Multi-level Embedding Representation for Reading Comprehension

Xiaoyu Wang   Yidi Zhang   Yihui Wu   Xinsheng Zhang
Center For Data Science, New York University
{xw1435, yz3464, yw1348, xz1757}@nyu.edu

Abstract

Reading comprehension is a popular task in question answering system. Previous methods mainly focus on exploring different architectures of deep neural networks and attention mechanisms. With the inspiration of recent works on improvement through linguistic features, we propose a novel multi-level embedding by incorporating Inside Outside Beginning (IOB) tagging for Named Entity Recognition (NER), IOB tagging for Noun Phrases (NP), and Part of NER tagging. The experiment evaluations show our model has improvement comparing with baseline model on the Stanford Question Answering Dataset (SQuAD). We provide project source code on GitHub[1] for reproduction.

1 Introduction

Question answering (QA) system is a system that can generate answers to corresponding question queries given articles (Kolomiyets and Moens, 2011). All successful products such as Apple Siri, Google Assistant, and Amazon Alexa rely on the precise understanding of natural language, which is a core of the question answering system. Reading comprehension is a specific task in the field of QA that requires understanding and reasoning. The success of this task has three components: context, question query and the answer.

In our project, the first challenge is machine reading, which requires a deep understanding of lexical, syntactic, and semantic meaning. The second challenge is question comprehension, which requires interaction with context and finding the correct answer through reasoning. A naive method to answer a question is to rank all possible sequential tokens and pick up the best answer based on the highest probability score. It is hard to pick up a correct answer from all possible answer spans without fully understanding of both context and question. Rajpurkar et al. (2016) release the Stanford Question Answering Dataset (SQuAD) which is a large high quality QA benchmark dataset. Researchers are able to explore and train end-to-end deeper neural networks to tackle reading comprehension task. Rajpurkar et al. (2016)'s baseline uses a simple linear model with enriched syntactic information, such as part-of-speech (POS) tags and dependency parse tree, which show that syntactic and lexical features provide strong meaning for the reading comprehension task.

Most previous significant works focus on building complex model structures and powerful attention mechanisms. With the inspiration of Pan et al. (2017) and Hashimoto et al. (2016)'s work, we decide to take another approach to tackle reading comprehension task. As far as we consider, understanding is the crucial key for many tasks in NLP research such as natural language inference, reading comprehension, machine translation, just to name a few. With better understanding of language, machine is able to capture synonymy and antonymy information, distinguish ambiguous sentences, or even comprehend sarcasms. But, how could a machine understand a sentence better? We think one of the crucial points for understanding is the better use of linguistic meaning.

Pan et al. (2017) propose a multi-layers embedding network to encode both context and question with a sequence of word-level embedding, POS embedding, and name-entity recognition (NER) embedding. This multi-layer embedding can fully use syntactic and semantic information of language. Hashimoto et al. (2016)'s work inspires us that exploring more linguistic hierarchical embeddings could be a good try in our task. They propose a framework to jointly train word, character, POS, chunking, and dependency parsing embed-

[1] https://github.com/nyuxz/ds1012_final_project

ding.

As our computing resources are limited, we decide to use pre-trained GloVe and Character n-gram embeddings and fine tuning during training process to form first 2-level embedding. In order to utilize and compare the influence of syntactic and semantic information, we concatenate POS and NER features with fine-tuned 2-level embedding (Chen et al., 2017). We also introduce new features including Inside Outside Beginning (IOB) tagging for noun phrases, IOB tagging for NER, and Part of NER tagging to make better use of linguistic meaning. We concatenate our new features to previous features as a multi-level embedding on both context and question query. Our goal is to use the new embedding representation enriched with syntactic and semantic concepts to improve model performance on reading comprehension task. We conjecture that with our novel embedding, the model can find starting position of answer span more precisely.

2 Related Work

The original SQuAD paper by Rajpurkar et al. (2016) sets a strong baseline by using a logistic regression model and representing contexts and questions with n-grams and part-of-speech sparse features. Recently, deep neural networks are proved successful in tackling reading comprehension tasks. Hermann et al. (2015) introduce an attention based model to tackle reading comprehension. Then a popular architecture is proposed by Wang and Jiang (2016), which combines match-LSTM with Pointer Net to sequentially match the attention between context and the question query and aggregate matching attention to predict the answer span. Seo et al. (2016) introduce a more powerful attention mechanism to solve complex interactions between context and questions. Their bi-directional attention flow can obtain a question-focused context representation by computing and attending the attention weights at every time step. Cui et al. (2016) propose Attention-over-attention model that employs attention over document-level attention to generate attended attention for question context pairs. Chen et al. (2016) introduce a multi-layer bidirectional LSTM with enriched context features. This model trains fast and achieves a high score. Yu et al. (2018) propose a convolution network with self-attention mechanism aiming to provide fast solution for reading comprehension. Their model achieves the state-of-the-art single model performance with exact match score 82.21, and F1 score 88.61.

Besides exploring different model architectures and attention mechanisms, exploring context and question embedding is also a crucial way to tackle the reading comprehension task. Most models on the SQuAD leader board use pre-trained GloVe embedding introduced by Pennington et al. (2014) as their word-level features. Context vectors (CoVe) embedding by McCann et al. (2017) and character-level-embeddings (Lee et al., 2016; Kim et al., 2015) are both good addition to GloVe embedding. Character-level embeddings handle out-of-vocabulary (OOV) word by representing low level structures. Context vectors as a co-product of machine translation LSTM, provide word context in sentences.

Recent powerful deep neural network with complex architecture and attention mechanism such as (Hu et al., 2017; Wang et al., 2017) only use word-level and character-level representation in context and question encoding. Pan et al. (2017) suggest using multi-layer embedding by adding part-of-speech tagging (POS) and name-entity recognition (NER) tagging to GloVe vectors. Chen et al. (2017) use exact matching (EM) and aligned question embedding to co-reference occurrences of words between context and query. While most embeddings are either character level or word level, there are also approaches that add embedding on top of question query vectors. Zhang et al. (2017) explicitly encode question type of a query by a one-hot vector indicating "what, how, who, when, which, where, why, be, whose, whom, and other" and add it to the query representation.

3 Data

SQuAD is a question answering dataset which consists of over 100k questions on Wikipedia articles and the answer to each question is directly derived from one corresponding document. It contains 87,599 training samples, 71,231 development samples, and 10,570 hidden test samples (Rajpurkar et al., 2016). CNN/Daily Mail dataset (Hermann et al., 2015) is another reading comprehension benchmark data, but it is relatively noisy due to coreferential errors and has limited level of reasoning and inference (Chen et al., 2016). Comparing with CNN/Daily Mail dataset, SQuAD has

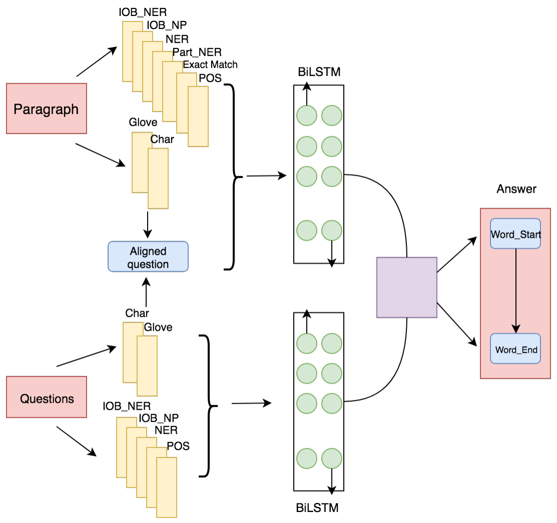


Figure 1: Model Architecture

relatively shorter context, and has good diversity of answers (Rajpurkar et al., 2016). So we conduct our experiments on SQuAD.

4 Methodology

Our model is based on the Document Reader model proposed by Chen et al. (2017). Given the context and question pair(c, q), the target is to predict a span from a_{start} to a_{end} that contains the answer. Based on the document reader model structure, we introduce multi-level embedding. The model architecture is shown in Figure 1.

4.1 Baseline

4.1.1 Word-level Embedding

We use 300-dimensional GloVe embeddings [2] which are trained on 840B common crawl tokens (Pennington et al., 2014). We encode the context and question query using pre-trained GloVe embeddings and tuned the parameters during training, then feed into model.

4.1.2 Exact match and Question Alignment

The exact match features and the aligned question embedding follow Chen et al. (2017), and Lee et al. (2016).

$$f_{exact\_match}(c_i) = I(c_i \in q) \tag{1}$$

Three binary features are used to indicate if c_i has a match in q in either original, lowercase or lemma form.

[2]https://nlp.stanford.edu/projects/glove/

$$f_{align}(c_i) = \sum_j a_{i,j}\mathbf{E}(q_j) \tag{2}$$

where $\mathbf{E}(\cdot)$ is the word embedding operation.

With this notation, $f_{align}(c_i)$ essentially gives an representation of c_i by question word embeddings, where $a_{i,j}$ denotes an affinity measure between c_i and q_j:

$$a_{i,j} = softmax(\alpha(\mathbf{E}(c_i)) \cdot \alpha(\mathbf{E}(q_j))) \tag{3}$$

where $\alpha(\cdot)$ is a ReLU nonlinearity layer.

Exact match feature and aligned question embeddings offer alignments between context word embeddings and question embeddings through word form matching or word sense matching.

4.1.3 Paragraph Encoder

A bi-directional long short term memory network (Bi-LSTM) is used as the paragraph encoder. With the feature vectors sequence $\{\tilde{c_1}, \tilde{c_2}, ...\tilde{c_n}\}$, let $\overrightarrow{h_i}$ denote the hidden state of the LSTM in forward pass of the sequence at position i and let $\overleftarrow{h_i}$ denote the hidden state of the LSTM in backward pass at position i.

The contextual embedding of $\mathbf{c}_i$ is then formulated as the concatenation of $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$. As a result, we will obtain the encoder output a sequence $\{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_n\}$ as our context encoding.

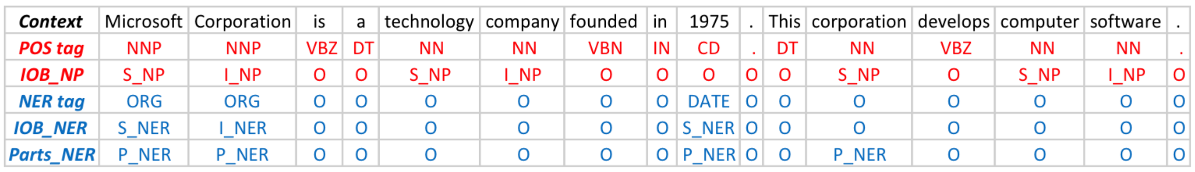
4.1.4 Question Encoder

The encoding process of questions follows a similar approach. Given a question q consists of token length m, each token q_i will be first mapped into a feature vector $\tilde{q_j}$ from embedding layer. We will then process them sequentially through another Bi-LSTM encoder to receive a sequence $\{\mathbf{q}_1, \mathbf{q}_2, ...\mathbf{q}_m\}$. The given question will then be encoded as a single vector $\mathbf{q}$ as the weighted sum of all $\mathbf{q}_j := \sum_{j=1}^{m} b_j \mathbf{q}_j$ where b_j is the importance parameter to be learned:

$$b_j = softmax(w \cdot \mathbf{q}_j) \tag{4}$$

4.1.5 Predictor

After we have obtained a sequential expression of the context encoding $\{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_n\}$ and a question encoding $\mathbf{q}$, the model will now proceed to predict the answer span. In order to find a span from a_{start} to a_{end} that most likely contains the true answer, we will train two independent pointer predictors to mark the beginning and end of the most probable answer range.



Context	Microsoft	Corporation	is	a	technology	company	founded	in	1975	.	This	corporation	develops	computer	software	.
POS tag	NNP	NNP	VBZ	DT	NN	NN	VBN	IN	CD	.	DT	NN	VBZ	NN	NN	.
IOB_NP	S_NP	I_NP	O	O	S_NP	I_NP	O	O	O	O	O	S_NP	O	S_NP	I_NP	O
NER tag	ORG	ORG	O	O	O	O	O	O	DATE	O	O	O	O	O	O	O
IOB_NER	S_NER	I_NER	O	O	O	O	O	O	S_NER	O	O	O	O	O	O	O
Parts_NER	P_NER	P_NER	O	O	O	O	O	O	P_NER	O	O	P_NER	O	O	O	O

Figure 2: Example of different tagging. We mark S, I, and O to indicate beginning, middle, and outside of named entities or noun phrases on top of POS and NER tags. In this example, *Microsoft Corporation* is a named entity. We mark *Microsoft* as S_NER and *Corporation* as I_NER. For Part_NER tagging, *corporation* refers to *Microsoft Corporation*, thus is marked as P_NER.

4.2 Additional Features

4.2.1 Character-level Embedding

In Chen et al. (2017)'s model, they only represent words in word-level by assigning a vector, and this may lose internal structure of words. In order to enrich sub-word information and to efficiently provide morphological features (Bojanowski et al., 2016), we introduce character-level embedding layer. We first represent a word as the sum of a bag of character n-grams vectors. This vector illustrates our character-based representation and can better represent rare words and unknown words. We use pre-trained 100-dimensional character n-grams embedding [3] trained by skip-gram model on Wikipedia data. For each word, we concatenate the GloVe representation with the character n-grams representation into a 400-dimensional vector embedding and fine tune during training process.

4.2.2 Part-of-Speech Feature

Part-of-Speech tagging assigns a word token in the text corpus with a part of speech tag. It provides syntactic information within sentences.

4.2.3 Named Entity Recognition Feature

Named Entity Recognition provides information for identifying named entities in text. As answers in the SQuAD dataset are mostly named entities or noun phrases (Rajpurkar et al., 2016), we believe this tagging information could be helpful to identify the correct answer.

4.2.4 IOB Features

IOB tagging provides syntactical information for chunked structures in text corpus. We perform IOB tagging for NER and noun phrases (NP). For IOB-NER tagging, we mark S_NER, I_NER, and O to indicate beginning, inside, and outside of a named entity. IOB-NP tagging would follow similarly. We mark S_NP, I_NP, and O for noun phrases. The example is shown in Figure 2.

4.2.5 Part of NER Feature

An entity can be represented by several words. While each entity phrase has its own NER tag, it is possible that occurrences of partial phrase that are not tagged can refer to the same entity. For example, in Figure 2, *corporation* refers to *Microsoft Corporation* but is not tagged. We use an indicator vector to represent part of NER tagging for each word in the context, where **1** represents a word occurring as part of the NER tag. Each word that is a partial entity will be tagged P_NER.

5 Experiment & Result

5.1 Training Details

The model architecture is shown in Figure 1. We use Adam optimizer (Kingma and Ba, 2014) with batch size of 32, learning rate of 0.1, hidden size of 128 and multi-level embedding dimension of 477. The training process for 40 epochs takes about 5 hours with a single GTX1080Ti GPU.

5.2 Ablation

We concatenate GloVe embedding with three features used by Chen et al. (2017): $f_{exact\_match}(c_i)$, $f_{align}(c_i)$, and $f_{term\_frequency}(c_i)$ as our baseline model.

We fix the model design, but change different combinations of embedding to explore how syntactic and semantic embeddings help to tackle reading comprehension task. Table 1 shows our experimental results on SQuAD dev-set with different embedding settings. We use F1 score and Exact Match (EM) score to evaluate the model performance.

As shown in Table 1, by adding character level embedding, model performance shows improvement. After adding our novel embedding features

[3] http://www.logos.t.u-tokyo.ac.jp/~hassy/publications/arxiv2016jmt/

Model	Multi-level Embedding Setting	EM	F1
1	Baseline (Glove + Term Frequency + Exact Match Feature + Aligned Question Embedding)	67.89	77.49
2	Baseline + Char	68.20	77.72
3	Baseline + POS + NER	68.18	77.68
4	Baseline + Char + POS + NER	68.49	77.71
5	Baseline + Char + POS + NER + IOB_NP	68.53	77.76
6	Baseline + Char + POS + NER + IOB_NER	68.61	77.81
7	Baseline + Char + POS + NER + IOB_NP + IOB_NER	69.23	78.15
8	Baseline + Char + POS + NER + IOB_NP + IOB_NER + Part_NER	**69.47**	**78.64**
9	Baseline + Char + POS + NER + IOB_NP + IOB_NER + Part_NER + Question Embedding	69.42	78.55

Table 1: Experiments results for different combination of multi-level embedding.

IOB-NER, IOB-NP, and Part-NER, the model further improved. One interesting finding is that including these IOB features together increases model performance by 0.74%, but adding IOB-NP or IOB-NER alone increases EM score by only 0.04% and 0.12% respectively. After adding Part_NER tagging, our best model (Model 8) is able to achieve F1 score at 78.64% and EM score at 69.47%. We also notice that adding question encoding does not contribute much to performance improvement.

5.3 Result Analysis

We analyze model's capacity to correctly predict starting position of answer span. As listed in Table 2, Model 4 predicts starting position correctly for 72.01% instances, and Model 8 with novel embeddings predicts correctly 73.58%. This result shows IOB tagging and Part of NER tagging can help model better predict starting positions of answer span. Table 3 shows two error answers given by Model 4 and Model 8. Although both model fail to predict the exact answer, Model 8 correctly locates the starting position while Model 4 predicts an irrelevant span.

Model Setting	% Correct
Model 4	72.01
Model 8	73.58

Table 2: Model comparison for finding starting position correctly. See Table 1 for model setting.

Context: ......Martin Parry, a climate expert who had been co-chair of the IPCC working group II, said that......
Question: What was Martin Parry's role in the IPCC ?
Model 4 answer: generally unfounded and also marginal to the assessment
Model 8 answer: co-chair
Reference answer: co-chair of the IPCC working group II.

Table 3: Error example

Our novel embeddings (IOB-NER, IOB-NP, and Part of NER) are built on top of POS tags and NER tags. To show how much improvement is achieved by adding IOB and Part of NER tagging, we select a subset of question-answer pairs where answer words are tagged as noun phrases or named entities only. There are 6343 such sample pairs out of 10570 in the dev-set. Table 4 shows model performance on the NER/NP tagged answers. By adding our novel embeddings, Model 8 outperforms Model 4. With question embedding, Model 9 achieves better scores on both EM and F1.

Model Setting	EM	F1
Model 4	75.45	81.26
Model 8	76.30	82.03
Model 9	76.53	82.06

Table 4: Model comparison on subset of question-answer pairs where answer words are tagged as noun phrases or named entities only. See Table 1 for model setting.

5.4 Error Analysis

We perform error analysis on 50 random selected incorrect answer cases for Model 4 and Model 8 respectively. We further divide them into 8 categories, as shown in Table 5.

For Model 8, among the 8 categories, 22% of the errors are due to model having trouble understanding different expressions referring to the same entity. 32% are due to complicated syntactical structures in the context. Through analyzing the error examples, we find that the model is less effective in understanding conjunction words (and, as) within compound-complex sentences. For 10% of errors, the information needed spans among multiple sentences. We also notice that there is a 6% error rate where the predictor finds correct answer starting position but gives imprecise ending position.

We see a similar distribution of error categories between the two models. We notice a 6% difference in the syntactical complications category.

This could be due to noises brought by our tagging methods.

Error Description	Model 4	Model 8
Need out of context information	6%	8%
Correct starter but imprecise span	10%	6%
Syntactical complications	26%	32%
Co-reference, paraphrase	24%	22%
Answer spans multiple sentences	10%	10%
Mistakes during parsing	8%	6%
Abstract or ambiguous questions	10%	10%
Data preprocessing problem	6%	6%

Table 5: Error Percentage for 8 categories on 50 error samples. See Table 1 for model setting.

6 Conclusion and Future Work

In this paper, we introduce multi-level embedding to enrich syntactic and semantic information for both context and questions. The experimental evaluations show that IOB-NER, IOB-NP, and Part of NER help improve model performance on the Stanford Question Answering Dataset (SQuAD). Our analysis further demonstrates that with these novel embeddings, the model is capable of better locating starting position of answers. We also show that our multi-level embedding model has better performance on NER/NP tagged answers.

For future work, we aim to improve NER and POS tagging to better capture linguistic information. Additionally, we will explore possible methods to more precisely predict ending positions of answer spans as well.

Acknowledgement

We would like to thank Nikita Nangia for reviewing this paper and providing helpful comments.

Collaboration Statement

We conduct experiments together. For writing part, Yihui Wu focuses on part of literature review and conclusion. Xinsheng Zhang focuses on introduction, part of literature review, and experiments setting. Yidi Zhang and Xiaoyu Wang focus on model architecture part, dataset, and error analysis. All of us work on Methodology part together.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. *CoRR*, abs/1606.02858.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2016. Attention-over-attention neural networks for reading comprehension. *CoRR*, abs/1607.04423.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple NLP tasks. *CoRR*, abs/1611.01587.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *CoRR*, abs/1506.03340.

Minghao Hu, Yuxing Peng, and Xipeng Qiu. 2017. Mnemonic reader for machine comprehension. *CoRR*, abs/1705.02798.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2015. Character-aware neural language models. *CoRR*, abs/1508.06615.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Oleksandr Kolomiyets and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Inf. Sci.*, 181(24):5412–5434.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *CoRR*, abs/1610.03017.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. *CoRR*, abs/1708.00107.

Boyuan Pan, Hao Li, Zhou Zhao, Bin Cao, Deng Cai, and Xiaofei He. 2017. MEMEN: multi-layer embedding with memory networks for machine comprehension. *CoRR*, abs/1707.09098.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. pages 1532–1543.

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

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *CoRR*, abs/1608.07905.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. pages 189–198.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution.

Junbei Zhang, Xiao-Dan Zhu, Qian Chen, Li-Rong Dai, Si Wei, and Hui Jiang. 2017. Exploring question understanding and adaptation in neural-network-based question answering. *CoRR*, abs/1703.04617.