Enhancing Embedding via Two-level Features for Machine Reading Comprehension

Shuyi Wang, Hui Song, Bo Xu, Hongkuan Zhang

School of Computer Science and Technology, Donghua University, Shanghai, 201620, China

東華大學
DONGHUA UNIVERSITY

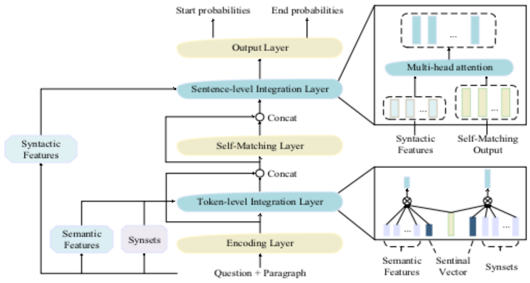
Contribution

- Through the algorithm analysis and case study of MRC task, we confirm the issues of existing models and propose a two-level approach for integrating token features and grammatical structure of token-pairs into sentence encoding.
- For token-level, we select the valid features from the candidates for different tokens according to the actual context, sentinel vector is added. And for sentence level, We transform the dependency tree of sentence into m-hop matrixes, and then integrate the information through multi-head attention mechanism.
- We combine different features that need to be introduced to conduct experiments, find that the impact of features varies on different datasets. We choose DocQA, pre-trained model Google BERT_base, KT-NET_base as baseline methods. Our model achieves +0.63 □□ /+0.68 $\square_{\square}$ improvements in SQuAD1.1 dataset and +6.43 □□ /+6.79 $\square_{\square}$ improvements in ReCoRD dataset than the best result of baseline method, better than the current state-of-the-art models.

Modeling

Method

- ***Task:*** Span extraction of MRC task, which predicts the answer by locating the start and the end position in a context.
  - Given a question $\square = \{\square_1, \square_2, \ldots, \square_{\square}\}$ with □tokens and a context $\square = \{\square_1, \square_2, \ldots, \square_m\}$ with □ tokens.
  - The extracted answer is a span in □.
- ***Construction:*** TLE-BERT model is based on BERT, and enhance the BERT embedding in two levels: token-level and sentence-level.
  - In token-level, for each token $\square \in \square \cup \square$, we add external knowledge □□□ from the synonym dictionary and the semantic features □ of the token. The features are introduced by sentinel mechanism.
  - In sentence-level, the dependency relations □ is fused by multi-head attention mechanism with the token embedding.



Encoding Layer based on BERT Model

- ***Input:*** $\square = \{\square_1, \square_2, \ldots, \square_n\}$, □ is the number of tokens.
  - Input contains the question sequence □ , the context sequence □ and three tags [□□□], [□□□], [□□□].
    - [□□□] is the classification label used in the classification task
    - [□□□] is used to distinguish the question and the context sequence
  - $\square = \{[\square\square\square], \square, [\square\square\square], \square, [\square\square\square]\}$
- ***Encoding:*** for each token $\square_i$, the initial text embedding $\square_i^{token}$, position embedding $\square_i^{pos}$ and segment embedding $\square_i^{seg}$ are added together to obtain the initial input $\square_i^0$.
  - $\square_i^0 = \square_i^{token} + \square_i^{pos} + \square_i^{seg}$
- ***Output:*** input embedding $\square_i^0$ passes through the N-layer *encoder* of BERT to get context-aware representation $\square_i^N$, and the output embedding sequence □ of this layer.
  - $\square = \{\square_1^N, \square_2^N, \ldots, \square_n^N\}, \square_i^N = \square\square\square\square\square\square(\square_i^{N-1})$

Token-level Integration Layer

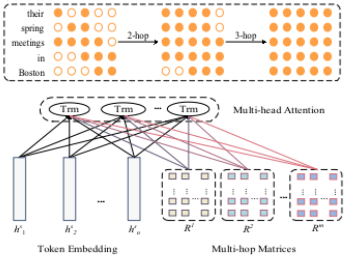
- This layer integrates the semantic features □ and external knowledge □□□ with the context-aware embedding representation $\square_i^N$.
  - Semantic features □ are identified from input sequence □.
    - include three embedding sequences, POS tag $\square_{POS} = \{\square_i^{POS}\}$, entity type tag $\square_{ET} = \{\square_i^{ET}\}$ and noun phrase tag $\square_{NP} = \{\square_i^{NP}\}$
    - enhance semantics on the basis of BERT embedding
  - External knowledge □□□ are synonyms retrieved from **WordNet**.
    - enrich the BERT embedding representation with external knowledge
- ***Input:*** semantic features$(\square_i^{POS}, \square_i^{ET}, \square_i^{NP})$, some synonyms $|\square\square\square_i|$ and BERT embedding $\square_i^N$. (for each token $\square_i$ in □)
- ***Integration:***
  - calculate two attention weights of □-token between □-semantic feature and □-Synonym, $\square_i^j$ and $\square_i^k$
    - $\square_i^j = \square_i^{j^T} \square\, \square_i^N, \square_i^k = \square\square\square_i^{k^T} \square\, \square_i^N$
  - employ sentinel attention mechanism to organically integrate these features and knowledge
    - $\square_i^1 = \overline{\square}^T \square\, \square_i^N, \square_i^2 = \overline{\square\square\square} \cdot \square\, \square_i^N$
  - calculation of the final semantic feature $\square_i'$ and knowledge $\square\square\square_i'$
    - $\square_i' = \sum_j \square_i^j \square_i^j + \square_i^1 \overline{\square}, \square\square\square_i' = \sum_k \square_i^k \square\square\square_i^k + \square_i^2 \overline{\square\square\square}$
- ***Output:*** concatenate the $\square_i^N, \square_i'$ and $\square\square\square_i'$ as the output of □-token in this layer
  - $\square_i^{\square} = [\square_i^N, \square_i', \square\square\square_i']$

Self-Matching Layer

- ***Input:*** token-level enriched representation $\{\square_i^{\square}\}_{i=1}^{N}$ as input.
- ***Method:*** use self-attention mechanism to interact context-aware embedding $\{\square_i^N\}$, semantic features $\{\square_i'\}$ and external knowledge $\{\square\square\square_i'\}$. This operation mainly discovers the direct and indirect interactions between tokens.
- ***Output:*** $\square^{\square} = \{\square_i^{\square}\}_{i=1}^{n}$

Sentence-level Layer

- This layer introduces the dependencies of sentence-level into token embedding to get the structure message of the tokens.
- ***Input:*** Self-Matching layer output $\square^{\square}$ and multi-hop dependency matrices
- ***Method:***
  - convert the pair-wise dependency of tokens into a 1-hop dependency matrix $\square^1$
    - $\square_{ij} = \begin{cases} 0, & \square_i \circ \square_j \\ 1, & \square_i - \square_j \end{cases}$
      - no relation
      - exist relation
  - m-hop dependency matrix based on $\square^1$
    - $\square^m = (\square^1)^m$
    - The example of multi-hop matrices is on the right.
  - get three input-based tensor □□□□□, □□□, □□□□□ through $\square^{\square}$
    - $\{\square\square\square\square\square^k, \square\square\square^k, \square\square\square\square\square^k\} = \{\square^{\square}\square^k_{Query}, \square^{\square}\square^k_{Key}, \square^{\square}\square^k_{Value}\}$
  - calculate the □-th head self-attention matrix $\square^k$
    - $\square^k = \square\square\square\square\square\square\left(\frac{Query^k Key^{k^T}}{\sqrt{d_{key^k}}} \square^k\right)$
  - calculate the □-th head self-attention based on □-hop matrix
    - $\square^k = \square^k \square\square\square\square\square^k$
  - concatenate all heads to get the embedding representation
    - $\square = [\square^1, \square^2, \ldots, \square^m]$
- ***Output:*** product the output $\square = \{\square_i\}$ through □
  - $\square = \square\,\square$



Output Layer

- ***Input:*** sentence layer output □
- ***Output:*** start probability $\square_i^1$ and end probability $\square_i^2$ for each token
  - $\square_i^1 = \frac{\square^{\square_1^T \square_i}}{\sum_j \square^{\square_1^T \square_j}}, \square_i^2 = \frac{\square^{\square_2^T \square_i}}{\sum_j \square^{\square_2^T \square_j}}$

Objective Function

- Objective function is the log-likelihood
  - $\square = -\frac{1}{N}\sum_{i=1}^{N}\left(\log \square^1_{\square_i^1} + \log \square^2_{\square_i^2}\right)$

Datasets

- ***Source:***
  - **SQuAD1.1**: StandFord Question Answering Datasets
  - **ReCoRD**: Reading Comprehension with Commonsense Reasoning Datasets
- ***Content:***
  - SQuAD1.1
    - 10,000+ questions based on 500 articles in Wikipedia
    - questions are in the form of interrogative sentences
    - the 11 types of samples are summarized in following table

Question type	Number	Question	Number
What	56905	Where	4090
Who	**9900**	Be/Do/etc.	1671
Which	6620	Why	**1353**
When	6258	Whom	394
How many	**5735**	Whose	**350**
How	**4893**		

  - ReCoRD
    - 110,730 samples with a large portion of queries requiring commonsense reasoning
    - questions are a sentence with @placeholder instead of the missing text span that needs to be predicted

Data Preprocessing

- ***Method:***
  - Features of the sentences we used are mainly obtained through the spaCy tool.
  - Modify the tokenized method of spaCy to make it consistent with BERT.
- ***Analysis:***
  - The tokenization mechanism of the pre-trained language model BERT is different from the spaCy's.

Original: For whos glory did Father O'Hara believed that the Notre Dame football team played ?
BERT: For who s glory did Father O'Hara believed that the Notre Dame football team played ?
spaCy: For whos glory did Father O ' Hara believed that the Notre Dame football team played ?

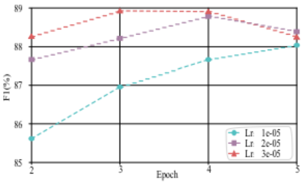
Model Evaluation

- ***Standard:*** MRC task usually needs $\square_{\square}$value and Extract Match (□□) for comprehensive evaluation.
  - $\square_{\square}$: calculate the degree of overlap between the predicted and the correct answer
    - □□□□□: the ration of the number of overlapped characters to the number of correct answer characters
    - □□□□□□□□□: he ration of the number of overlapped characters to the predicted answer characters
    - $\square_{\square} = 2 * \frac{precession*recall}{precession+recall}$
  - □□ *:* precise matching
    - the value of □□ is 1 means that the predicted answer is exactly the same as the correct answer, otherwise is 0
    - in the SQuAD1.1 dataset, each question may have multiple candidate answers
    - the predicted answer only needs to exactly match one of the candidate answers

Result Analysis

Environment And Setup

- ***Environment:***
  - python3.5.2
  - PaddlePaddle-gpu1.8.2
  - NLTK3.5
  - spaCy2.2.4
  - CUDA10.0.130
  - experiments are performed on 2 2080ti GPUs
- ***Setup:***
  - parameters of the BERT_base in PaddlePaddle version
  - the trained model has 12 layers with 768 hidden layer dimension
  - maximum sequence length is set to 384
  - token-level feature's dimension is 20
  - learning rate is 3e-05
  - epoch 3
  - tokenized mechanism is WordPiece



Results

- ***General result:*** compared TLE-BERT with three models DocQA, Google BERT_base and KT-NET (based on BERT_base) in our environment.
  - On SQuAD1.1, TLE-BERT gets a gain of **+0.63 □□ /+0.68 $\square_{\square}$** over the BERT_base.
  - On ReCoRD, TLE-BERT achieves **+6.43 □□ /+6.79 $\square_{\square}$** improvements.

Model	SQuAD1.1		ReCoRD	
	EM	***F1***	***EM***	***F1***
DocQA(ELMo)	-	-	44.13	45.39
Google BERT_base(ours)	81.25	88.41	55.99	57.99
KT-NET_base(ours)	81.56	88.75	60.79	62.91
TLE-BERT(ours)	**81.88**	**89.09**	**62.42**	**64.78**

- ***Different feature result:*** Introduce different features to the model to compare results.
  - NAN in following tabel is the KT-NET_base model containing synset knowledge, without any token or sentence level features.

Model	SQuAD1.1		ReCoRD	
	EM	***F1***	***EM***	***F1***
NAN	81.56	88.75	60.79	62.91
+POS	81.91	**89.06**	62.06	64.55
+ET	81.64	88.91	62.33	**64.67**
+NP	81.69	88.90	61.70	64.21
+DEP	81.83	**89.05**	62.23	**64.69**
ALL	**81.88**	**89.09**	**62.42**	**64.78**

- ***Analysis:***
  - Token-level
    - POS achieve better performance in SQuAD1.1 than in ReCoRD, while ET is the opposite.
    - POS provides grammatical information that the SQuAD1.1 dataset lacks, and ReCoRD needs entity information for reasoning.
  - Sentence-level
    - Dependency relation improves the performance of the model on both datasets, which also confirms our previous conjecture that the pre-trained language model does not obtain the sentence dependency as well.

Conclusion

- This paper proposes a new method of enhancing language representation through token-level and sentence-level features.
  - We can select appropriate features as external information according to the characteristics of the dataset or the application background to improve the model performance.
- We employ an attention mechanism to fuse multiple token-level features and make the model learn to pick a valid feature set by sentinel vectors.
- The sentence-level features enhance the language representation with syntactic information of the sentence.
- Experiments have shown that our method is better than the current SOTA BERT_base models.