A Deep Cascade Model for Multi-Document Reading Comprehension

Ming Yan, Jiangnan Xia, Chen Wu, Bin Bi,
Zhongzhou Zhao, Ji Zhang, Luo Si, Rui Wang, Wei Wang, Haiqing Chen
Alibaba Group
{ym119608, jiangnan.xjn, wuchen.wc, b,bi}@alibaba-inc.com
{zhongzhou.zhaozz, zj122146, luo.si, masi.wr, hebian.ww, haiqing.chenhq}@alibaba-inc.com

Abstract

A fundamental trade-off between effectiveness and efficiency needs to be balanced when designing an online question answering system. Effectiveness comes from sophisticated functions such as extractive machine reading comprehension (MRC), while efficiency is obtained from improvements in preliminary retrieval components such as candidate document selection and paragraph ranking. Given the complexity of the real-world multi-document MRC scenario, it is difficult to jointly optimize both in an end-to-end system. To address this problem, we develop a novel deep cascade learning model, which progressively evolves from the document-level and paragraph-level ranking of candidate texts to more precise answer extraction with machine reading comprehension. Specifically, irrelevant documents and paragraphs are first filtered out with simple functions for efficiency consideration. Then we jointly train three modules on the remaining texts for better tracking the answer: the document extraction, the paragraph extraction and the answer extraction. Experiment results show that the proposed method outperforms the previous state-of-the-art methods on two large-scale multi-document benchmark datasets, i.e., TriviaQA and DuReader. In addition, our online system can stably serve typical scenarios with millions of daily requests in less than 50ms.

Introduction

Machine reading comprehension (MRC), which empowers computers with the ability to read and comprehend knowledge and then answer questions from textual data, has made rapid progress in recent years. From the early cloze-style test (Hermann et al. 2015; Hill et al. 2015) to answer extraction from a single paragraph (Rajpurkar et al. 2016), and to the more complex open-domain question answering from web data (Joshi et al. 2017; Nguyen et al. 2016), great efforts have been made to push the MRC technique to more practical applications.

The rapid progress of MRC in recent years mostly owes to the release of the single-paragraph benchmark dataset SQuAD (Rajpurkar et al. 2016), on which various deep attention-based methods have been proposed to constantly push the state-of-the-art performance (Seo et al. 2016; Wang et al. 2017c; Yu et al. 2018). It is a significant milestone that several MRC models have exceeded the performance of human annotators on the SQuAD dataset[1]. However, the SQuAD dataset makes a strong assumption that the answers are contained in the given paragraphs. Besides, the parapraphs are rather short, approximately 200 words on average, while a real-world scenario usually involves multiple documents of much longer length. Therefore, several latest studies (Joshi et al. 2017; Clark and Gardner 2017; Tan et al. 2017) begin to re-design the task into more realistic settings: the MRC models are required to read and comprehend multiple documents to reach the final answer.

In multi-document MRC, depending on the way of combining the two components, document selection and extractive reading comprehension, there are two categories of approaches: 1) The pipeline approach treats the document selection and extractive reading comprehension as two separate parts, where a document is firstly selected through document ranking and then passed to the MRC model for extracting the final answer (Joshi et al. 2017; Wang et al. 2017a); 2) Several recent studies (Tan et al. 2017; Clark and Gardner 2017; Wang et al. 2018) adopt a joint learning method to optimize both sub-tasks in a unified framework simultaneously.

The pipeline method relies heavily on the quality of the document ranking module. When it fails to give the relevant documents higher ranks or filters out the ones that contain the correct answers, the downstream MRC module has no way to recover and extract the answers of interest. For the joint learning method, it is computationally expensive to jointly optimize both tasks with all the documents. This computation cost limits its application to the operational online environment, such as Amazon[2] and Taobao[3], where efficiency is a critical factor to be considered.

To address the above problems, we propose a deep cascade model which combines the advantages of both methods in a coarse-to-fine manner. The deep cascade model is designed to properly keep the balance between the effectiveness and efficiency. At early stages of the model, simple features and ranking functions are used to select a candidate set of most relevant contents, filtering out the irrelevant

Copyright © 2019, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] https://rajpurkar.github.io/SQuAD-explorer/
[2] https://www.amazon.com/
[3] https://www.taobao.com/

documents and paragraphs as much as possible. Then the selected paragraphs are passed to the attention-based deep MRC model for extracting the actual answer span at word level. To better support the answer extraction, we also introduce the document extraction and paragraph extraction as two auxiliary tasks, which helps to quickly narrow down the entire search space. We jointly optimize all the three tasks in a unified deep MRC model, which shares some common bottom layers. This cascaded structure enables the models to perform a coarse-to-fine pruning at different stages, better models can be learnt effectively and efficiently.

The overall framework of our model is demonstrated in Figure 1, which consists of three modules: document retrieval, paragraph retrieval and answer extraction. The first module takes the question and a collection of raw documents as input. The module at each subsequent stage consumes the output from the previous stage, and further prunes the documents, paragraphs and answer spans given the question. For each of the first two modules, we define a ranking function and an extraction function. The ranking function is first used as a preliminary filter to discard most of the irrelevant documents or paragraphs, so as to keep our framework efficient. The extraction function is then designed to deal with the auxiliary document and paragraph extraction tasks, which is jointly optimized with the final answer extraction module for better extraction performance. The local ranking functions in different modules gradually increase in cost and complexity, to properly keep the balance between the effectiveness and efficiency.

The main contributions can be summarized as follow:

- We propose a deep cascade learning framework to address the practical multi-document machine reading comprehension task, which considers both the effectiveness and efficiency in a coarse-to-fine manner.
- We incorporate the auxiliary document extraction and paragraph extraction tasks to the pure answer span prediction, which helps to narrow down the search space and improves the final extraction result in multi-document MRC scenario.
- We conduct extensive experiments on two large-scale multi-document MRC benchmark datasets: TriviaQA (Joshi et al. 2017) and DuReader (He et al. 2017). The results show that our deep cascade model can outperform the previous state-of-the-art performance on both datasets. Besides, the proposed model has also been successfully applied in our online system and stably serve various scenarios in a quick response time of less than 50ms.

Related Work

Machine Reading Comprehension

Recently, we can see emerging interests in multi-document MRC research (Nguyen et al. 2016; Clark and Gardner 2017; Wang et al. 2017b; He et al. 2017; Wang et al. 2018), where multiple documents are given as input. There are two categories of approaches: the pipeline-based approaches and the joint learning models. The pipeline approach firstly selects a single document via ranking and then pass it to the MRC model to extract the precise answer (Joshi et al. 2017; Wang et al. 2017a). This approach gives huge burden to the document ranking model, in which the downstream MRC model has no way to extract the right answer if the relevant documents are missed. The joint learning approaches take all the documents into consideration and extract the answer by comparing it against other documents (Clark and Gardner 2017; Tan et al. 2018; Wang et al. 2018). (Clark and Gardner 2017) proposes a confidence-based method with a shared normalization training objective, which enables the model to produce globally correct output. (Tan et al. 2018) proposes an extraction-then-synthesis framework, by also incorporating passage ranking to answer span prediction. (Wang et al. 2018) further proposes a verification method to make use of the extracted answers in different documents to verify each other for more accurate prediction. However, taking all the documents into consideration will inevitably bring more computation cost, which can be unbearable in the operational online environment. Our deep cascade model can serve as a proper tradeoff between the pipeline method and joint learning method. It has a coarse-to-fine structure which can eliminate irrelevant documents and paragraphs in the early stages with simple features and models, and better identify more relevant answers in a well-designed multi-task deep MRC model on the remaining content.

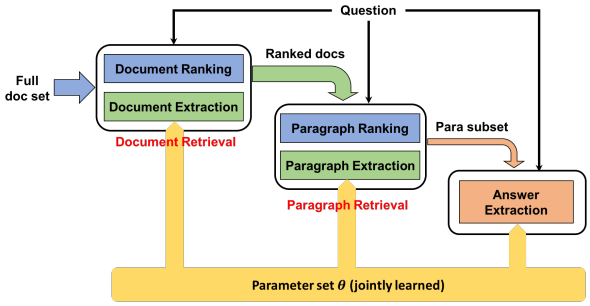

Figure 1: The overall framework of our deep cascade model, which consists of the document retrieval, paragraph retrieval and answer extraction modules.

Cascade Learning

In designing online systems, trade-off between effectiveness and efficiency remains a long-standing problem. Cascade learning is an alternative strategy that can better balance these two, which utilizes a sequence of functions in different stages and allows using different sets of features for different instances. It is firstly introduced in the traditional classification and detection problems such as fast visual object detection (Schneiderman 2004; Bourdev and Brandt 2005), and then widely applied in ranking applications for achieving high top-k rank effectiveness in an efficient manner (Lefakis and Fleuret 2010; Wang, Lin, and Metzler 2011; Liu et al. 2017). (Wang, Lin, and Metzler 2011) uses an Adaboost style framework with two independent ranking functions in each stage, one for pruning the input ranked documents and the other for refining the rank order.

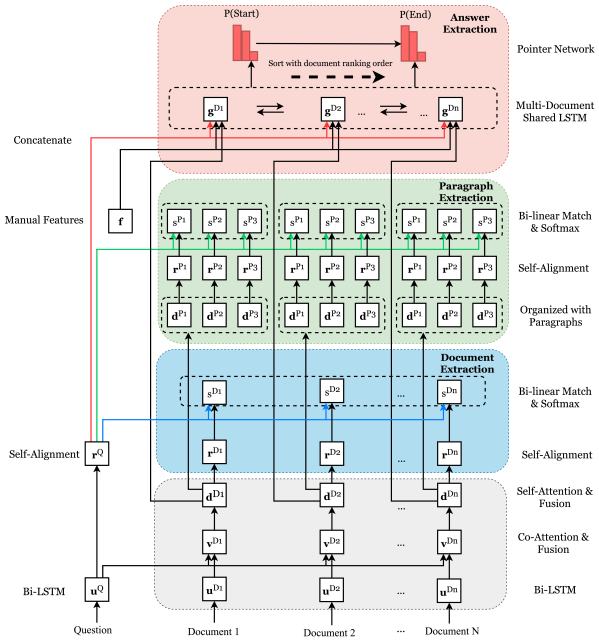

Figure 2: The deep attention-based multi-task MRC model.

We apply the idea of cascade learning to machine reading comprehension, from a preliminary document-level and paragraph-level ranking of the candidate texts, to a more precise answer span extraction. The extracted answer spans are progressively narrowed down across different levels, and the ranking and extraction functions also progressively increase in complexity for more precise answer prediction.

The Deep Cascade Model

Following the overview in Figure 1, our approach consists of three cascade modules: document retrieval, paragraph retrieval and answer extraction. The cascade ranking functions in the first two modules aim to fast filter out the irrelevant document content based on the basic statistical and structural features, and obtain a coarse ranking for the candidate documents. For the remaining document content, we design three extraction tasks at different granularities, with the goal to simultaneously extract the right document, paragraph and also the answer span. A deep attention-based MRC model is designed to jointly optimize all the three extraction tasks, by sharing the common bottom layers, as is shown in Figure 2. The final answer is thus determined by not only the answer span prediction score, but also the corresponding document and paragraph prediction score.

Cascade Ranking Functions

Given a question $\mathbf{Q}$ and a set of candidate documents $\{\mathbf{D}_i\}$, we first introduce the cascade ranking functions of the first two modules for pruning the documents, which gradually increases in complexity.

Document Ranking This part aims at fast filtering out the irrelevant documents and obtaining a coarse ranking for the candidate documents. We utilize the traditional information retrieval methods, such as BM25 and TF-IDF distance, to measure the relevance between the question and document. The matching is conducted between the textual metadata of question and document, including the document title and main content. Besides, the recall ratio of the question words from the document metadata is used as another feature to indicate the relevance of the document.

To learn the importance of different features, we use a learning-to-rank model to assign a weighted relevance score to each retrieved document. By design, the first stage needs to be quick and simple, so we cast the task as a binary classification problem and adopt the pointwise logistic regression as the ranking function. The documents containing the answer are labeled as positive. After this ranking, we only keep the top-K ranked documents for further processing.

Paragraph Ranking This part aims at fast discarding the irrelevant content within each document at a paragraph level. Specifically, given an output document $\mathbf{D}_i = \{\mathbf{P}_{ij}\}$ from the previous stage, we first prune the noisy paragraphs without word or entity match. The simple question and paragraph textual matching features are also extracted as in that of document ranking. Moreover, the document structure can contain some sort of inherent information, for example, the first paragraph within a document may tend to possess more informative content as a document abstract. Therefore, we also add some structural features, such as whether the paragraph is the first or last paragraph of the document, the length of the paragraph, the length of the previous or subsequent paragraphs. To understand the question, we also incorporate the question type information as several binary features if is given, e.g. for DuReader dataset.

To better combine different kinds of features, we adopt a scalable tree boosting method XGBoost (Chen and Guestrin 2016) for ranking, which is widely used to achieve state-of-the-art results on many large-scale machine learning challenges. Again, we use the binary logistic loss for model training and label the paragraph containing the answer as positive. As a result, we select the top-N paragraphs from each document for the subsequent answer prediction.

Multi-task Deep Attention Model

Given the selected P paragraphs from the top-ranked K documents, the final task is to extract an answer span to answer the question $\mathbf{Q}$. A deep attention-based MRC model is designed to achieve this goal. However, with all these documents and paragraphs, it may be still difficult to directly conduct the pure answer prediction at a precise word level, as in that of SQuAD dataset. The document and paragraph information is also not fully exploited. Therefore, we split the answer prediction task into three joint tasks: document extraction, paragraph extraction and answer span extraction. The three tasks share the same bottom layers, which represents the semantics of the document context with respect to the question words, as is shown in Figure 2. By introducing the auxiliary document extraction and paragraph extraction tasks, the proposed model can progressively narrow down the search space from coarse to fine, which helps to better

locate on the final answer span. The final answer prediction is based on the results of all the three tasks, which is jointly optimized with a joint learning method.

Shared Q&D Modeling Given a question $\mathbf{Q}$ and a set of selected documents $\{\mathbf{D}_i\}$, one of the keys in MRC model lies in how to incorporate the question context into the document, so that important information can be highlighted. We follow the attention & fusion mechanism used in (Wang, Yan, and Wu 2018), which is a previous state-of-the-art MRC method on SQuAD dataset.

Specifically, we first map each word into the vector space by concatenating its word embedding and CNN-based character embedding. Then we use bi-directional LSTM (BiLSTM) to encode the question $\mathbf{Q}$ and documents $\{\mathbf{D}_i\}$ as:

$$\begin{aligned}\mathbf{u}_t^Q &= \mathrm{BiLSTM}_Q(\mathbf{u}_{t-1}^Q, [\mathbf{e}_t^Q, \mathbf{c}_t^Q]) \\ \mathbf{u}_t^D &= \mathrm{BiLSTM}_D(\mathbf{u}_{t-1}^D, [\mathbf{e}_t^D, \mathbf{c}_t^D])\end{aligned} \tag{1}$$

where $\mathbf{e}_t$ and $\mathbf{c}_t$ are the word embedding and character embedding of the t^{th} word. $\mathbf{u}_t^Q$ and $\mathbf{u}_t^D$ are the encoding vectors of the t^{th} word in $\mathbf{Q}$ and $\mathbf{D}$, respectively.

After the encoding, we use the co-attention method to effectively incorporate the question information into the document context, and obtain the question-aware document representation $\tilde{\mathbf{u}}_t^D = \sum_j \alpha_{tj} \cdot \mathbf{u}_j^D$. We adopt the attention function used in DrQA (Chen et al. 2017a), which computes the attention score α_{ij} by the dot products between nonlinear mappings of word representations:

$$\alpha_{ij} = \mathrm{softmax}(\mathrm{ReLU}(\mathrm{W_l^\top u_i^Q})^\top \cdot \mathrm{ReLU}(\mathrm{W_l^\top u_j^D})) \tag{2}$$

where W_l is a linear projection matrix, $\mathrm{softmax}$ is the normalization function, and ReLU is the nonlinear activation function.

To combine the original representation $\mathbf{u}_t^D$ and the attention vector $\tilde{\mathbf{u}}_t^D$, we adopt the fusion kernel used in (Wang, Yan, and Wu 2018) for better semantic understanding:

$$\mathbf{v}_t^D = \mathrm{Fuse}(\mathbf{u}_t^D, \tilde{\mathbf{u}}_t^D) \tag{3}$$

where the fusion kernel $\mathrm{Fuse}(\cdot,\cdot)$ is actually a gating layer to combine two representations, we do not give the details here due to space limitation.

To model the long distance dependency issue of document context, we also introduce the self-attention layer to further align the document representation $\mathbf{v}_t^D$ against itself, as:

$$\begin{aligned}\beta_{ij} &= \mathrm{softmax}(\mathbf{v}_i^D \cdot W_s^\top \cdot \mathbf{v}_j^D) \\ \tilde{\mathbf{v}}_t^D &= \sum_j \beta_{tj} \cdot \mathbf{v}_j^D \\ \mathbf{d}_t^D &= \mathrm{Fuse}(\mathbf{v}_t^D, \tilde{\mathbf{v}}_t^D)\end{aligned} \tag{4}$$

where W_s is a trainable bilinear projection matrix. Another fusion kernel is again used to combine the original and self-attentive representations. For all the previous encoding and attention steps, we process each document independently given the question. Finally, we obtain a question-aware representation $\mathrm{D^{D_i}} = \{\mathbf{d}_\mathrm{t}^{\mathrm{D_i}}\}$ for each word in each document.

For the question side, since it is generally short, we directly self-align the question to a vector $\mathbf{r}^Q$, which is independent from the document, as

$$\begin{aligned}\gamma_t &= \mathrm{softmax}(\mathbf{w_q^\top} \cdot \mathbf{u_t^Q}) \\ \mathbf{r}^Q &= \sum_t \gamma_t \cdot \mathbf{u}_t^Q\end{aligned} \tag{5}$$

where $\mathbf{w}_q$ is a trainable linear weight vector.

The shared question and document modeling lay the foundation for the subsequent three extraction tasks. Based on the document and question representations $\mathrm{D^{D_i}} = \{\mathbf{d}_\mathrm{t}^{\mathrm{D_i}}\}$ and $\mathbf{r}^Q$, we introduce the three joint extraction tasks.

Document Extraction In multi-document MRC, in addition to annotating the answer span, the benchmark datasets generally also annotate which documents are correct for extracting the answer, or it can also be easily obtained given the labeled answer. Therefore, we also introduce an auxiliary document extraction task, to help improve the answer prediction. Compared to the answer span extraction, the document extraction is relatively easier. The aim is to better lay the foundation for the answer prediction and help learn the shared bottom layers.

Firstly, we also self-align the document representation $\mathrm{D^{D_i}} = \{\mathbf{d}_\mathrm{t}^{\mathrm{D_i}}\}$ for each selected document $\mathbf{D}_i$, to obtain a weighted document vector $\mathbf{r}^{D_i}$ as:

$$\begin{aligned}\mu_t &= \mathrm{softmax}(\mathbf{w_d^\top} \cdot \mathbf{d_t^{D_i}}) \\ \mathbf{r}^{D_i} &= \sum_t \mu_t \cdot \mathbf{d}_t^{D_i}\end{aligned} \tag{6}$$

Next, the question vector $\mathbf{r}^Q$ and document vector $\mathbf{r}^{D_i}$ are matched in a bilinear function for a relevance score as,

$$s^{D_i} = \mathbf{r}^Q \cdot W_{qd} \cdot \mathbf{r}^{D_i} \tag{7}$$

where W_{qd} is a trainable bilinear projection matrix, which helps to match the two vectors in the same space.

For one question, each selected document $\mathbf{D}_i$ has a matching score s^{D_i}. We normalize their scores and optimize the following objective function:

$$\tilde{s}^{D_i} = 1/(1+\exp^{-s^{D_i}}) \tag{8}$$

$$\mathcal{L}_{DE} = -\frac{1}{K}\sum_{i=1}^{K}[y^{D_i} log\tilde{s}^{D_i} + (1-y^{D_i}) log(1-\tilde{s}^{D_i})] \tag{9}$$

where K is the number of selected documents. $y^{D_i} \in \{0,1\}$ denotes the label, $y^{D_i}=1$ means document $\mathbf{D}_i$ contains one golden answer, otherwise $y^{D_i}=0$.

Paragraph Extraction In general, the golden answer usually comes from one or two paragraphs in each document. We can also annotate the correct paragraphs where the answer is extracted from, by some distant supervision method (Chen et al. 2017a). Therefore, we introduce a mid-level paragraph extraction task, so that our model can not only distinguish among different documents, but it can also select the relevant paragraphs within each document.

We first organize each selected document with paragraphs, and follow the same way as in document extraction to calculate a question-paragraph matching score for each paragraph. Specifically, for each paragraph in document $\mathbf{D}_i$ with $\mathrm{D^{Di}} = \{\mathrm{V^{P_{i1}}}, \cdots, \mathrm{V^{P_{iN}}}\}$, we first self-align the word-level paragraph representation $\mathrm{V^{P_{ij}}}$ to a weighted vector representation $\mathbf{r}^{P_{ij}}$ as in Equ. 6. Then a bilinear matching function is used between $\mathbf{r}^Q$ and $\mathbf{r}^{P_{ij}}$ to obtain the corresponding relevance score as:

$$s^{P_{ij}} = \mathbf{r}^Q \cdot W_{qp} \cdot \mathbf{r}^{P_{ij}} \tag{10}$$

where W_{qp} is the trainable bilinear projection matrix between question and paragraph.

For one document, each paragraph $\mathbf{P}_{ij}$ in the document has a matching score $s^{P_{ij}}$. We normalize the scores among each document and obtain $\tilde{s}^{P_{ij}}$ as in Equ. 8. In this sub-task, we optimize the average cross-entropy loss among all the selected documents and paragraphs as:

$$\mathcal{L}_{PE} = -\frac{1}{K}\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}^{N}[y^{P_{ij}} log\tilde{s}^{P_{ij}} + (1-y^{P_{ij}})log(1-\tilde{s}^{P_{ij}})] \tag{11}$$

where N is the number of remaining paragraphs for each document. $y^{P_{ij}} \in \{0, 1\}$ denotes the paragraph-level label for the j^{th} paragraph in i^{th} document.

Answer Span Extraction The ultimate goal is to predict a correct answer, where the afore-mentioned document extraction and paragraph extraction actually act as two auxiliary tasks, so that the shallow semantic representations can be better learnt. In this stage, we aim to combine all the available information to accurately extract the answer from all the selected documents at a span level. To make the document representation aware of information in different documents and enable a direct comparison across different documents, we concatenate all the selected documents together and introduce a muilti-document shared LSTM layer for contextual modeling as:

$$\mathbf{g}_t^D = \mathrm{BiLSTM}(\mathbf{g}_{\mathrm{t-1}}^{\mathrm{D}}, [\mathbf{d}_{\mathrm{t}}^{\mathrm{D}}; \mathbf{r}^{\mathrm{Q}}; \mathbf{f}]) \tag{12}$$

where $\mathbf{f}$ is a manual feature vector including the popular features such as whether each document word occurs in the question words and whether the word is a sentence ending separator. Here we also concatenate the question vector $\mathbf{r}^Q$ to each word representation $\mathbf{d}_t^D$ of the document for better modeling the interaction.

Since all the words from different documents will be passed to the shared LSTM layer, the sequence order is thus very important. We follow the document ranking order obtained via the document ranking function in document retrieval module, as is shown in top of Figure 2. In this way, we expect that the answer prediction model can also bear the ranking relevance in document retrieval module in mind and it shows good performance in our experiment.

Finally, the pointer network (Wang and Jiang 2016) is used to predict the start and end position of the answer with the probabilities α_t^1 and α_t^2, and the answer extraction model can be trained by minimizing the negative log probabilities of the true start and end indices:

$$\alpha_t = \exp(\mathbf{w}_a^\top \mathbf{g}_t^D) / \sum_{j=1}^{|D_w|} \exp(\mathbf{w}_a^\top \mathbf{g}_j^D) \tag{13}$$

$$\mathcal{L}_{AE} = -\frac{1}{M}\sum_{i=1}^{M}(log\alpha_{y_i^1}^1 + log\alpha_{y_i^2}^2) \tag{14}$$

where $\mathbf{w}_a$ is a trainable vector, $|D_w|$ is the total number of words. M is the number of question samples, y_i^1, y_i^2 are the golden start and end positions across the entire documents.

Joint Training and Prediction According to the design, the three extraction tasks share the same embedding, encoding and matching layers. Therefore, we propose to train them together as multi-task learning. The joint objective function is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{AE} + \lambda_1 \mathcal{L}_{DE} + \lambda_2 \mathcal{L}_{PE} \tag{15}$$

where λ_1 and λ_2 are two hyper-parameters that control the weights of those tasks.

To keep the training process stable, we adopt a coarse-to-fine joint training strategy and progressively finetune one upper task with the joint loss. Specifically, we first train the downside document extraction and paragraph extraction tasks to obtain an initial shallow representation, and then jointly train the three tasks with Equ.15 based on it. Besides, when training a new upper task, we follow the method in (Hashimoto et al. 2016) and introduce a successive regularization term on the shared parameters, as:

$$\mathcal{L} = \mathcal{L} + \delta||\theta_s - \theta_s'||^2 \tag{16}$$

where θ_s, θ_s' are the shared parameters at successive training stages. In this way, we can restrain the joint training process so that the shared parameters will not change so much.

When predicting the final answer, we take the document matching score, paragraph matching score and answer span score into consideration and choose the answer with the highest prediction score, given as:

$$s = (\alpha_k^1 \cdot \alpha_k^2) \cdot \tilde{s}^{D_i} \cdot \tilde{s}^{P_{ij}} \tag{17}$$

Experiments

This section presents the experimental methodology. We first verify the effectiveness of our model on two benchmark datasets: TriviaQA (Joshi et al. 2017) and DuReader (He et al. 2017). Then we test our model in operational online environment, which can stably and effectively serve different scenarios promptly.

Datasets

Off-line Benchmark Dataset We choose the TriviaQA Web and DuReader benchmark datasets to test our method, since both of them are multi-document MRC datasets which is more realistic and challenging.

TriviaQA is a recently released large-scale multi-document MRC datasets, which consists of 650K context-query-answer triples. There are 95K distinct question-answer pairs, which are authored by Trivia enthusiasts, with

6 evidence documents (context) per question on average, which are generated from either Wikipedia or Web search. In this paper, we focus on the TriviaQA Web dataset, which contains more context data for each question.

DuReader is so far the largest Chinese MRC dataset, which contains 200K questions, 1M documents and more than 420K human-summarized answers. All the questions and documents are extracted from real data, by the largest Chinese search engine Baidu. The average length of the documents is 396.0 words, and on average each question has 5 evidence documents, each document has about 7 paragraphs.

On-line Environment We also apply our model to the AliMe Chatbot system, which is an intelligent online assistant designed for creating an innovative online shopping experience in e-commerce. Currently, it serves millions of customer questions per day. We test our model in two practical scenarios, i.e., e-commerce promotion and tax policy reading. E-commerce promotion scenario is about consulting instructions on shopping games and sales promotion, which mostly involves with a short document with no more than 500 words. Tax policy scenario is about reading tax policy articles, which can be viewed as a multi-document MRC task. The length of the article is much longer, which consist of many sections and paragraphs.

Implementation Details

For the cascade ranking functions, the number of selected documents K and paragraphs N are the key factors to balance the effectiveness and efficiency trade-off. We choose $K = 4$ and $N = 2$ for the good performance when evaluating on the dev set. Since the TriviaQA documents often contain many small paragraphs, we also restructure the documents by merging consecutive paragraphs to a maximum size of 600 words for each paragraph as in (Clark and Gardner 2017). The detailed analysis will be given and discussed in the next section.

For the multi-task deep attention framework, we adopt the Adam optimizer for training, with a mini-batch size of 32 and initial learning rate of 0.0005. We use the GloVe 300 dimensional word embeddings in TriviaQA and train a word2vec word embeddings with the whole DuReader corpus for DuReader. The word embeddings are fixed during training. The hidden size of LSTM is set as 150 for TriviaQA and 128 for DuReader. The task-specific hyper-parameters λ_1 and λ_2 in Equ. 15 are set as $\lambda_1 = \lambda_2 = 0.5$. Regularization parameter δ in Equ. 16 is set as a small value of 0.01. All models are trained on Nvidia Tesla M40 GPU with Cudnn LSTM cell in Tensorflow 1.3.

Off-line Evaluation

Main Results The results of our single deep cascade model [4] on TriviaQA Web and DuReader 1.0 are summarized in Table 1 and Table 2, respectively. We can see that by adopting the deep cascade learning framework, the proposed model outperforms the previous state-of-the-art methods by an evident margin on both datasets, which validates the effectiveness of the proposed method in addressing the challenging multi-document MRC task.

[4]We only submit the single model without any model ensemble.

Table 1: Performance of our method and competing models on the TriviaQA Web leaderboard.

	Full	Verified
Model	**EM / F1**	**EM / F1**
BiDAF Baseline (Joshi et al. 2017)	40.74 / 47.05	49.54 / 55.80
Smarnet (Chen et al. 2017b)	40.87 / 47.09	51.11 / 55.98
M-Reader (Hu, Peng, and Qiu 2017)	46.65 / 52.89	56.96 / 61.48
Re-Ranker (Wang et al. 2017b)	63.04 / 68.53	69.70 / 74.57
S-Norm (Clark and Gardner 2017)	66.37 / 71.32	79.97 / 83.70
Weissenborn (Weissenborn 2017)	67.46 / 72.80	77.63 / 82.01
Our-Single	**68.65 / 73.07**	**82.44 / 85.35**

Table 2: Performance on the DuReader 1.0 test set.

Model	**BLEU-4**	**ROUGE-L**
Match-LSTM (Wang and Jiang 2016)	31.8	39.0
BiDAF (Seo et al. 2016)	31.9	39.2
PR + BiDAF (Wang et al. 2018)	37.55	41.81
Cross-Passage Verify (Wang et al. 2018)	40.97	44.18
R-net (Wang et al. 2017c)	44.88	47.71
Our-Single	**49.39**	**50.71**
Human Performance	56.1	57.4

Ablation Study To get better insight into our model architecture, we conduct an in-depth ablation study on the development set of DuReader and TriviaQA, which is shown in Table 3. The main goal is to validate the effectiveness of the critical components in our architecture including the manual features and multi-document shared LSTM in the pure answer span extraction task, the cascade document and paragraph ranking functions for pruning irrelevant document content and the adoption of multi-task learning strategy.

From the results, we can see that: 1) the shared LSTM plays an important role in answer extraction among multiple documents, the benefit lies in two parts: a) it helps to normalize the content probability score from multiple documents so that the answers extracted from different documents can be directly compared; b) it can keep the ranking order from document ranking component in mind, which may serve as an additional signal when predicting the best answer. By incorporating the manual features, the performance can be further improved slightly. 2) Both the preliminary cascade ranking and multi-task answer extraction strategy are vital for the final performance, which serve as a good trade-off between the pure pipeline method and fully joint learning method. By removing the rich irrelevant noisy data in the cascade document and paragraph ranking stage, the downside MRC model can better extract the answer from the more relevant content data. Jointly training the three extraction tasks can provide great benefits, which shows that the three tasks are actually closely related and can boost each other with shared representations at bottom layers.

Effectiveness v.s. Efficiency Trade-off Now we further examine how the performance of our model changes with respect to the number of selected documents and paragraphs in cascade ranking stage, which is the key factor to control the effectiveness and efficiency trade-off. The result on

Table 3: Ablation study on model components.

Model	DuReader		TriviaQA	
	Bleu-4 score	Δ	F1	Δ
Complete Model	**50.8**	–	**73.8**	–
w/o Manual Features	49.8	-1.0	73.0	-0.8
w/o Shared LSTM	48.7	-2.1	70.5	-3.3
w/o Cascade Ranking	47.0	-3.8	71.1	-2.7
w/o Multi-task Learning	48.5	-2.3	70.9	-2.9
Boundary Baseline	41.0	-9.8	61.5	-12.3

Table 4: Effectiveness and efficiency w.r.t document and paragraph selection number on DuReader development set (Efficiency is indicated by time cost at prediction stage).

Document No.	Paragraph No.	Time cost(s) / batch	Bleu-4 score
1	1	0.42	32.1
	2	0.53	35.4
	3	0.69	36.0
2	1	0.56	40.5
	2	0.89	44.8
	3	1.14	44.2
3	1	0.71	48.1
	2	1.09	49.5
	3	1.36	49.0
4	1	0.88	49.6
	2	1.39	**50.8**
	3	1.75	50.0
5	1	0.98	49.6
	2	1.70	50.0
	3	2.03	48.8

DuReader development set is presented in Table 4. We can see that: 1) By properly taking more documents or paragraphs into consideration, the performance of the model gradually increases when it reaches 4 documents and 2 paragraphs, and then the performance decreases slightly which may be due to that much noisy data is introduced. 2) The time cost can be largely reduced by removing more irrelevant documents and paragraphs in the cascade ranking stage, while keeping the performance not change that much. For example, for the best setting at 4 documents and 2 paragraphs, if we instead only keep the top-1 paragraph for each document, the time cost will be reduced by 36.7%, while the performance only decreases about 2.4%. As a result, we can adaptively change our model to meet the practical situation and we choose 4 documents and 2 paragraphs in our off-line experiment where effectiveness is most emphasized.

Advantage of Multi-task Learning Next, we also analyze the benefits brought in via the adoption of the multi-task learning strategy in detail. The performance of jointly training the answer extraction module with different auxiliary tasks on DuReader development set is shown in Table 5. We can see that by incorporating the auxiliary document extraction or paragraph extraction task in the joint learning framework, the performance can always improve which again shows the advantage of introducing auxiliary tasks for helping to learn shared bottom representations. Besides, the performance gain by adding document extraction task is larger, which may be due to that it can better lay the foundation of the model with that information from different documents can be distinguished.

Table 5: Performance with different extraction tasks.

Task	Bleu-4 score	Δ
Pure Answer Span Extraction	48.5	–
+ Document Extraction	49.7	+1.2
+ Paragraph Extraction	49.2	+0.7
+ Document & Paragraph Extraction	50.8	+2.3

Table 6: Performance and response time (RT) in two real-world online scenarios.

	Tax	E-Commerce
Model	**F1 / RT**	**F1 / RT**
BiDAF (Joshi et al. 2017)	40 / 130ms	63 / 65ms
DrQA (Chen et al. 2017a)	46 / 122ms	67 / 61ms
Our-Single (w/o Cascade Ranking)	55 / 138ms	71 / 70ms
Our-Single (K=3, N=1)	**76.5 / 45ms**	**73 / 38ms**

On-line Evaluation

Results on E-commerce and Tax data We also test the effectiveness and efficiency of our model in two practical scenarios, E-commerce and tax policy reading, where real-time responses are expected and a large number of customers are being served simultaneously. The comparative result is shown in Table 6. We can see that by introducing the cascade ranking stage and keeping the selected number properly, our method can serve the requests with a much higher speed of less than 50ms, especially for tax scenario where the improvement is about 3 times. Besides, the performance with respect to F1 score is also largely improved with the proposed multi-document MRC model, which demonstrates the effectiveness of our method for removing the rich irrelevant noisy content in our online scenario.

Results on Different Document Lengths We further examine how the F1 score and response time change on tax scenario when processing documents with different lengths, ranging from 50 to 2000 words. The result is shown in Figure 3. We can see that without incorporating with the cascade ranking module, the answer extraction module performs rather poorly both in effectiveness and efficiency as the document length increases. In particular, when the document length exceeds 1,000 the total response time increases 3 to 6 times, while for our full cascade model only 15ms more are needed.

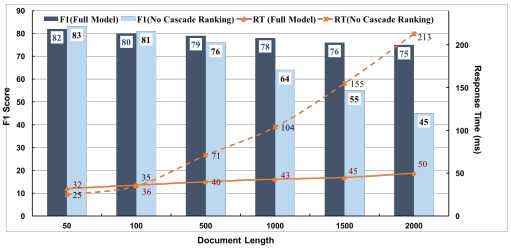

Figure 3: F1 score and average response time w.r.t different document lengths.

Conclusion

In this paper, we propose a novel deep cascade learning framework to balance the effectiveness and efficiency in the more realistic multi-document MRC. We design three cascade modules, which can eliminate irrelevant document content in the earlier stages with simple features and models, and discern more relevant answers at later stages. The experiment results show that our method can achieve state-of-the-art performance on two large-scale benchmark datasets. Besides, the proposed method has also been effectively and efficiently applied in our online system.

References

Bourdev, L., and Brandt, J. 2005. Robust object detection via soft cascade. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, 236–243. IEEE.

Chen, T., and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794. ACM.

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017a. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Chen, Z.; Yang, R.; Cao, B.; Zhao, Z.; Cai, D.; and He, X. 2017b. Smarnet: Teaching machines to read and comprehend like human. *arXiv preprint arXiv:1710.02772*.

Clark, C., and Gardner, M. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Hashimoto, K.; Xiong, C.; Tsuruoka, Y.; and Socher, R. 2016. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*.

He, W.; Liu, K.; Lyu, Y.; Zhao, S.; Xiao, X.; Liu, Y.; Wang, Y.; Wu, H.; She, Q.; Liu, X.; et al. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073*.

Hermann, K. M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, 1693–1701.

Hill, F.; Bordes, A.; Chopra, S.; and Weston, J. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301*.

Hu, M.; Peng, Y.; and Qiu, X. 2017. Reinforced mnemonic reader for machine comprehension. *CoRR, abs/1705.02798*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Lefakis, L., and Fleuret, F. 2010. Joint cascade optimization using a product of boosted classifiers. In *Advances in neural information processing systems*, 1315–1323.

Liu, S.; Xiao, F.; Ou, W.; and Si, L. 2017. Cascade ranking for operational e-commerce search. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1557–1565. ACM.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Schneiderman, H. 2004. Feature-centric evaluation for efficient cascaded object detection. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, II–II. IEEE.

Seo, M.; Kembhavi, A.; Farhadi, A.; and Hajishirzi, H. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Tan, C.; Wei, F.; Yang, N.; Lv, W.; and Zhou, M. 2017. S-net: From answer extraction to answer generation for machine reading comprehension. *arXiv preprint arXiv:1706.04815*.

Tan, C.; Wei, F.; Yang, N.; Du, B.; Lv, W.; and Zhou, M. 2018. S-net: From answer extraction to answer synthesis for machine reading comprehension. In *AAAI*.

Wang, S., and Jiang, J. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

Wang, S.; Yu, M.; Guo, X.; Wang, Z.; Klinger, T.; Zhang, W.; Chang, S.; Tesauro, G.; Zhou, B.; and Jiang, J. 2017a. R 3: Reinforced reader-ranker for open-domain question answering. *arXiv preprint arXiv:1709.00023*.

Wang, S.; Yu, M.; Jiang, J.; Zhang, W.; Guo, X.; Chang, S.; Wang, Z.; Klinger, T.; Tesauro, G.; and Campbell, M. 2017b. Evidence aggregation for answer re-ranking in open-domain question answering. *arXiv preprint arXiv:1711.05116*.

Wang, W.; Yang, N.; Wei, F.; Chang, B.; and Zhou, M. 2017c. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 189–198.

Wang, Y.; Liu, K.; Liu, J.; He, W.; Lyu, Y.; Wu, H.; Li, S.; and Wang, H. 2018. Multi-passage machine reading comprehension with cross-passage answer verification. *arXiv preprint arXiv:1805.02220*.

Wang, L.; Lin, J.; and Metzler, D. 2011. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 105–114. ACM.

Wang, W.; Yan, M.; and Wu, C. 2018. Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 1705–1714.

Weissenborn, e. a. . 2017. Dynamic integration of background knowledge in neural nlu systems. *arXiv preprint arXiv:1706.02596*.

Yu, A. W.; Dohan, D.; Luong, M.-T.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.