A Bi-directional Attention Flow Model for the SQuAD Dataset

Cody Kala
Department of Computer Science
Stanford University
Stanford, CA 94305
codykala@stanford.edu

Horace Chu
Department of Computer Science
Stanford University
Stanford, CA 94305
horace@stanford.edu

Abstract

For our CS224n final project, we implemented a QA model for the SQuaD dataset, borrowing ideas heavily from the Bi-directional Attention Flow model introduced by Seo et al. in 2016. In the end, our model achieves an F_1 score of 71.5 and an exact match score of 60.6 on the development set (test set results are to come).

1 Introduction

Machine reading comprehension is a question answering task in which machines answer questions about passages they are given to read. Machine reading comprehension is an open area of natural language processing research with many applications in both research and industry.

Machine reading comprehension is a complex task. In order for a model to answer questions correctly, it must be able to maintain long-range dependencies, decipher context, and have some understanding of the underlying grammatical structure of the text. Until recently, the top-performing models were rule-based and required a great deal of maintenance and hand-engineered features to perform well. But the resurgence of deep learning has brought about a machine reading comprehension renaissance, with many researchers eager to incorporate deep learning architectures into their models.

Neural network based models have been very successful on the machine reading comprehension task. Within the span of a couple of years, neural network based models have quickly overtaken the rule-based models and have become the state-of-the-art. Furthermore, the advent of deep learning has lead to the discovery of novel architectures, namely recurrent neural networks and attention mechanisms.

In 2016, Rajpurkar et al. introduced the Stanford Question Answering Dataset (SQuAD) to further research in machine reading comprehension. The dataset draws on crowdworker responses to questions from over 500 Wikipedia articles and features over 100,000 question-answer pairs, making it the largest question answering dataset to date. [1]

For our final project, we built a question-answering model for the SQuAD dataset. Our model design heavily borrows from the bi-directional attention flow model introduced by Seo et al. in 2017 [2] with minor additions. We detail our question answering model for the SQuAD dataset in the following sections.

In section 2, we formally define the question answering problem on the SQuAD dataset. In section 3, we describe in detail the architecture of our model. In section 4, we discuss some of the methods we employed during training to improve our model. In section 5, we analyze and discuss our results. Finally, in section 6, we conclude with a discussion of future directions for this work.

2 Problem Definition

Given a sequence of context words $C = \{c_1, \ldots, c_M\}$ of length M and a sequence of question words $Q = \{q_1, \ldots, q_N\}$ of length N, we seek to learn a model $f(C, Q) = (i_s, i_e)$, where $1 \leq i_s \leq i_e \leq M$ denote indices into the context C that span the answer to the question Q.

3 Model

Our model consists of five integral parts. We will describe these parts in detail in addition to other features that we have decided not to include in our final project.

3.1 GloVe Word Embeddings

We use pretrained GloVe vectors to represent the words in the context and question. Empirically, we found that model performance did not vary appreciably with the size of the embeddings, so we defaulted to use size $h = 100$ embeddings. That is, $c_i, q_i \in \mathbb{R}^h$.

3.2 Encoder Layer

The encoder layer consists of a bi-directional GRU that encodes the GloVE vectors for each context and question word.

$$\{\overrightarrow{c_1}, \overrightarrow{c_1}, \ldots, \overleftarrow{c_M}, \overrightarrow{c_M}\} = \text{Bi-GRU}\left(\{c_1, \ldots, c_M\}\right) \tag{1}$$

$$\{\overrightarrow{q_1}, \overrightarrow{q_1}, \ldots, \overleftarrow{q_N}, \overrightarrow{q_N}\} = \text{Bi-GRU}\left(\{q_1, \ldots, q_N\}\right) \tag{2}$$

We concatenate the forward and backward hidden states to form the context and question hidden states:

$$\tilde{c}_i = [\overrightarrow{c_i}; \overleftarrow{c_i}] \in \mathbb{R}^{2h}, \quad \forall i \in \{1, \ldots, M\} \tag{3}$$

$$\tilde{q}_i = [\overrightarrow{q_i}; \overleftarrow{q_i}] \in \mathbb{R}^{2h}, \quad \forall i \in \{1, \ldots, N\} \tag{4}$$

The weights and biases of the Bi-GRU models are shared for the context and question hidden states and are learned. The context and question hidden states are the final outputs of the encoder layer.

3.3 Attention Layer

The attention layer is a bi-directional attention flow layer, first by Seo et al. in 2017 [2]. First, a similarity matrix $S \in \mathbb{R}^{M \times N}$ is computed according to

$$S_{ij} = w_{\text{sim}}^T \left[\tilde{c}_i; \tilde{q}_j; \tilde{c}_i \circ \tilde{q}_j\right], \tag{5}$$

where $w_{\text{sim}} \in \mathbb{R}^{6h}$ are learned weights. Using the similarity matrix, we compute the context-to-question (C2Q) and question-to-context (Q2C) attention outputs. The C2Q outputs are computed according to

$$\alpha^{(i)} = \text{softmax}\left(S_{i,:}\right) \in \mathbb{R}^N \tag{6}$$

$$a_i = \sum_{j=1}^{N} \alpha_j^{(i)} q_j \in \mathbb{R}^{2h} \tag{7}$$

for all $i \in \{1, \ldots, M\}$. The Q2C attention outputs are computed according to

$$m_i = \max_j (S_{ij}) \in \mathbb{R}, \quad \forall i \in \{1, \ldots, N\} \tag{8}$$

$$\beta = \text{softmax}\,(m) \in \mathbb{R}^M \tag{9}$$

$$c' = \sum_{i=1}^{M} \beta_i c_i \in \mathbb{R}^{2h} \tag{10}$$

From these outputs, we form the blended representations

$$b_i = [c_i; a_i; c_i \circ a_i; c_i \circ c'] \in \mathbb{R}^{8h}, \quad \forall i \in \{1, \ldots, M\} \tag{11}$$

The blended representations are the final outputs of the attention layer.

3.4 Modeling Layer

The modeling layer consists of two Bi-LSTM layers that perform a down-projection of the blended representations to produce hidden states $e_i \in \mathbb{R}^{2h}$ for all $i \in \{1, \ldots, M\}$.

$$\{\overrightarrow{b_1}, \overleftarrow{b_1}, \ldots, \overrightarrow{b_M}, \overleftarrow{b_M}\} = \text{Bi-LSTM}\,(\{b_1, \ldots, b_M\}) \tag{12}$$

$$d_i = \left[\overrightarrow{b_i}; \overleftarrow{b_i}\right] \in \mathbb{R}^{2h}, \quad \forall i \in \{1, \ldots, M\} \tag{13}$$

$$\{\overrightarrow{d_1}, \overleftarrow{d_1}, \ldots, \overrightarrow{d_M}, \overleftarrow{d_M}\} = \text{Bi-LSTM}\,(\{d_1, \ldots, d_M\}) \tag{14}$$

$$e_i = \left[\overrightarrow{d_i}; \overleftarrow{d_i}\right] \in \mathbb{R}^{2h}, \forall i \in \{1, \ldots, M\} \tag{15}$$

The choice to down-project to hidden states in $\mathbb{R}^h$ in the first Bi-LSTM is arbitrary. The weights and biases of the two Bi-LSTM models are not shared, but they are both learned. The hidden states e_i are the final output of the modeling layer.

3.5 Output Layer

The output layer takes the hidden states from the attention and modeling layers and combines them to calculate the probability distributions for the start and end positions of the answer span. Letting $B \in \mathbb{R}^{M \times 8h}$ and $E \in \mathbb{R}^{M \times 2h}$, the probability distributions for the start and end positions are computed as follows:

$$p_{\text{start}} = \text{softmax}\left(w_{\text{start}}^T [B; E]\right) \tag{16}$$

$$E_2 = \text{Bi-LSTM}\,(E) \tag{17}$$

$$p_{\text{end}} = \text{softmax}\left(w_{\text{end}}^T [B; E_2]\right) \tag{18}$$

The weights $w_{\text{start}}, w_{\text{end}} \in \mathbb{R}^{10h}$ are learned by the model during training.

3.6 Character Convolutional Neural Network

In addition to using pre-trained GloVe word embeddings, we also used character embeddings generated from a convolutional neural network. In theory, this will allow us to condition on the internal structure of words and deal better with out-of-vocabulary words. [3]

Before describing the model, we first define V as the vocabulary size, d as the dimension of the character embeddings and $Q \in \mathbb{R}^{d*V}$ as the matrix of character embeddings. For each word k in the vocabulary with l characters, the character embedding for that word will be $C^k \in \mathbb{R}^{d*l}$.

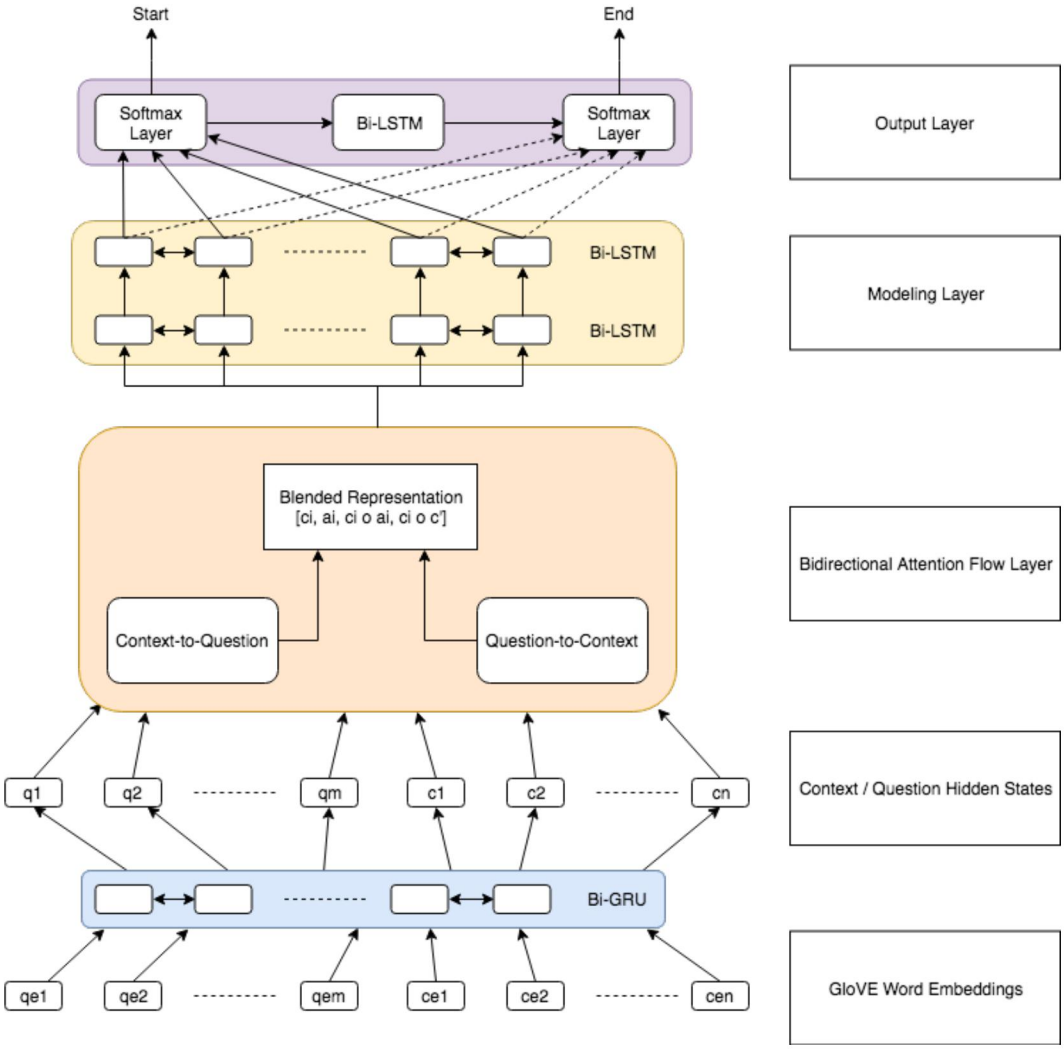

Figure 1: Architecture of our model.

We apply a narrow convolution between C^k and a filter $H \in \mathbb{R}^{d*w}$ of width w, before adding a bias term and a non-linearity layer to obtain a feature map $f^k \in \mathbb{R}^{l-w+1}$, where f^k is defined as

$$f^k[i] = \tanh \langle C^k[*, i : i + w - 1], H \rangle + b \tag{19}$$

where $C^k[*, i : i + w - 1]$ is the i-to-$i + w - 1$ column of C^k and $\langle A, B \rangle$ is the Frobenius inner product of A and B. Afterwards, we take the max over time

$$y^k = \max_i \left\{ f^k[i] \right\} \tag{20}$$

which returns the feature representing the filter H. This captures the most important feature for a given filter, and we will repeat the process n times to output a matrix with an n-dimension CNN embedding. This matrix will then be concatenated with the pre-trained word embeddings, and since we have used a word embedding size of 100, the resulting matrix will have an embedding size of $100 + n$. [4]

Model	F1 Score	EM Score
Baseline	44.0	34.8
BiDAF (ours)	71.5	60.6
BiDAF (Seo et al., 2016)	77.3	67.7
Dynamic Coattention (Xiong et al., 2016)	75.6	65.4
Dynamic Coattention Ensemble (Xiong et al., 2016)	79.4	70.3
Human Performance	91.0	81.4

Figure 2: Dev performance comparison with other models.

3.7 Additional Input Features

In addition to character CNNs, there are a number of input features that have the potential to dramatically increase the accuracy and reliability of the model. One particular input feature that we explored is the exact match input feature.

This is how the exact match input feature works: given all the context tokens in the context, we can add an additional boolean feature that has value 1 if the context token appears in the question and value 0 if it doesn't. The rationale behind this is that a context token is more likely to be included in the final answer if it also appears in the question.

4 Training Details

4.1 Padding

As described above, we use the same Bi-GRU to encode the GloVe embeddings for the context and question. Since the Bi-GRU expects inputs to be the same size, we included padding so that the number of question word embeddings matched the number of context word embeddings.

4.2 Hyperparameter Searching

To determine the best hyperparameter settings, we performed a greedy search on the hyperparameter space. Specifically, we tuned each hyperparameter with all of the other hyperparameters held fixed and used the best hyperparameters found. To conserve time and resources, we only allowed runs to continue to about 3k-4k iterations before cutting them off. We performed greedy searches on the learning rate, batch size, hidden size, regularization constant, and dropout rate.

4.3 Avoiding Overfitting

To avoid overfitting in our models, we experimented with different dropout rates and L_2-regularization.

4.4 Choosing an Optimizer

We experimented with using Adam, AdaDelta, and SGD to optimize the learning models. Keskar and Socher showed that switching from Adam to SGD at later stages of learning can improve generalization error [5], though we did not observe this in our experiments. Seo et al. used AdaDelta to optimize their learning models with an annealing learning schedule. We experimented with this idea for some time, but eventually backed off to using Adam with a fixed learning rate after noticing that training took much longer with AdaDelta than with Adam.

5 Final Results

5.1 Quantitative Analysis

See figure 2 for a comparison of our model's performance to other start-of-the-art models. On the development set, our model achieved an F_1 score of 71.5 and an exact match score of 60.6 using

a batch size of 100, hidden size of 125, learning rate of 0.001, dropout rate of 0.15, regularization constant of 0.1, and a maximum context length of 400. We optimized our model using Adam for 12k iterations.

For plots of the F_1 and exact match scores for various hyperparameter settings, see Figures 3, 4, and 5 in the Supplemental Materials section at the end of the report.

5.2 Qualitative Analysis

Generating examples for random question-answer pairs in the development set, we noticed some shortcomings of our model.

5.2.1 Misplaced Attention

On some examples, we saw that our model paid attention to the wrong parts of the context. Consider the following example:

Example:

- **Context**: Southern California is home to many major business districts. Central business districts (cbd) include Downtown Los Angeles, Downtown San Diego, Downtown San Bernardino, Downtown Bakersfield, South Coast Metro and Downtown Riverside.
- **Question**: What is the only district in the cbd to not have "downtown" in its name?
- **Prediction**: central business districts
- **Ground truth**: South Coast Metro

Here, we see that the model likely placed a lot of attention on the words "district" and "cbd". The model likely fails for question-answer pairs like this one because no part of the ground truth appears in the question, so it has a difficult time finding paying attention to relevant parts of the context.

5.2.2 Model lacks background knowledge

Unsurprisingly, the model does not perform well on question-answer pairs for which the answer cannot be determined from the context paragraph alone. Consider the following example.

Example

- **Context**: Not only are all the major British architects of the last four hundred years represented, but many European (especially Italian) and American architects' drawings are held in the collection. The Riba's holdings of over 330 drawings by Andrea Palladio are the largest in the world, other Europeans well represented are Jacques Gentilhatre and Antonio Visentini. British architects whose drawings, and in some cases models of their buildings, in the collection, include: Inigo Jones, Sir Christopher Wren, Sir John Vanbrugh, Nicholas Hawksmoor, William Kent, James Gibbs, Robert Adam, Sir William Chambers, James Wyatt, Henry Holland, John Nash, Sir John Soane, Sir Charles Barry, Charles Robert Cockerell, Augustus Welby Northmore Pugin, Sir George Gilbert Scott, John Loughborough Pearson, George Edmund Street, Richard Norman Shaw, Alfred Waterhouse, Sir Edwin Lutyens, Charles Rennie Mackintosh, Charles Holden, Frank Hoar, Lord Richard Rogers, Lord Norman Foster, sir Nicholas Grimshaw, Zaha Hadid and Alick Horsnell.
- **Question**: Which architect, famous for designing London's St. Paul Cathedral, is represented in the Riba collection?
- **Prediction**: Andrea Palladio
- **Ground Truth**: Sir Christopher Wren

Nowhere in the context paragraph does it describe Sir Christopher Wren designing London's St. Paul Cathedral, so the only information the model is able to gleam from the question is the fact that the architect is represented in the Riba collection. The model pays attention to the word "Riba" and stumbles upon the name "Andrea Palladio", which appears closest to "Riba".

This is a difficult for any model to solve, and even humans are likely to get this one wrong if they don't know the history of London's St. Paul Cathedral. Ways to address the issue would be to give the model access to a large textual database (perhaps an encyclopedia) from which it can gather additional background information.

5.2.3 Attention is too narrow

Sometimes, the model pays attention to only a part of the ground truth, such as in the following example.

- **Context**: To remedy the causes of the fire, changes were made in the Block II spacecraft and operational procedures, the most important of which were use of a nitrogen/oxygen mixture instead of pure oxygen before and during launch, and removal of flammable cabin and space suit materials. The Block II design already called for replacement of the Block I plug-type hatch cover with a quick-release, outward opening door. NASA discontinued the manned Block I program, using the Block I spacecraft only for unmanned Saturn V flights. Crew members would also exclusively wear modified, fire-resistant Block II space suits, and would be designated by the Block II titles, regardless of whether a LM was present on the flight or not.
- **Question**: What type of materials inside the cabin were removed to help prevent more fire hazards in the future?
- **Prediction**: space suit
- **Ground truth**: flammable cabin and space suit materials

In this example, the model likely pays most of its attention to "space suit" due to its close proximity to the word "materials," which appears in the context.

6 Conclusion

Overall in this project, we were able to achieve solid results by training with a bidirectional attention flow layer and its corresponding modelling layer, in addition to several other features. In this section, we will detail some of the most important observations we've made during this process and what we've learnt from those experiences.

Through working on this project, we realized that ensembling is a powerful tool in improving performance, but the computational costs are steep. Having first implemented bidirectional attention flow with a modelling layer, we also implemented co-attention in the hope that combining the two models would give us better results. However, we realized that training our model with both of these models would dramatically increase the number of parameters we have to train and hence the training time. As a result, we decided to follow the procedure used in Kim et al. and stick with bidirectional attention.

We also realized that getting multiple models to work well with each other is particularly difficult, since there are naturally more hyperparameters we have to train, and finding the optimal set of hyperparameters for our model takes a lot more trial and error. In some cases, like we had with the original bidirectional attention flow layer and character CNN, it was so difficult to find the optimal set of hyperparameters to improve the model that we decided to ignore the character CNN completely and focus on the hyperparameters for the bidirectional attention flow layer alone. We learnt that sometimes less is more and optimizing more simple models as much as possible could often give better results than more complicated and gnarly models.

In that vein, it is clear from this project that the most significant improvements often come from the simplest ideas and concepts. While the exact match and modelling layer were relatively straightforward to implement, they gave us a massive boost in accuracy scores. On the other hand, more complicated features like co-attention and character CNN didn't give nearly the same boost to our model. As a result, this project has taught us to look for the simplest improvements with the biggest impact.

7 Future Work

Due to time constraints, we were not able to implement all of the features that we had originally planned to, but we include below what we would have done given more time.

We attempted to incorporate a character CNN into our model to augment the pre-trained GloVe embeddings, but we ran into time constraints and were not able to fully integrate this feature into the model.

In addition to the more complicated feature models, there are also many small features we did not have a chance to incorporate into our model, such as exact match features and checking that the start position occurs before the end position. These would certainly lead to improved efficiency and performance.

Combining intuitions from other neural network based attention models, such as the Dynamic Coattention model, would likely result in improved performance. Given additional time and resources, we would create an ensemble model that uses BiDAF and Dynamic Coattention models and uses a majority-vote to make predictions.

8 References

[1] SQuAD: 100,000+ Questions for Machine Comprehension of Text. Rajpurkar et al. June 2016.

[2] Dynamic Coattention Networks For Question Answering. Xiong et al. November 2016.

[3] CS 224N Default Final Project: Question Answering.

[4] Character-Aware Neural Language Models. Kim et al. December 2015.

[5] Improving Generalization Performance by Switching from Adam to SGD. Keskar and Socher. December 2017.

[6] Bidirectional Attention Flow for Machine Comprehension

9 Supplemental Materials

9.1 Hyperparameter search plots

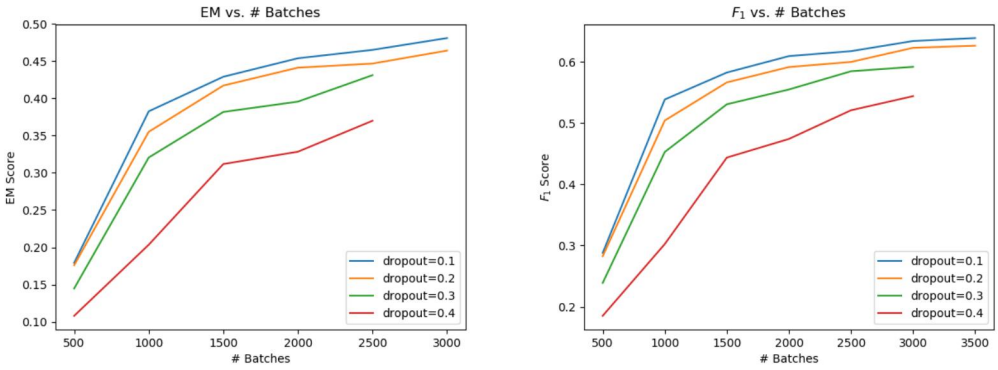

Figure 3: EM and F1 scores for varying dropout rates.

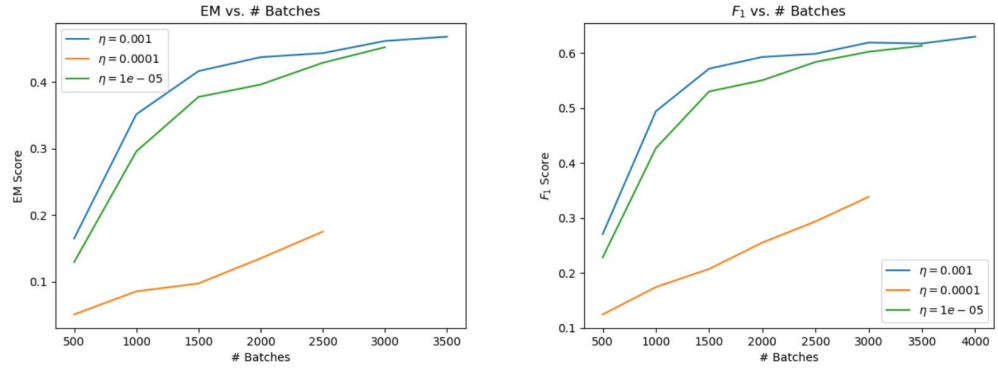

Figure 4: EM and F1 scores for varying learning rates.

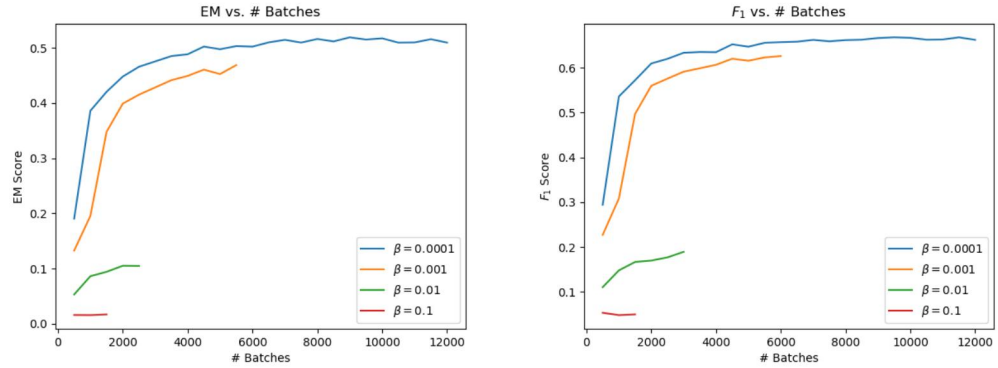

Figure 5: EM and F1 scores for varying regularization constants.