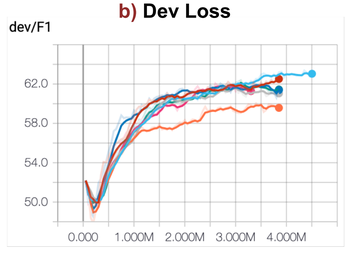


a) Training Loss



b) Dev Loss

c) Dev EM



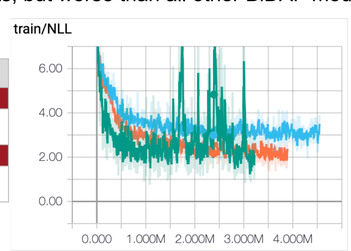
d) Dev F1

Orange: Baseline, **Cyan**: All Embed Tuned, **Red**: All Embed, **Blue**: Word/Char Embed, **Magenta**: Word/Char/POS Embed, **Gray**: Word/Char/NER Embed, **Green**: Word/Char/TF Embed

QANet Model

These scores are marginally better than the baseline results, but worse than all other BiDAF models.

Model	EM Score	F1 Score
Dev		
All Embeddings with QANet	**57.00**	**60.59**
Test		
All Embeddings with QANet	**56.30**	**59.79**



e) Training Loss

Orange: Baseline, **Cyan**: Best BiDAF, **Green**: QANet

Analysis

Ablation Studies

Of all model components analyzed, the **character-level embeddings had the greatest effect**. It is likely that, with the addition of character embedding information, the model is better able to handle instances of unknown words appearing in either the question or context by using character-level information to gain insight into these unknown words.

We also see evidence that the POS embeddings provided a significant positive effect while the other two provided very little, since performance drops off heavily when removing the POS embeddings, but remains nearly constant when removing both NER and TF.

Characteristic Examples

1. **Best BiDAF model**
   - **Often chose answers close to correct, but too lengthy**
   - Ex: **Question**: *'What is the most important type of Norman art preserved in churches?'* ***Correct****: 'mosaics'*, ***Model:*** *'sculptured fonts, capitals, and more importantly mosaics'*
2. **QANet model observed errors**
   - **Demonstrated an ability to answer with the correct type of item or idea, but often struggled with choosing which of the items from the context to choose from**
   - Ex: **Question**: *`Which directive mentioned was created in 1994?'* ***Context****: `...the 1994 Works Council Directive, which required workforce consultation in businesses, and the 1996 Parental Leave Directive...'*, **Correct**: *`Works Council Directive'*, **Model**: *`Parental Leave Directive'*

Conclusions/Future Work

BiDAF: Supplementing word-level embeddings with character-level embeddings leads to better performance However, adding only the part-of-speech feature seems to be nearly as effective as adding all three additional token features. Further tuning the model's hyperparameters and adding different, more descriptive token feature embeddings could result in even better performance.

QANet: For the QANet model there are many opportunities to further improve its performance. Many more experiments could be run using different hyperparameters. Specifically, **decaying the learning rate at later time steps and increasing the batch size and the model's hidden size would likely improve the sometimes eratic training behavior**.

References

[1] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. arXiv:1611.01603, 2016.
[2] 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:1804.09541, 2018.
[3] Attention is all you need in pytorch repository: https://github.com/jadore801120/attention-is-all-you-need-pytorch