Machine Comprehension on SQuAD using Bi-Directional Attention Flow

Daisy Ding, Ruohan Zhan
Institute for Computational and Mathematical Engineering
Stanford University
{dingd, rhzhan}@stanford.edu

Abstract

This project applies deep learning with the Bi-Directional Attention Flow (BiDAF) network to train a model for the machine comprehension task on the Stanford Question Answering Dataset (SQuAD). We implement the BiDAF model that represents the context with character-level, word-level, and contextual-level embeddings and utilizes the bi-directional attention flow to capture the interactions between context and query. We experiment with exponential moving average and conduct hyperparameter tuning. During the evaluation stage, we make the answer-span prediction by searching the pair of start and end positions with the highest joint probability. Our single model achieves competitive results of 75.594% F1 score and 65.299% EM on the test set.

1 Introduction

Machine comprehension (MC) of text is a challenging task that aims to extract information from a given context in response to a given query. As a benchmark task to demonstrate natural language understanding, machine comprehension has gained significant popularity within the natural language processing community. The importance of this task also closely relates to its wide applications. Various industries are paying close attention to artificial intelligent agents with machine comprehension capability, and have applied the cutting edge technology to their current service systems, from legal service support to financial trading.

Since the release of the Stanford Question Answering Dataset (SQuAD)[1] in 2016, rapid progress has been made on training end-to-end models for the machine comprehension task. In particular, neural attention, a mechanism that allows the system to focus on more relevant parts of the information, has led to major advancements. Among these works, the Bi-Directional Attention Flow (BiDAF) model introduced by Seo et. al.[2] achieved benchmark scores on the SQuAD leaderboard, and since then has attracted certain research attention.

In this project, we re-implement the BiDAF network, including the character and word embedding layers, the contextual embedding layer, the bi-directional attention flow layers, the LSTM modeling layers, and the softmax output layer. We experiment with the exponential moving average technique and conduct hyper-parameter tuning. During the evaluation stage, we make the start and end positions prediction by searching the pair of the start and end positions with the highest joint probability. Our single model achieves competitive results of 75.594% F1 score and 65.299% EM on the test set, and 75.295% F1 score and 64.702% EM on the development set. We also provide an qualitative analysis on the model's performance by illustrating several question-answer pairs and visualizing the attention mechanism. Al last, motivated by the error analysis, we identify the current model's limitations and suggest several directions for further research.

2 Related Works

Before the end-to-end neural network has gained its popularity in solving machine comprehension task, the task was mainly approached by feature-based systems. Richardson et. al[3] introduce a model features including bag-of-word and word distance. Smith et al.[4] apply augmented bag-of-word features. As for deep learning methods, in 2015, Hermann et. al[5] embed sentences into vector space to train LSTMs. When it comes to neural attention mechanism, Wang and Jiang[6] utilize match-LSTM to account for the different importance of words in a passage corresponding to a given query. Xiong et al.[7] propose dynamic co-attention that involves a two-way attention between the context and the question. Wang et al[8] match the context against the question from multiple perspectives.

3 Data

3.1 Dataset Description

Stanford Question Answering Dataset (SQuAD) consists of 100,000+ question-answer pairs on 500+ Wikipedia articles where the answer to every question is a span taken from the corresponding context[1]. To better understand our dataset, we examine the data by plotting the lengths of context, question and answer, as well as the word length in each sentence from the dev set (Figure 1). From the length distribution, we identify the best choices for the hyperparameters *max-context-length, max-question-length, max-answer-length, max-word-length*. We also examine the distribution of different question types characterized by "what", "who", "how", "when", "which", "where", and "why" (Figure 2). As shown in the graph, the "what" type of question accounts for the largest proportion in the dataset.

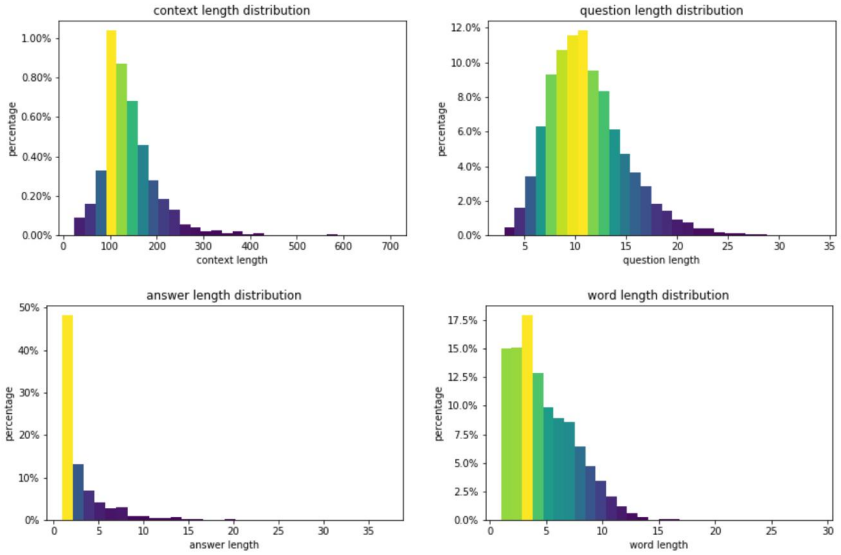

Figure 1: Development dataset length analysis

3.2 Problem Formulation

Machine comprehension of text typically contains a context **C** and one or many question **Q**, AND requires the model to apply information form **C** to predict answer **A** to its corresponding question **Q**. For SQuAD dataset, questions are problems without multiple choices, and the answers **A** are continuous sub-spans of **C**. Thus, the task is to search for direct answer excerpts from **C** instead of synthezing answers.

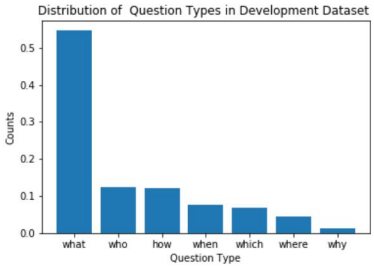

Figure 2: Distribution of different questions

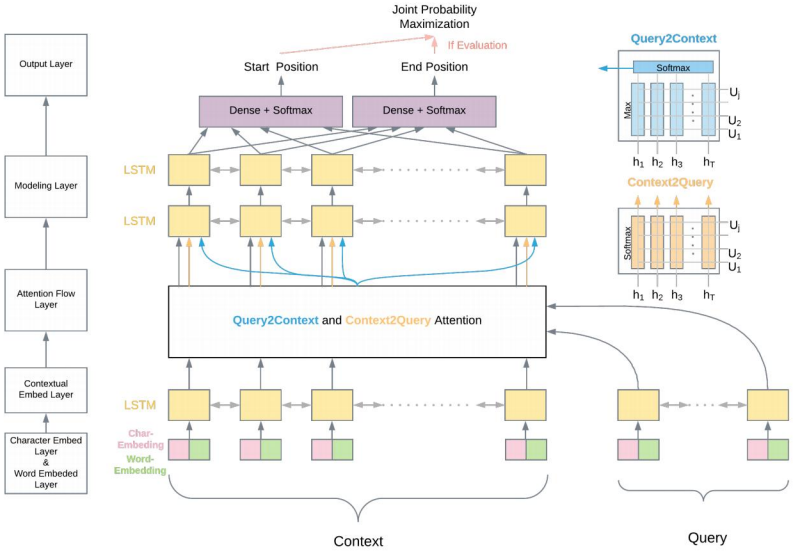

Figure 3: BiDirectional Attention Flow Model Architecture

4 Model

4.1 Model Architecture

Our model has five components: **embedding layer**, which embeds both context and question words; **RNN encoder layer**, which encodes embedded words into hidden states; **attention layer**, which combines context and question representations, **modeling layer**, which encodes context with the information of question, **output layer**, which gives predicted distribution for answer's start and end positions. Below we demonstrate each of the five components.

Embedding Layer

- **Character Embedding** Each word is padded to have the same length W, and each character is embeded into a vector of length L_i. Note here the embeded character vectors are also model parameters. We firstly represent a word with a matrix of size $W \times L_i$. Then we use one-dimensional convolutional neural network with L_i in-channels and L_o out-channels to get a matrix of size $d_c \times L_o$. Finally we take the maximum along out-channels and represent each word with a character embedding vector of length d_c.
- **Word Embedding** Pre-trained GloVe vectors are used with embeded size $d_w = 100$.

We concatenate the character and word embeddings vectors to represent each word in contexts and questions.

RNN Encoder Layer Sequences in contexts and questions are padded to have the same lengths N and M. The embeded sequences are $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and $\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_M\}$ for contexts and questions respectively, and are fed into a 1-layer bidirectional LSTM to acquire the hidden states.

$$\{\overrightarrow{\mathbf{c}_1}, \overleftarrow{\mathbf{c}_1}, \ldots, \overrightarrow{\mathbf{c}_N}, \overleftarrow{\mathbf{c}_N}\} = LSTM(\{\mathbf{x}_1, \ldots, \mathbf{x}_N\})$$
$$\{\overrightarrow{\mathbf{q}_1}, \overleftarrow{\mathbf{q}_1}, \ldots, \overrightarrow{\mathbf{q}_M}, \overleftarrow{\mathbf{q}_M}\} = LSTM(\{\mathbf{y}_1, \ldots, \mathbf{y}_N\})$$

where $\overrightarrow{\mathbf{c}_i}, \overrightarrow{\mathbf{q}_j}$ are forward hidden states of contexts and questions and $\overleftarrow{\mathbf{c}_i}, \overleftarrow{\mathbf{q}_j}$ are backward hidden states of contexts and questions. The output of this layer is the concatenation of forward and backward hidden states.

$$\mathbf{c}_i = [\overrightarrow{\mathbf{c}_i}; \overleftarrow{\mathbf{c}_i}], \mathbf{q}_j = [\overrightarrow{\mathbf{q}_j}; \overleftarrow{\mathbf{q}_j}].$$

Attention Layer We first calculate the similarity matrix A and then get the context-to-question attention and question-to-context attention.

- **Similarity Matrix** $A_{i,j} = Attn(\mathbf{c}_i, \mathbf{q}_j) = \mathbf{w}_a[\mathbf{c}_i, \mathbf{q}_j, \mathbf{c}_i \circ \mathbf{q}_j]$ where $\mathbf{w}_a \in \mathbf{R}^{3d}$ is a similarity weight vector.
- **Context-to-Question Attention** This attention allows us to get a question-aware representation of each individual context word. For every context word i, we get a similarity distribution with respect to question words: $D_i = softmax(A[i,]) \in \mathbf{R}^N$. Then the question-aware representation of context word i is $\tilde{\mathbf{c}}_i = \sum_j D_i(j)\mathbf{q}_j \in \mathbf{R}^d$.
- **Question-to-Context Attention** This attention allows us to get a question-aware representation of the whole context. We firstly take the maximum similarity with question for each individual context word i: $\mathbf{s} = \{s_i\} = \{\max(A[i,])\} \in \mathbf{R}^N$, which is later softmaxed to get a distribution $D_c = softmax\mathbf{s}$. Finally the whole context can be represented as a vector $\bar{\mathbf{c}} = \sum_i D_c(i)\mathbf{c}_i \in \mathbf{R}^d$, the weighted sum of context words based on its relevancy with question.

We get a blended representation by concatenating the above vectors:

$$\mathbf{a}_i = [\mathbf{c}_i; \tilde{\mathbf{c}}_i; \bar{\mathbf{c}}; \bar{\mathbf{c}} \circ \mathbf{c}_i]$$

Modeling Layer The blended representations of context are fed into a 2-layer bidirectional LSTM network to encode the context with the information of question. Similarly for the RNN Encoder layer, we concatenate the forward hidden states and backward hidden states.

$$\{\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_N\} = LSTM(\{\mathbf{a}_1, \ldots, \mathbf{a}_N\})$$
$$\{\mathbf{v}_1, \ldots, \mathbf{v}_N\} = LSTM(\{\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_N\})$$

The final blended representations are as follows:

$$\mathbf{b}_i = [\mathbf{a}_i; \mathbf{v}_i].$$

Output Layer In this layer we output two distributions for the start location and end location in the contexts. Feed the final blended representation $\mathbf{b}_i$ into a fully connected layer to score the context location i, and then get a distribution.

$$score_{start}(i) = \mathbf{w}_{start}^T \mathbf{b}_i + u_{start}, \quad score_{end}(i) = \mathbf{w}_{end}^T \mathbf{b}_i + u_{end}$$
$$p_{start} = softmax(score_{start}) \in \mathbf{R}^N, \quad p_{end} = softmax(score_{end}) \in \mathbf{R}^N.$$

Loss Cross entropy loss is used here to characterize our model accuracy. If the ground truth locations are $l^{start}, l^{end} \in [N]$, then the loss is defined as follows:

$$loss = -\log p_{start}(l^{start}) - \log p_{end}(l^{end})$$

Prediction We make predictions by maximizing the joint probability of $p_{start}(i)p_{end}(j)$. Besides, we use the heuristic that $0 \leq l_{end} - l_{start} \leq len_{answer}$, where len_{answer} is a predefined model parameter that determines that longest answer length. Therefore, the predicted span (pl^{start}, pl^{end}) is chosen by:

$$(pl^{start}, pl^{end}) = argmax_{(i,j),0 \leq j-i \leq len_{answer}} p_{start}(i)p_{end}(j).$$

which can be calculated linearly by going through all feasible (i, j) locations.

Model	Dev Set		Test Set	
	F1	**EM**	**F1**	**EM**
Baseline	43.6	34.3	-	-
OurBiDAF1	72.4	62.2	-	-
OurBiDAF2, joint-pred	73.7	62.7	-	-
OurBiDAF3, joint-pred, char-embed	75.3	64.7	75.6	65.3
BiDAF	-	-	77.3	68.0
Dynamic Coattention	-	-	75.9	66.2
r-net	-	-	68.4	77.5
Human Performance	-	-	91.2	82.3

Table 1: Result comparison among different models (single) on Dev and Test set

4.2 Training Details

We train 12 epochs with the initial learning rate 0.003, dropout rate 0.2, and batch size 36. We also decrease the learning rate by a factor of 10 after the 5th epoch. We have also experimented with exponential moving average, which, however, does not improve the model's performance.

5 Results and Analysis

5.1 Results

Our model achieves F1 score **75.29** and EM **64.70** on the development set, and F1 score **75.59** and EM **65.30** on the test set. We provide a comparison of our models with the state-of-art models in (Table 1), in which **OurBiDAF1** refers to the use of bidirectional attention flow, **joint-pred** refers that we do predictions by maximizing the joint start-end position probabilities and **char-embed** refers to the use of character-level embedding.

This table also shows the incremental improvement of each implementation. The implementation of the bi-directional attention flow gives the most significant improvement on both F1 and EM. The joint probability maximization method improves the F1 score by 1.3 point by tackling the situation when the probability of independent position maximization is high but the joint probability is low. For example, before we implement the joint probability prediction technique, the model could give predictions where the end position is before the start position. As for the character embedding, this method handles the out-of-vocabulary words and increases the F1 score by 1.6 point.

Below we show an example of context-question-answer triplet, in which the model is able to produce the correct answer to a complex question, which further demonstrates the model's effectiveness.

Example 1:

- **Context**: In time, Kublai Khan 's successors lost all influence on other mongol lands across asia, while **the mongols beyond the middle kingdom saw them as too chinese**. Gradually, they lost influence in china as well... Uninterested in administration, they were separated from both the army and the populace, and china was torn by dissension and unrest.
- **Question**: Why did Kublai 's successors lose control of the rest of the mongol empire?
- **Prediction**: The mongols beyond the middle kingdom saw them as too chinese
- **Answer**: The mongols beyond the middle kingdom saw them as too chinese

5.2 Attention Visualization

We visualize the similarity matrices for context-to-query attention in figure 4. Note that the two examples have the same context but different questions. The red boxes mark the true answer positions.

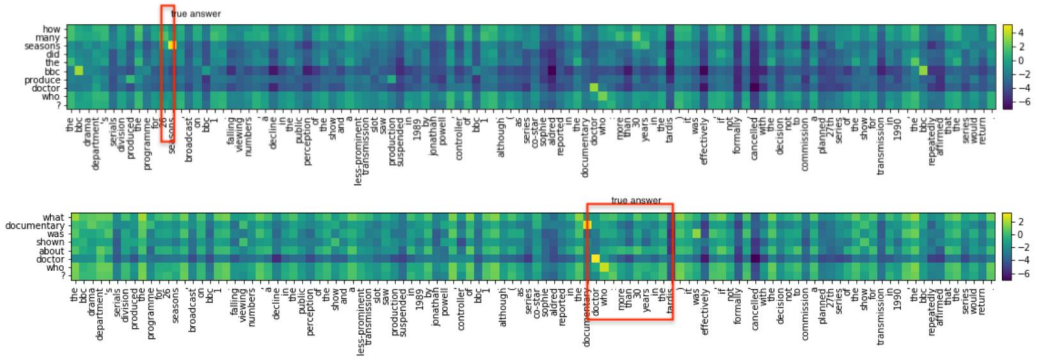

Figure 4: Attention Visualization

Question Type	F1	EM
What	64.23	49.53
When	81.28	68.32
How	75.91	54.20
Who	74.82	67.05
Where	64.06	64.10
Which	75.29	50.58
Why	53.63	17.30

Table 2: Influence of Question Type on Dev-set Accuracy

As they are consistent with the brightest attention logits in the similarity matrices, the visualization illustrates that our model is able to effectively attend to the correct locations of the context when given different queries.

5.3 Model Performance by Attributes

5.3.1 Question Type

We examine our model accuracy on the development set, and calculate the average F1 and EM scores for seven different question types: **What, When, How, Who, Where, Which, Why**. Table 2 shows that our model has better performances on **When** and **Who**, while performs poorly on **Why** questions. This could be explained by the fact that **Why** questions require a deeper level of context comprehension, as the relationship between context and query is not as direct as those of other types of questions.

5.3.2 Context/Question/Answer Length

We also examine our model's performance with respect to context/question/answer length. As seen in Figure 5, we find that the context and answer lengths do not exert significant influence on the predictive accuracy, while for the answer length, the longer the true answer is the lower the evaluation metrics are for our model. Similar to the analysis of question type, the queries with long answers are often more complicated and therefore require the model to gain a deeper comprehension of the context and characterize a longer term of dependency.

5.4 Error Analysis

We provide a qualitative error analysis by comparing some incorrect prediction to the ground truth. We find that the errors can mainly be categorized into the following types.

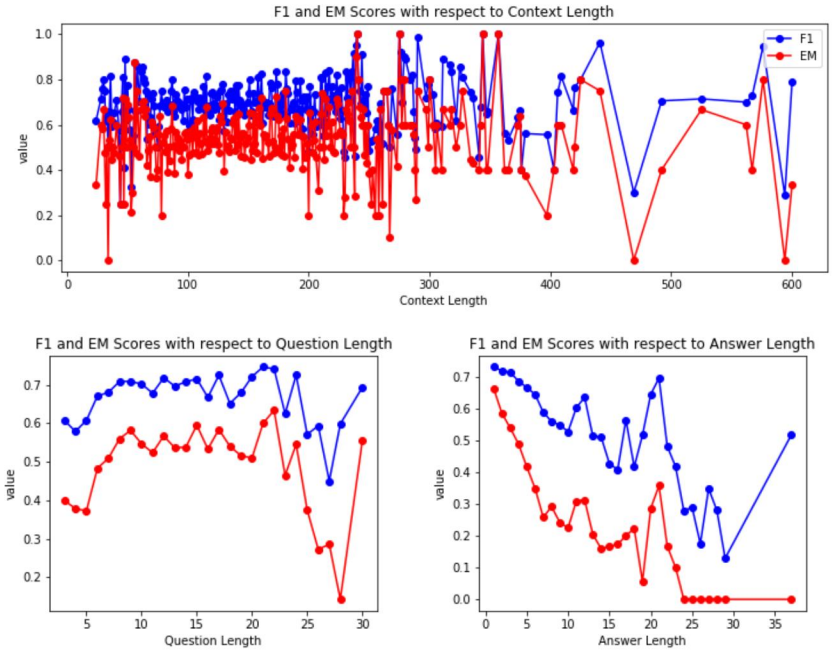

Figure 5: Influence of Context/Question/Answer Length on Model Accuracy in the Development Set

5.4.1 Long Predicted Answers

Example 2:

- **Predicted Answer**: Many governments also wanted to increase their independence and strengthen legislation to limit foreign ownership of broadcasting properties
- **True Answer**: Legislation to limit foreign ownership of broadcasting properties

Example 2 shows a common problem of our model that the prediction is longer than the true answer but with roughly the correct position. On one hand, it shows that our model is able to abstract and extract context information and correctly locate the answer when given a query. On the other hand, it also indicates that the model does not have the full capability to understand the problem, since the redundant parts either show no logical connection with the query or do not contribute to the query to a certain extent.

5.4.2 Imprecise Answer Boundaries

Example 3:

- **Predicted Answer**: Desire to encourage consensus amongst elected members
- **True Answer**: Encourage consensus amongst elected members

This type of error is similar to the one shown in Example 2 but is less severe. One possible reason is similar to that of the Long Predicted Answers error - the model does not have good enough understanding of the context and the question. However, this problem could also be caused by personal preferences of human testers who provide the question answers. And it is difficult for a model to capture this type of nuanced difference.

5.4.3 Same Type Wrong Answer

Example 4:

- **Context**: The revived series has received recognition from critics and the public, across various awards ceremonies. It won five bafta tv awards, including best drama series, the highest-profile and most prestigious **british television award** for which the series has ever been nominated.
- **Question**: What was the most revered award that doctor who has won?
- **Prediction**: British television award
- **Answer**: Best drama series

This type of error might be caused by the mechanism of attention. Note that the predicted answer and the true answer are both sound and serve the same functionality as they are of the same type as short phrases. However, it is more likely for attention to focus on "British television award" instead of "best drama series", because "most prestigious" and "award" in context are more similar to "most revered award" in the question while "best drama series" are farther.

6 Conclusion and Future Studies

In this project, we implement the BiDAF model that utilizes multiple-level embeddings and the bi-directional attention flow with both context-to-query and query-to-context. We modify the prediction mechanism during the evaluation stage by searching the pair of start and end positions with the highest joint probability. Our single model achieves competitive results of 75.594% F1 score and 65.299% EM on the test set, and is able to attend to the correct context locations given queries.

Motivated by the error analysis, we suggest several directions of further research. Firstly, even though the model is able to extract relevant information from contexts in order to answer questions, it does not perform as well when deeper context comprehension or logical reasoning are required. Therefore, in order to tackle more complicated context-query pairs, we could incorporate multiple hops of the attention layer. Alternatively, we could apply iterative reasoning - conduct reasoning multiple times to enhance context comprehension. In addition, we could use enhanced gated mechanism in order to improve the longer-term dependencies. Moreover, we could improve the loss function by combining the F1 and EM scores to the cross-entropy loss, thus establishing a more direct connection between the loss function and the evaluation metrics. From this line of thought, we could incorporate the idea of reinforcement learning in solving the machine comprehension task.

References

[1] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100, 000+ questions for machine comprehension of text," *CoRR*, vol. abs/1606.05250, 2016.

[2] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," *CoRR*, vol. abs/1611.01603, 2016.

[3] M. Richardson, "Mctest: A challenge dataset for the open-domain machine comprehension of text," October 2013.

[4] E. Smith, N. Greco, M. Bosnjak, and A. Vlachos, "A strong lexical matching method for the machine comprehension test," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 1693–1698, Association for Computational Linguistics, September 2015.

[5] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, (Cambridge, MA, USA), pp. 1693–1701, MIT Press, 2015.

[6] S. Wang and J. Jiang, "Machine comprehension using match-lstm and answer pointer," *CoRR*, vol. abs/1608.07905, 2016.

[7] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," *CoRR*, vol. abs/1611.01604, 2016.

[8] Z. Wang, H. Mi, W. Hamza, and R. Florian, "Multi-perspective context matching for machine comprehension," *CoRR*, vol. abs/1612.04211, 2016.