Extended BiDAF with Character-Level Embedding

Stanford CS224N Default Project

Alyssa Romanos
Department of Computer Science
Stanford University
aromanos@stanford.edu

Gabe Saldivar
Department of Computer Science
Stanford University
gjsaldiv@stanford.edu

Kavita Selva
Department of Computer Science
Stanford University
kselva@stanford.edu

Abstract

With the rise of NLP and ML, we've seen much progress in regards to the task of machine comprehension and building robust question answering systems. we want to focus on investigating and improving the BiDAF model, starting from extending the baseline model by including character-level word embeddings. We then ran experiments using the improvements recommended in section 5.11 of the default project handout. Two major goals were accomplished: we implemented character-level embeddings and adjusted dropout rate and learning rate in addition to other hyper-parameters in order to improve our model. On our best model, we were able to achieve an F1 score of **65.106** and a EM score of **61.369** in the non-PCE division.

1 Key Information to include

- Mentor: Alvin Hou
- External Collaborators (if you have any): N/A
- Sharing project: No

2 Introduction

Question answering (QA) is an Natural Language Processing task that is concerned with building machine systems that answer questions in a natural language used by humans. Over the years, there has been much progress in this area of computer NLP. This task consists of a document that has information that contextualizes the question, and the goal is to answer the question using the information provided as context.

The work that has been done in this area of NLP has centered around model performance on the Stanford Question Answering Dataset (SQuAD). One of the most revolutionary model architectures has been the Bi-Directional Attention Flow (BiDAF) [1]. The BiDAF model uses an attention mechanism that computes attention for every step, thus reducing the information lost by summarizing the context paragraph. This attention mechanism is also memory-less and used in both directions, Q2C and C2Q. At the time of submission, their BiDAF model outperforms all other models using the Stanford Question Answering Dataset (SQuAD).

In this project, we will be further exploring this problem space by extending the BiDAF model. We extend the baseline BiDAF model by adding in a character-level embedding as well as making changes that are not related to the architecture of the model, such as trying hyper-parameter search, different types of RNNs, and optimization algorithms.

3 Related Work

There are many different models within the QA problem space that are trained on the SQuAD dataset. Although we are using SQuAD 2.0, we will be discussing models that are trained on SQuAD and or SQuAD 2.0. We used the BiDAF model by Seo et al. as our inspiration for our project, using a BiDAF without character-level embedding as our baseline model [1]. From this paper, we saw that we could improve our model by implementing the character-level embedding. We looked to this paper in order to implement the character-level embedding on our own. The single BiDAF model implemented by Seo et al. achieved an EM of 68.0 and an F1 of 77.3.

Other models that perform well include QANet, a model that does not use RNNs [2]. Instead, QANet utilizes convolution to imitate local interactions and self-attention to imitate global interactions. It runs much faster than models that use RNN, while achieving a similar level of accuracy. The model built by Yu et al. achieved a 84.6 F1 score1 on the test set, compared to the best published F1 score at the time, 81.8 [2].

Beause we hadn't seen many prior research done on the non-architectural changes mentioned in section 5.11 of the SQuAD II Default Project handout, we were inspired to explore hyper-parameter tuning and regularization. Based on results from other BiDAF papers [3], we adjusted our dropout rate and learning rate accordingly, and proceeded based on trial and error. We also attempted to explore using a different RNN, by switching out the LSTM in the baseline BiDAF for a GRU [4].

4 Approach

4.1 Baseline

The default baseline model is a simplified version of the Bi-Directional Attention Flow (BiDAF) model [1]. The original BiDAF model uses learned character-level word embeddings in addition to the word-level embeddings, as described in the default project handout. Instead, the baseline model does not use these character-level word-embeddings. The model consists of the following layers:

- Embedding layer: Performs an embedding lookup to convert the indices into word embeddings for both the context and question. These embeddings are further refined through a projection and then sent to be passed through a two-layer Highway Network [5].
- Encoder layer: Passes the output of the embedding layer into a bidirectional LSTM in order to encode temporal dependencies between timesteps.
- Attention layer: Performs bidirectional attention as described in the default project handout, specifically context-to-question and question-to-context attention.
- Modeling layer: Uses a bidirectional LSTM to refine the output from the attention layer.
- Output layer: Concatenates the outputs of the attention and modeling layers (after applying another bidirectional LSTM to the modeling layer), which is projected and passed through the softmax equation in order to produce the vector of probabilities for each position in the context.

4.2 Character-Level Embeddings

Next, we extended to include character-level embeddings in addition to word-level embeddings, matching the original BiDAF model. Accordingly, the baseline we use is that provided by the default project. As mentioned in the default project handout, the inclusion of character-level embeddings allows better handling of out-of-vocabulary words due to the conditioning on the internal structure of words learned by such embeddings.

Following the original BiDAF paper, we implemented the character embedding layer by using a 1D Convolutional Neural Network (CNN) [1]. For our current results, we set the kernel size for the CNN to 3. We set the number of channels produced by this convolution to be $\frac{H}{2}$, where H is the size of the original hidden layer (`hidden_size` in the code). We also modified the size of the output for the Linear transformation applied to the words to be $\frac{H}{2}$ as well, giving us a total embedding size of H when the character and word embedding vectors are concatenated before being sent to the

Highway Network.

Continuing to follow the BiDAF model, we apply max-pooling to the outputs of the CNN over the entire width, resulting in a fixed-size vector for each word [1]. We set the kernel size to 2, dilation to 13, and both kernel size and stride to their default values in order to satisfy the output shape desired for our implementation given the equation below, where the input to the max pooling is size (N, C, L_{in}) and output size (N, C, L_{out}):

$$L_{out} = \left\lfloor \frac{L_{in} + 2 \times \text{padding} - \text{dilation} \times (\text{kernel size} - 1) - 1}{\text{stride}} + 1 \right\rfloor$$

Finally, we reshaped the output accordingly in order to concatenate the character and word embedding vectors, then passed this concatenated embedding to the Highway Network provided [1].

4.3 Other Improvements

4.3.1 Regularization

With regard to dropout regularization, we experimented with various values of the dropout rate (0.1, 0.15, 0.2, and 0.3), which denotes the proportion of inputs that are randomly "dropped" or excluded from each update cycle.

4.3.2 Types of RNN

In addition to testing our model with a bidirectional LSTM, we also tested the model with a GRU in the encoder layer. GRU is more computationally efficient, as it does not need to use a memory unit (which the LSTM does); thus the GRU was expected to increase the speed of the model, which we confirmed with our results.

4.3.3 Model Size and Number of Layers

We experimented our model with various hidden layer sizes (76, 100, and 126) in order to test different internal model architectures.

4.3.4 Optimization Algorithms

Finally, we investigated the Adam optimizer in addition to the default Adadelta optimizer. Adam keeps a momentum, a rolling average of past gradients, lowering the variance of the model's parameter vector, and thus lowering the fluctuation of the objective function when computing gradient descent. In comparison, Adadelta uses a fixed size for the window of past gradients [6].

5 Experiments

In this section we discuss the dataset we trained our models on, the metrics we used for evaluation, and the results obtained by our models.

5.1 Data

We utilized the Stanford Question Answering Dataset 2.0 (SQuAD 2.0). SQuAD 2.0 is a reading comprehension dataset consisting of over 100,000 questions produced by Amazon Mechanical Turk workers from approximately 536 Wikipedia articles. The information in the dataset is contained in a (context, question, answer) triple, where the context is an excerpt from a Wikipedia article, the question is the question needed to be answered based on the context, and the answer is an excerpt of text from the context. One major difference in the SQuAD 2.0 compared to the original SQuAD dataset is that of the 150,000 questions in the dataset, roughly half of the questions cannot be answered using the context. This means to perform well on the SQuAD 2.0 dataset, we needed to produce a model that not only answers questions when possible, but also abstain from answer when no answer is supported by the context. [7]

- **Question:** The adaptive immune system recognizes non-self antigens during a process called what?
- **Context:** The adaptive immune system evolved in early vertebrates and allows for a stronger immune response as well as immunological memory, where each pathogen is "remembered" by a signature antigen. The adaptive immune response is antigen-specific and requires the recognition of specific "non-self" antigens during a process called antigen presentation. Antigen specificity allows for the generation of responses that are tailored to specific pathogens or pathogen-infected cells. The ability to mount these tailored responses is maintained in the body by "memory cells". Should a pathogen infect the body more than once, these specific memory cells are used to quickly eliminate it.
- **Answer:** antigen presentation
- **Prediction:** antigen presentation

Figure 1: Example of a question, context, answer triple

5.2 Evaluation method

We evaluated our results using F1 score and Exact Match (EM) evaluation. EM is a binary measurement of whether the output answer matches exactly with the ground truth answer, the answer that is recorded in the dataset. F1 scores are computed by taking the harmonic mean of precision and recall. Precision is defined by the number of correct words in the output answer divided by the number of correct words + incorrect words in the output answer, and recall is defined by the number of correct words in the output answer divided by the number of words in the ground truth answer. When evaluating, the answer output by the model has it's F1 and EM scores calculated across three human-provided answers for that question, and the maximum of these scores is used. The final reported scores are calculated by averaging the EM and F1 scores across the entire evaluation dataset.

5.3 Experimental details

We obtained a baseline through training the partially implemented BiDAF model given by the teaching staff. This baseline produced an F1 score of 58 and an EM score of 55. We planned to compare our future models against the existing score the baseline model achieved, expecting improvement seeing as we planned to extend the BiDAF baseline model by adding character embeddings and by experimenting with different hyperparameters and RNN's.

For the baseline, we made no changes to any of the default code, meaning we used default model configurations. Once we had implemented our character embedding model, we experimented with using different RNN's, optimizers, learning rates, dropout rates, and hidden sizes. Outside of these parameters we kept the other model configurations constant, such as training using 30 epochs and a batch size of 100. Training the baseline model took roughly 11 hours, while training the character embedding model took roughly 19 hours.

5.4 Results

Model	**RNN**	**Optimizer**	**LR**	**Dropout**	**Hidden Size**	**Dev NLL**	**F1**	**EM**	**AvNA**
Baseline	LSTM	Adadelta	0.5	0.2	100	3.22	60.42	56.78	67.79
CharEmb	LSTM	Adadelta	0.5	0.2	100	2.78	65.88	62.31	72.16
CharEmb	GRU	Adam	0.001	0.2	100	2.76	65.67	62.33	72.02
CharEmb	GRU	Adadelta	0.5	0.2	128	2.79	65.73	62.43	72.22
CharEmb	GRU	Adadelta	0.5	0.2	100	2.81	65.76	62.28	72.26
CharEmb	LSTM	Adadelta	0.9	0.3	100	2.57	65.21	61.77	71.53
CharEmb	LSTM	Adadelta	0.4	0.2	100	2.88	65.78	62.14	72.51
CharEmb	LSTM	Adadelta	0.5	0.1	100	3.05	66.47	62.78	73.87
CharEmb	LSTM	Adadelta	0.5	0.15	100	2.81	66.57	63.03	73.16
CharEmb	LSTM	Adadelta	0.5	0.15	76	2.78	65.46	62.09	72.09
CharEmb	LSTM	Adadelta	0.6	0.15	100	2.74	**68.02**	**64.51**	74.19
CharEmb	LSTM	Adadelta	0.65	0.15	100	2.62	66.92	63.92	72.98
CharEmb	LSTM	Adadelta	0.7	0.15	100	2.82	66.58	62.96	72.83
CharEmb	LSTM	Adadelta	0.6	0.15	126	3.04	66.41	62.93	73.21
CharEmb	GRU	Adadelta	0.6	0.15	100	2.67	66.37	62.88	72.64
Non-PCE	LSTM	Adadelta	0.5	0.2	100	-	64.08	60.37	-
Non-PCE	LSTM	Adadelta	0.6	0.15	100	-	**65.11**	**61.37**	-

Table 1: Experiment results.

Table 1 summarizes the results we found from our experiments on our models. As highlighted in the table, our best performing model on the dev set was our character-level embedding model using a bidirectional LSTM with an Adadelta optimizer, using a learning rate of 0.6, a dropout rate of 0.15, and a hidden size of 100. This model produced a F1 score of 68.017 and a EM score of 64.510 on the validation leaderboard. Testing this model on the test set and submitting to the non-PCE test leaderboard has our best model producing a F1 score of 65.106 and a EM score of 61.369.

As we had hypothesized, adding character-level embeddings to our baseline BiDAF model led to our model outperforming the baseline model in most key measurements. Specifically, the our base character-level embeddings model obtained dev scores of F1 = 65.88 and EM = 62.31, which outperforms the baseline's F1 of 60.42 and EM of 56.78. From this result, we tried to improve upon this base character-level embedding model by testing a different optimizer, namely Adam, but found that this reduced our scores and we did not want to have to try to re-optimize hyperparameters for this new optimizer, and thus continued to use Adadelta for the remaining experiments. Specifically, one of our main goals through these experiments was to systematically find the best values for our hyperparameters used in the Adadelta optimizer to try and boost our scores. This systematic approach led to us finding the sweet spot for the learning rate and drop out rate of 0.6 and 0.15 respectively. This search ultimately led to producing our best model, increasing the base character-level embeddings model F1 score by 2.13 and EM score by 2.2 on the dev set. Changing the learning learning rate and dropout rate also led to training time to decrease significantly, going from 19 hours being needed to train to roughly 12 hours being needed. Table 1 also shows our experiments using a GRU RNN over a bidirectional LSTM, which ended up reducing training time by roughly 2 hours but produced worse scores. This led to us continue using the bidirectional LSTM over a GRU since the trade-off of speeding up training but producing worse results was not worth it for us. Overall, we are content with our approach because we expected an increase in our scores by doing a systematic search for the best hyperparameters, and ended up confirming our hypothesis through producing a better model from the search.

6 Analysis

- **Question:** When had the Brotherhood renounced violence as a means of achieving its goals?
- **Context:** While Qutb's ideas became increasingly radical during his imprisonment prior to his execution in 1966, the leadership of the Brotherhood, led by Hasan al-Hudaybi, remained moderate and interested in political negotiation and activism. Fringe or splinter movements inspired by the final writings of Qutb in the mid-1960s (particularly the manifesto Milestones, a.k.a. Ma'alim fi-l-Tariq) did, however, develop and they pursued a more radical direction. By the 1970s, the Brotherhood had renounced violence as a means of achieving its goals.
- **Answer:** By the 1970s
- **Prediction:** 1970s

Figure 2: Example of a question and answer from baseline model.

- **Question:** When had the Brotherhood renounced violence as a means of achieving its goals?
- **Context:** While Qutb's ideas became increasingly radical during his imprisonment prior to his execution in 1966, the leadership of the Brotherhood, led by Hasan al-Hudaybi, remained moderate and interested in political negotiation and activism. Fringe or splinter movements inspired by the final writings of Qutb in the mid-1960s (particularly the manifesto Milestones, a.k.a. Ma'alim fi-l-Tariq) did, however, develop and they pursued a more radical direction. By the 1970s, the Brotherhood had renounced violence as a means of achieving its goals.
- **Answer:** By the 1970s
- **Prediction:** By the 1970s

Figure 3: Example of a question and answer from best BiDAF with char-embedding model.

Our BiDAF model with a character-level embedding did much better than our baseline in regards to the EM and F1 scores. This was also true for our AvNA score, showing that our best BiDAF model outperforms the baseline model for having an answer versus having no answer. Because we implemented our BiDAF with a character-level embedding, our model performed better on out

of vocabulary words and on questions that relied on identifying the morphology – or the internal structure of words - of a sentence. For example, as shown in figure 2, the baseline model incorrectly answers a question regarding when an event occurred. Our updated model was better able to identify the morphology of the sentence, and connect the time-object taken by "the Brotherhood" as the entire phrase "By the 1970s", rather than just the singular word "1970s".

- **Question:** Who designed the garden for the University Library?
- **Context:** Another important library – the University Library, founded in 1816, is home to over two million items. The building was designed by architects Marek Budzyński and Zbigniew Badowski and opened on 15 December 1999. It is surrounded by green. The University Library garden, designed by Irena Bajerska, was opened on 12 June 2002. It is one of the largest and most beautiful roof gardens in Europe with an area of more than 10,000 m2 (107,639.10 sq ft), and plants covering 5,111 m2 (55,014.35 sq ft). As the university garden it is open to the public every day.
- **Answer:** Irena Bajerska
- **Prediction:** Marek Budzyński and Zbigniew Badowski

Figure 4: Example of a question and answer with a co-reference error.

Our model falls short in regards to co-reference, unable to find expressions that refer to the same entity in a given document. For example, we see that the model understands that it must identify a person or people, who design something, in this case, a garden. However, the model answers incorrectly because it finds people who design a library, but not a garden, as the question had asked.

- **Question:** When did King Harold II conquer England?
- **Context:** In 1066, Duke William II of Normandy conquered England killing King Harold II at the Battle of Hastings. The invading Normans and their descendants replaced the Anglo-Saxons as the ruling class of England. The nobility of England were part of a single Normans culture and many had lands on both sides of the channel. Early Norman kings of England, as Dukes of Normandy, owed homage to the King of France for their land on the continent. They considered England to be their most important holding (it brought with it the title of King—an important status symbol).
- **Answer:** N/A
- **Prediction:** 1066

Figure 5: Example of a question and answer with a lack of understanding of context.

Additionally, there are examples where we see that the model does not fully understand the context in which words are being used. For example, the question in Figure 4 wants a date or year as its answer, but the answer to the question being asked is not in the context. However, our model understands that the answer must be a date or year, and so it grabs the only year present in the context because it thinks it should be the answer.

We can improve our model's performance in regards to the aforementioned errors by using self-attention and ensembling models rather than using a singular model would yield better results. Using self-attention would give results that better collects evidence from the entire passage in order to infer the correct answer. This is helpful for passages that are longer because self-attention can capture the relationship between context and query, even if the distance is large.

7 Conclusion

On our best model, we were able to achieve an F1 score of **65.106** and a EM score of **61.369** in the non-PCE division. Through this project, we have solidified our understanding of the benefit of character-level embeddings on the BiDAF model–namely, character-level embeddings allow for the conditioning on the morphology of text. We were able to effectively implement the extension to the full BiDAF model using carefully-selected parameter values when implementing the 1D convolutional neural network and max-pooling function in the Embedding layer. Additionally, through rigorous testing, we identified optimized values for a variety of parameters–learning rate, dropout rate, and hidden size–and optimized the type of RNN and optimizers used in our model. The primary limitations of our current model are co-reference errors, and errors regarding misunderstandings of word contexts. With regards to these limitations, we would especially like to continue our exploration into self-attention, in order to better answer questions that involve context-query relationships of longer distances from one another.

References

[1] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. 2018.

[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. 2018.

[3] Jingwei Ji Zibo Gong. R-net with bidaf for reading comprehension. 2019.

[4] Colin Dolese. Exploring embedding and attention improvements to bidaf squad model. 2019.

[5] Klaus Greff Rupesh Kumar Srivastava and Jürgen Schmidhuber. Highway networks. 2015.

[6] Matthew D Zeiler. Adadelta: an adaptive learning rate method. 2012.

[7] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.