文章编号：1003-0077(2018)11-0103-09

基于多篇章多答案的阅读理解系统

刘家骅[1,2]，韦 琬[2]，陈 灏[2]，杜彦涛[2]

（1. 清华大学 计算机系，北京 100084；
2. 北京奇点机智科技有限公司，北京 100080）

摘 要：机器阅读理解任务一直是自然语言处理领域的重要问题。2018 机器阅读理解技术竞赛提供了一个基于真实场景的大规模中文阅读理解数据集，对中文阅读理解系统提出了很大的挑战。为了应对这些挑战，我们在数据预处理、特征表示、模型选择、损失函数的设定和训练目标的选择等方面基于以往的工作做出了对应的设计和改进，构建出一个最先进的中文阅读理解系统。我们的系统在正式测试集 ROUGE-L 和 BLEU-4 上分别达到了 63.38 和 59.23，在 105 支提交最终结果的队伍里面取得了第一名。

关键词：机器阅读理解；问答系统；深度循环神经网络

中图分类号：TP391 **文献标识码**：A

Machine Reading Comprehension for Multi-document and Multi-answer

LIU Jiahua[1,2], WEI Wan[2], CHEN Hao[2], DU Yantao[2]

(1. Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China;
2. Naturali Ltd, Beijing 100080, China)

Abstract: Machine Reading Comprehension (MRC) has become a popular issue in Natural Language Processing (NLP). The 2018 NLP Challenge on Machine Reading Comprehension provides a large-scale application-oriented dataset for Chinese Machine Reading Comprehension, which is much more challenging than previous Chinese MRC dataset. To cope with those challenges, we present a system with improvements in all aspects, including preprocessing strategy, feature expression, model design, loss function and training criterion. Our system achieves 63.38 in ROUGE-L score and 59.23 in BLEU-4 score on the final test set, ranking first among 105 participating teams.

Keywords: machine reading comprehension; question answering system; deep recurrent neural network

0 引言

机器阅读理解任务是自然语言处理的核心问题。近年来各个数据集的发布，大大推进了机器阅读理解任务的发展。最近，在由中国中文信息学会和中国计算机学会主办，百度公司、中国中文信息学会评测工委会和中国计算机学会中文信息技术专委会承办的 2018 机器阅读理解技术竞赛中，发布了一个大规模的源于真实搜索引擎任务场景的中文阅读理解数据集，这将对中文阅读理解领域的发展产生很大的推动作用。

与其他早前发布的数据集相比，本次比赛的数据有如下特点：(1)来源于真实数据，包含不同类型的问题。数据不但包括事实类问题，还包括大量意见型问题，而意见型问题常常对应多个答案；(2)问题对应的文档以网页全文的形式给出，其长度也大大超过常见阅读理解模型的限制；(3)包括百度搜索和百度知道两部分不同来源的数据，相互间存在很大差异。这些特点决定了这个数据集比以往的机器阅读理解数据集具有更大的挑战性。为了应对这些挑战，我们从数据预处理、特征表示、模型选择、损失函数的设定和训练方法的选择等多个方面对系统进行改进。实验表明，每个部分的改进都对结果的提高做出了贡献。最终，我们的系统在 ROUGE-L 和 BLEU-4 上分别取得了 63.38 和 59.23 的分数，获得了 2018 机器阅读理解技术竞赛的第一名。

收稿日期：2018-06-25 **定稿日期**：2018-07-24

接下来,我们将从竞赛数据集和评价方法、数据预处理、模型结构、实现细节、实验结果和样例分析等方面分节介绍我们的工作,并在结论部分对工作进行总结。

1 竞赛数据集和评价方法

本次竞赛的数据集包括 30 万问题(其中训练集 27 万,开发集 1 万,测试集 2 万),其中 20 万来自先前公开发布的 DuReader 数据集[1],所有问题均为来自搜索引擎场景中用户的真实问题。每个问题提供最多 5 个候选文档以及人工撰写的一个或者多个答案作为参考答案。数据集分为百度搜索和百度知道两部分(以下简称搜索和知道),每一部分各包括 15 万问题。搜索部分的文档来源于百度搜索引擎提供的相关度最高的网页,而知道部分的文档来源于百度知道网站的站内搜索引擎提供的相关度最高的问题对应的网页。与大多数常用的阅读理解数据集不同的是,本次竞赛数据集提供的每个文档都包含来源网页的全文文本内容。

竞赛以测试集的人工标注答案为参考答案,采用 ROUGH-L 为主评价和指标,BLEU4 作为评价指标,其中以 ROUGH-L 作为主评价指标。同时对于是非及实体类型问题,对 ROUGE-L 和 BLEU4 评价指标进行了轻微的改动,适当增加了正确识别是非答案类型及匹配实体的得分奖励,一定程度上弥补传统 ROUGE-L 和 BLEU4 指标对是非和实体类型问题评价不敏感的问题。

2 数据预处理

2.1 文档预处理

由于这次比赛数据集提供的每个文档都包含来源网页的全部文本内容,长度比其他常用数据集长很多,最长的文档包含多达 9 万多个词,大大超出了常用的机器阅读理解模型可以接受的范围(通常为数百词)。因此,我们需要对文档进行预处理,抽取出文档中可能含有答案的部分作为篇章输入,放入模型中进行训练和预测。

对于知道和搜索两部分数据,我们采取了不同的预处理方法。知道的数据来源是百度知道问答中相关问题的对应网页,所有内容都是与网页标题对应的问题(和用户提出的问题不一定相同)相关的信息。基于越前面的信息可能越有用的假设,对于知道数据里面的每一篇文档,我们将文档标题和所有段落顺次连接,中间插入特殊的符号表示段落分割,然后截取最前面不超过预设最大长度的内容,将得到的结果作为预处理的结果。

对于搜索部分的数据,文档包含一般网页的全部文本信息,前面部分的内容有可能与问题没有关系。因此,我们采取如下策略:

(1) 将标题和各段内容以特殊符号分隔后连接在一起,如果得到的结果没有超过预设最大长度,则将其作为预处理的结果;

(2) 否则,我们计算各段落和问题的 BLEU-4 分数,以衡量段落和问题的相关性;

(3) 在分数排名前 k 的段落中,选择最早出现的段落;

(4) 选取标题、这个段落以及下一个段落;

(5) 对于此后第 3 到第 10 个段落,选取每个段落的第一句话;

(6) 将所有选取的内容以特殊符号分隔后连接在一起,截取最前面不超过预设最大长度的内容,将得到的结果作为预处理的结果。

上述方法基于两个假设:第一,答案可能出现在与问题相似的段落后面;第二,越靠前面的内容越重要。引入文档标题的原因在于从文档标题很容易判断文档内容是否与问题相关。

引入上述启发式预处理机制后,采用同样的模型,得到的结果比基线系统提供的简单预处理方法有大幅提高,对此我们将在实验部分具体说明。值得注意的是,数据集提供了未经过分词的原始文本以及分词后的结果,我们的模型选择分词后的结果作为输入。

2.2 参考答案片段

与大多数主流工作相同,我们将阅读理解任务建模成篇章中的片段抽取问题。在数据集中,每个问题可能对应多个人工撰写的答案。对于每个答案,我们在篇章中寻找一个与答案 F1 分数最高的片段,作为这个答案的参考答案片段,在训练时使用。对于仅使用一个答案片段的模型,我们采用与任意一个答案的 F1 分数最高的片段作为参考答案片段训练模型(这种情况与百度提供的基线系统一致)。

3　模型结构

3.1　特征表示层

在现在常见的机器阅读理解模型中，词向量是最常用的特征表示方法。以往的工作表明，使用预先训练好的词向量作为输入，并在阅读理解任务的训练过程中固定保持词向量不变，相比于使用随机初始化的词向量，并在训练过程中同步训练词向量，效果会有提升。因而，我们的模型也使用相同的方法。

在英文阅读理解模型中，加入基于字符序列得到的词表示对于结果有稳定的提升效果。这是因为英文单词有丰富的词缀变化，语义相同词性不同的单词拥有相同的词干，只在词缀上有所区分，而词缀相同的单词往往词性相同。这样的特性使得引入字符序列的信息能够缓解未登录词的问题。而汉语词汇没有这样的特性，含有相同字的不同词语的意思可能完全没有任何联系。我们也尝试使用循环神经网络整合预训练好的字向量的信息得到词的表示，作为额外的特征表示，但是实验效果并没有明显地变好。

在以往的工作如文献[2]中，除了词向量以外，一些其他的特征也被运用在阅读理解模型当中。我们的模型使用了词性标注信息。我们使用词性标注工具，对每个问题和对应的篇章进行词性标注，得到其中每一个词的词性信息。对于每一种词性，我们预先训练好对应的词性向量。问题和篇章中的每个词对应的词性向量也作为词的特征表示之一。

对于篇章里面的每一个词，我们还使用一个额外的二值特征 wiq，表示这个词有没有在对应的问题中出现。

问题的类型也能够对回答问题提供帮助。2018 机器阅读理解技术竞赛将问题分成三种类型：实体类，描述类和判断类。我们认为这样的划分不够细致，通过简单的关键字匹配，将问题划分成九种常见类型，大致对应英文的各个疑问词。具体的问题分类和对应的关键词信息见表 1。

给定任意一个问题，首先按表 1 的顺序从上到下匹配各个关键词，如果问题包含某一分类的其中一个关键词，则被匹配到该分类。如果问题没有包含上述任意关键词，则认为问题是数据集给定的类型。因此，我们一共将问题分为 12 类（包括表 1 定义的九种类型和数据集本身给定的三种类型），对于每个问题，都会在问题开始加上一个特殊符号表示其对应的问题类型。

表 1　问题分类和对应关键词

问题分类	对应的关键词
＜how long＞	多长时间、多久
＜when＞	什么时候、几月几日、几月、几号、几点、哪一年、哪年、哪个月、哪天、时间
＜where＞	什么地方、在哪、哪里、从哪、到哪、来哪、去哪、地方
＜who＞	什么人、哪些人、谁
＜how much＞	多少钱、多少、价格、票价、费用
＜why＞	为什么、原因
＜what＞	什么、有哪些、区别、排名
＜which＞	哪个、哪
＜how＞	如何、怎样、怎么、方法、攻略

3.2　问题篇章分别编码层

参照已有的工作，我们采取双向 LSTM 模型分别对问题和篇章的输入特征 x_i^q 和 x_j^p 进行编码，得到问题和篇章的编码表示 u_i^q 和 u_j^p，如式(1)、式(2)所示。

$$u_i^q = \mathrm{BiLSTM}(u_{i-1}^q, x_i^q) \tag{1}$$

$$u_j^p = \mathrm{BiLSTM}(u_{j-1}^p, x_j^p) \tag{2}$$

3.3　问题篇章匹配层

分别给定问题和篇章的编码表示 u_i^q 和 u_j^p，匹配层负责融合两方面的信息，生成包含问题和篇章信息的隐层表示。现在主流的模型大多数都利用注意力机制将问题和篇章的编码信息进行融合。我们在模型中尝试使用了 BiDAF、MatchLSTM 和 DCA 三种不同的匹配层。

3.3.1　BiDAF 匹配层

Seo[3]等提出 BiDAF 模型利用双向注意力机制来融合问题和篇章的信息的方法。我们首先通过内积计算出每个问题中的词和每个篇章中的词的相似度，如式(3)所示。

$$s_{ji} = {u_j^p}^{\mathrm{T}} \cdot u_i^q \tag{3}$$

然后利用他们的方法计算出问题到篇章(context2query)和篇章到问题(query2context)的注意力表示，如式(4)～式(7)所示。

$$\mathrm{A} = softmax(\mathrm{S}) \tag{4}$$

$$c_j^p = \sum_{i=1}^{m} a_{ji} u_i^q \tag{5}$$

$$b = softmax(max_col(S)) \tag{6}$$

$$d^p = \sum_{j=1}^{n} b_j u_j^p \tag{7}$$

然后再利用一层双向 LSTM 得到隐层表示，如式(8)所示。

$$h_j^p = \text{BiLSTM}(h_{j-1}^p, [u_j^p; c_j^p; u_j^p \circ c_j^p; u_j^p \circ d^p]) \tag{8}$$

3.3.2 MatchLSTM 匹配层

参照 Wang 和 Jiang[4] 提出的 MatchLSTM 模型，我们首先计算：

$$s_{ji} = v^{\mathrm{T}} \tanh(W_p u_j^p + W_q u_i^q) \tag{9}$$

$$a_{ji} = \text{softmax}(s_{ji}) \tag{10}$$

$$c_j^p = \sum_{i=1}^{m} a_{ji} u_i^q \tag{11}$$

$$d_j^p = \text{BiLSTM}(d_{j-1}^p, [u_j^p; c_j^p; u_j^p - c_j^p; u_j^p \circ c_j^p]) \tag{12}$$

然后再利用一层双向 LSTM 得到隐层表示，如式(13)所示。

$$h_j^p = \text{BiLSTM}(h_{j-1}^p, d_j^p) \tag{13}$$

3.3.3 DCA 匹配层

Hasan 和 Fischer[5] 提出在使用双向注意力机制之后，再使用一层注意力机制进一步融合信息的方法：

$$s_{ji} = u_j^{p\mathrm{T}} \cdot u_i^q \tag{14}$$

$$A = \text{softmax}_{\text{r}}\text{ow}(S) \tag{15}$$

$$c_j^p = \sum_{i=1}^{m} a_{ji} u_i^q \tag{16}$$

$$B = \text{softmax}_{\text{c}}\text{ol}(S) \tag{17}$$

$$d_i^q = \sum_{j=1}^{n} b_{ji} u_j^p \tag{18}$$

$$r_{ji} = c_j^{p\mathrm{T}} \cdot d_i^q \tag{19}$$

$$g_j^p = \sum_{i=1}^{m} r_{ji} d_i^q \tag{20}$$

然后再利用一层双向 LSTM 得到隐层表示，如式(21)所示。

$$h_j^p = \text{BiLSTM}(h_{j-1}^p, [u_j^p; c_j^p; g_j^p]) \tag{21}$$

在实验结果部分，我们会比较不同匹配层对模型性能带来的影响。

3.4 答案抽取层

我们将阅读理解任务建模成篇章片段抽取问题，答案抽取层利用两步的指针网络，以匹配层输出的隐层表示 h_j^p 为输入，分别预测答案片段在篇章中的开始位置和结束位置的概率分布 y_j^1 和 y_j^2：

$$s_{tj} = v_a^{\mathrm{T}} \tanh(W_1 h_j^p + W_2 h_{t-1}^a) \tag{22}$$

$$y_j^t = \text{softmax}(s_{tj}) \tag{23}$$

$$c_t = \sum_{j=1}^{n} s_{jt} h_j^p \tag{24}$$

$$h_t^a = \text{BiLSTM}(h_{t-1}^a, c_t) \tag{25}$$

其中，$t=1,2$。

在答案抽取层，为了给一个问题统一寻找一个答案片段，同一问题对应的不同篇章得到的隐层表示被连接起来，这部分的实现和基线系统相同。

3.5 损失函数

对于片段抽取模型来说，通常采取如下损失函数，如式(26)所示。

$$L = -\frac{1}{N}\sum_{i=1}^{N} (\log y_{a_1}^1 + \log y_{a_2}^2) \tag{26}$$

其中，a_1 和 a_2 分别表示参考答案片段对应篇章中的开始位置和结束位置。

与大多数已有的阅读理解数据集不同，2018 机器阅读理解技术竞赛的数据集为每个问题提供不止一个参考答案。最终评判的时候，只要机器预测的答案与其中一个答案相同或者类似，就会得到比较高的分数。因而，与只考虑一个参考答案的经典模型相比，在训练的时候考虑所有参考答案可以更有效地利用数据，训练出更好的模型。对于给出多个参考答案的问题，我们利用 2.2 节介绍的方法对于每个答案找到它在篇章中的参考答案片段，并定义如下损失函数，如式(27)所示。

$$L = -\frac{1}{N}\sum_{i=1}^{N} \frac{1}{|A_i|}\sum_{k=1}^{|A_i|} (\log y_{a_1^k}^1 + \log y_{a_2^k}^2) \tag{27}$$

其中，a_1^k 和 a_2^k 分别表示第 k 个答案对应片段在篇章中的开始位置和结束位置。

3.6 利用辅助任务联合训练

Tan[6] 等提出了以预测答案从哪个篇章得到的问题作为辅助任务，进行联合训练，提高多篇章阅读理解模型性能的方法。我们借鉴了他们的思路，设计了篇章选择损失函数。

对于匹配层得到的每一个篇章的隐层表示，我们使用“注意力池化”(attention pooling)和投影变化，从而计算每一个篇章和问题的匹配分数 g：

$$s_j = v_1^{\mathrm{T}} \tanh(W_{sp} h_j^p + b) \tag{28}$$

$$a_j = \text{softmax}(s_j) \tag{29}$$

$$r^p = \sum_{j=1}^{n} a_j h_j^p \tag{30}$$

$$g = \text{sigmoid}(v_{sp}^{\mathrm{T}} r^p) \tag{31}$$

然后将辅助任务损失函数 L_{sp} 定义为：

$$L_{sp} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K}\sum_{k=1}^{K}(sel_k \log g_k + (1-sel_k)\log(1-g_k)) \quad (32)$$

其中，sel_k 表示是否有答案片段出自该篇章。

于是，我们使用最大似然估计联合训练优化目标函数，如式(33)所示。

$$J_{\text{MLE}} = L + \lambda_{sp} L_{sp} \quad (33)$$

3.7　利用最小风险训练

前面提出的损失函数使用最大似然估计，目标是最大化训练集中每个问题的参考答案区间出现的概率。与之相比，使用最小风险训练直接对评估指标的期望进行优化，在机器翻译等方向取得了很好的结果。

在阅读理解任务上，最小风险训练的目标函数可以被定义为式(34)。

$$J_{\text{MRT}} = -\frac{1}{N}\sum_{i=1}^{N} E(\Delta(A_i, A_i^*) \mid \theta) \quad (34)$$

其中，A_i 表示预测的答案，A_i^* 表示参考答案，Δ 是衡量两者差异的函数，我们直接使用 ROUGE-L 作为 Δ。

参考前人的工作，进行最小风险训练的时候，我们以最大似然估计训练得到的模型参数，作为初始化参数。在这种情况下，我们发现，联合训练最大似然估计和最小风险训练的目标函数可以得到更好的结果，如式(35)所示。

$$J = J_{\text{MLE}} + \lambda_{\text{MRT}} J_{\text{MRT}} \quad (35)$$

3.8　判断类问题模型

对于判断类问题，给出和参考答案一致的判断结果(Yes，No 或者 Depends)会得到分数的加成。于是我们单独训练了一个模型，对给定问题和提取出来的答案进行分类。模型结构与上面描述的阅读理解主模型类似，只是篇章输入变成了提取出来的答案。模型仅使用预训练词向量作为特征输入，经过相同的编码层和匹配层，得到隐层表示 h_j^p，然后使用“注意力池化”(attention pooling)得到向量表示 r^p：

$$s_j = v_o^{\mathrm{T}} \tanh(W_o h_j^p + b) \quad (36)$$

$$a_j = \text{softmax}(s_j) \quad (37)$$

$$r^p = \sum_{j=1}^{n} a_j h_j^p \quad (38)$$

然后将 r^p 做投影到分类维度，再用交叉熵做损失函数训练分类模型。

4　实现细节

4.1　预训练词向量

我们利用数据集提供的未分词文本和分词结果，使用一层的 LSTM 模型训练了一个分词器，然后用这个分词器将 SogouT 的部分文档进行分词，用分词得到的结果作为输入，以语言模型为训练目标，使用一层的 LSTM 模型训练 256 维的中文词向量。得到的预训练词向量被用在阅读理解模型中，并且在训练阅读理解模型的过程中保持不变。

4.2　模型参数和训练方法

在训练过程中，我们设定每个篇章的最大长度为 500 词，batch size 为 32。为防止模型过拟合，层与层之间采用了 dropout 技术，所有隐层表示向量为 150 维，单模型系统的 dropout 比例设为 0.15。我们使用 $\lambda_{\text{MRT}}=10.0$，在单一模型情况下 $\lambda_{sp}=5.0$。

我们采用 Adam 算法来优化我们的模型，学习率(learning rate)在训练过程中保持 0.001 不变。对于知道部分和搜索部分的数据，我们使用两个模型分别训练和预测。使用最大似然估计训练模型时，我们训练 10 轮(epoch)，每一轮训练之后在开发集测试性能，最终选择开发集上性能最好的模型。使用最小风险训练时，模型以最大似然估计训练得到的参数作为初始化参数进行训练。由于使用最小风险训练速度较慢，且使用最大似然估计训练好的模型参数进行初始化以后，模型通常在第一轮训练结束的时候就达到最好的效果，于是使用最小风险训练时我们只训练 1 轮。

4.3　实验环境和训练时间

实验所用的服务器使用 4 核 Intel(R) Xeon(R) CPU E5-2630 v4 2.20GHz CPU，配有 128GB 内存，显卡为 TITAN X Pascal，显存大小为 12GB。每个实验使用单个 GPU 进行训练。

在搜索部分数据上，BiDAF 模型每一轮数据训练和验证时间约为 3h，MatchLSTM 模型约为 5h，DCA 模型约为 3h。知道部分数据每一轮训练所用时间与搜索部分数据相仿。使用最小风险训练时，由于每次要计算所有片段和参考答案的 Rouge-L 值，训练速度相对缓慢，BiDAF 模型每一轮大概需要 16h。

4.4 预测结果和结果后处理

预测时,我们使用模型预测答案片段在篇章的开始位置和结束位置的概率分布,并寻找同一篇章内使得开始位置概率乘以结束位置概率最大的片段作为预测结果。

值得说明的是,数据集对每个问题提供多个文档,这些文档来源于搜索结果,并按照搜索结果顺序呈现,也就是说搜索系统认为更靠前的文档与问题更相关,而这个信息在我们的系统里面并没有显式体现。尽管模型在训练的时候增加了辅助任务预测答案从哪一个篇章得到,我们发现,对于训练好的模型,在开发集中仅使用前3篇文档进行预测的结果,比使用全部(最多5篇)文档进行预测得到的结果有明显提高。在实验部分我们会给出数据进行具体说明。因此,在最后预测的时候,对于每个问题,我们仅使用前3篇文档进行预测,得到结果。

我们还对提取出来的片段进行简单后处理,包括去除我们在预处理阶段增加的特殊分隔符和"\u3000""\n"等特殊符号,以及对标点符号进行规范化处理。最终提交的结果是经过后处理的结果。

4.5 集成模型

由于实验结果表明,使用BiDAF、MatchLSTM和DCA作为匹配层的效果相近,为了使得集成模型来源的各个模型具有多样性,我们采用了来自不同匹配模型的结果进行集成。我们还使用了不同的dropout比例,不同的联合学习比率训练不同的模型。表2详细介绍了集成结果来源的各个模型的不同部分具体选项。所有18种不同选择分别被用于训练,共得到18个模型,这些模型的结果最后被集成得到最终结果。

表2 集成结果来源的模型选择

项目	选择
匹配模型	BiDAF;MatchLSTM;DCA
dropout比例	0.1;0.15;0.2
联合学习比率 λ_{sp}	4.0;5.0

5 实验结果

5.1 不同匹配模型的选择

在模型结构的匹配层部分,我们介绍了BiDAF、MatchLSTM和DCA三种不同的选择。我们在开发集上比较了选用不同匹配方法的模型结果。表3展示了不同匹配层在搜索和知道上的结果。由于不同模型结果非常接近,为了减少误差,对于每种情况,我们使用两个不同的随机种子训练模型,最终汇报的结果是两个模型结果的平均值。

表3 不同匹配层模型的表现

匹配层	ROUGE-L	
	Search	Zhidao
BiDAF	48.72	56.89
MatchLSTM	48.79	57.03
DCA	48.69	56.79

可以看出,使用不同的匹配层得到的结果非常接近。MatchLSTM得到的结果最好,但是和BiDAF和DCA得到的结果非常接近。因此,在之后的单模型实验中,我们选择使用速度较快的BiDAF匹配层。而在集成模型的时候,为了使来源的各个模型具有多样性,我们使用了BiDAF,MatchLSTM和DCA匹配层得到的不同模型。

5.2 模型其他各部分的贡献

我们从基线系统开始,逐步增加各个部分,验证每个部分对于效果带来的影响。表4总结了这部分的实验结果,其中数据是模型在开发集上没有经过后处理的直接预测结果的表现。值得说明的是,比赛提供的结果评估脚本里计算BLEU4指标的方式造成了每个问题权重不等,答案长的问题实际权重更高,这是不合理的,而计算ROUGE-L的时候每个问题的权重是相同的,且最终比赛结果也选择了ROUGE-L作为主要的评价指标,所以我们汇报实验结果的时候主要关注ROUGE-L的结果。

表4 模型各部分的影响

模型	Search		Zhidao	
	ROUGE-L	Δ	ROUGE-L	Δ
基线系统	30.77	—	45.90	—
+篇章预处理	42.07	+11.30	50.52	+4.62
+预训练词向量	46.98	+4.51	55.02	+4.50
+其他特征	47.15	+0.17	55.67	+0.65
+多个答案	48.75	+1.60	56.97	+1.30
+联合训练	48.76	+0.01	57.20	+0.23
+最小风险训练	49.61	+0.85	58.11	+0.91

从实验结果可以看出，每项改变都使系统性能得到提升。其中，我们提出的启发式篇章预处理的方法非常有效，使用这种方法相对于基线系统的预处理方法有大幅提高，在搜索部分提升了 11.30，在知道部分也提升了 4.62。使用预训练词向量使得搜索和知道部分的结果分别进一步提升了 4.51 和 4.50。进一步增加其他特征（词性标注、问题类型、wiq）会在搜索和知道上分别带来 0.17 和 0.65 的额外提升。与仅利用一个答案相比，在损失函数中考虑多个答案使 ROUGE-L 指标在搜索和知道上分别进一步上升了 1.60 和 1.30。值得说明的是，加上辅助任务进行联合训练后，虽然 ROUGE-L 值没有大幅提高，但是模型训练的稳定性得到了增强。

最终，基于上面的模型，再使用最小风险训练，我们的模型在开发集上搜索和知道的 ROUGE-L 值分别达到 49.61 和 58.11，相比基线系统分别提高 18.84 和 12.21。这是我们的单系统模型在开发集上没有使用后处理的结果。

5.3　仅用前三篇文档进行预测的的结果

如 4.4 节所述，我们在实验中发现，对于同一个训练好的模型，在开发集中仅使用前 3 篇文档得到的篇章进行预测的结果，比使用全部文档进行预测得到的结果有明显提高，表 5 给出了对比的实验结果。

表 5　仅使用前三篇进行预测的结果

匹配层	ROUGE-L	
	Search	Zhidao
使用全部文档	49.61	58.11
仅使用前三篇文档	50.88	59.69

5.4　判断模型的影响

表 6 比较了使用 3.8 节描述的判断模型给判断类问题抽取出来的答案加上判断结果后，对于在线测试集上 ROUGE-L 和 BLEU4 带来的变化。可以看出，加上判断结果后，ROUGE-L 和 BLEU4 分别提高了 0.26 和 0.23。

表 6　判断模型的影响

	ROUGE-L	Δ	BLEU-4	Δ
没有判断结果	60.16	—	58.24	—
加上判断结果	60.42	+0.26	58.47	+0.23

5.5　在线测试集的结果

表 7 显示了我们的单模型和集成模型对在线测试集的实验结果。集成模型的具体设置见实验细节中的集成模型部分。可以看出，相比于单一模型，集成模型在 ROUGE-L 和 BLEU-4 上分别有 1.58 和 2.28 的提高。

表 7　在线测试集结果

	ROUGE-L	Δ	BLEU-4	Δ
单一模型	62.04	—	57.23	—
集成模型	63.62	+1.58	59.51	+2.28

5.6　最终结果

图 1 显示了比赛公布的最终结果，我们的系统在 ROUGE-L 和 BLEU-4 上分别取得了 63.38 和 59.23 的结果，排名第一，分别领先第 2 名 2.39 和 3.30。

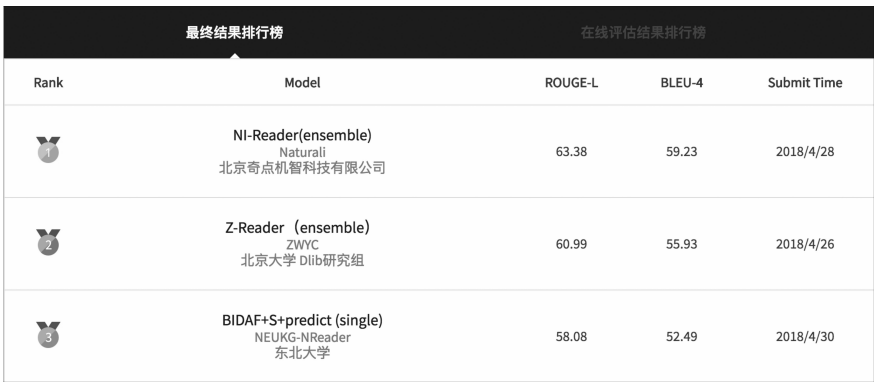


图 1　最终比赛结果

6 样例分析

图 2 给出了一些开发集中我们系统预测答案没有完全和参考答案相符的例子。其中，对于问题 1，我们的预测答案包含了额外的非答案信息。问题 2 的预测答案只包括了参考答案的一部分，没有完整包含应有的信息。问题 3 对应了另外的情况，我们的预测答案是文档里完整包含了答案信息的片段，但是标注者重新整理了这个片段并编写得到了更简洁和合适的参考答案。问题 4 则对应了最差的情况，我们系统预测的结果完全不是回答给定问题的答案。从这些例子可以看出，我们的系统尽管整体上取得了不错的结果，但在一些情况下仍然存在预测的答案不够准确（缺少必要部分，包含不必要的部分），甚至完全错误的情况。一种可能的解决方法是考虑引入答案验证模块，提高答案的准确性。

question 1
夜深人静的意思
answers:
- 夜深人静形容深夜没有人的声响,非常寂静。
- 深夜没有人声，非常寂静。
- 夜深人静的深，是说夜很深，也就是形容入夜时间久，离天亮时间也很长，正是夜色最浓的时候。

pred_answers:
夜间的计时单位，一夜分五更，每更约两小时。形容深夜没有人的声响,非常寂静。

question 2
广州去澳门
answers:
- 在广州去澳门的方法如下：（以广州市政府作为起点）方法一：1、从广州市人民政府步行约430米,到达公园前站2、乘坐地铁2号线,经过12站,到达广州南站3、广州南坐广珠城轨到珠海站下车，出站后往左走三百米左右到达拱北口岸，过关后就到达澳门。方法二：1、从广州市人民政府步行约430米,到达公园前站2、乘坐地铁2号线,经过3站,到达广州火车站3、步行约440米,到达省站4、搭乘直达拱北客运站的大巴车，下车后直走三百多米到达拱北口岸，过关后就到达澳门。方法三：乘坐中旅社的直达巴士，班次密集，约20分钟发一班车，随到随走。广州至澳门（拱北）中旅社直达巴士的上车点：1）华厦大酒店（始发站，最早班次：早上6：30。地铁2号线侨光路海珠广场地铁A出口）2）海涛酒店（途径站，最早班次：早上6：50。黄埔大道与体育东路交界）3）柏丽酒店（地铁8号线C2出口，或者D出口）（接驳车到海涛酒店统一发车）4）东方宾馆（地铁2号线，越秀公园D出口）（接驳车到海涛酒店上大巴）注：大巴在海涛酒店最后站接齐旅客后直接上高速（车程约2小时到拱北口岸）
- 直接广州南坐城轨到珠海站下车，珠海站出来右手边就是拱北口岸，直接过关就是澳门。或者搭乘到拱北客运站的大巴，出了客运站对面就是拱北口岸，拿着通行证就可以直接过去了。
- 广州去澳门的话建议乘坐动车到达珠海北站，珠海北到拱北口岸有通勤大巴车，需要带足量的钱、有效证件与港澳通行证，广州经珠海到澳门时间最久不会超过5个小时。

pred_answers:
1、从广州市人民政府步行约430米,到达公园前站2、乘坐地铁2号线,经过12站,到达广州南站3、广州南坐广珠城轨到珠海站下车，出站后往左走三百米左右到达拱北口岸，过关后就到达澳门。

question 3
机械键盘灯光怎么调
answers:
- 机械键盘灯光般都fn键位加上下调节。
- 每个键盘不一样，键盘右上角找Fn键，配合右角PS，PL这些键。
- FN+↑，FN+↓。

pred_answers:
每个键盘都不一样的，我的雷柏v500s，直接在右上角关掉就可以你的键盘，你可以在键盘右下角找一个Fn键，然后配合右上角的PS，PL这些键试看看。

question 4
芜湖方特哪一期最好玩
answers:
- 1期属于20-30岁年龄段去玩，1期比较刺激。2期属于5-15岁或40-60岁之间去玩，那里比较梦幻，有一种梦幻谷的感觉，女孩子那是不用说了，也有少量的刺激，3D4D电影比较多。
- 一期比较成熟，但是没有侧重点，4d的东西做的还好，二期夜场的烟花值得一看，三期主打水世界，喜欢玩水可以去，四期主打刺激性项目，据说华东第一大过山车，还有灵魂之旅等。
- 方特一期和四期最好玩 。

pred_answers:
方特欢乐世界里面的室外项目比较多，挺刺激的~方特东方神画里面有木质过山车（就是跑男和极限里面都出现过的），然后还有室内跳楼机一类的，也挺刺激的~。

图 2 样例分析

7　结论

2018 机器阅读理解技术竞赛的数据集是基于真实场景的大规模数据集，包含不同类型的问题和不同来源的文档，非常具有挑战性。针对数据集的特点，我们从数据预处理、特征表示、模型选择、损失函数的定置和训练方法的选择等多个方面入手构造系统，在正式测试集中 ROUGE-L 和 BLEU-4 上分别取得了 63.38 和 59.23 的分数，在最终提交结果的 105 支参赛队伍里取得了第一名的成绩。

参考文献

[1] Wei He, et al. DuReader: a Chinese machine reading comprehension dataset from real-world applications [J]. arXiv preprint arXiv:1711.05073, 2018

[2] Minghao Hu, et al. Reinforced mnemonic reader for machine reading comprehension [J]. arXiv preprint arXiv:1705.02798, 2017.

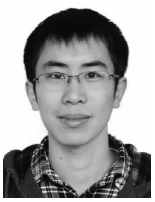
[3] Minjoon Seo, et al. Bidirectional attention flow for machine comprehension [J]. arXiv preprint arXiv: 1611.01603.2016.

[4] Shuohang Wang, Jing Jiang. Machine comprehension using match-lstm and answer pointer//Proceedings of International Conference on Learning Representations [J]. arXiv preprint arXiv: 1608.07905.2016.

[5] Hasan Z, Fischer S. Pay more attention-neural architectures for question-answering [J]. arXiv preprint arXiv: 1803.09230.2018.

[6] Chuanqi Tan, et al. S-net: From answer extraction to answer synthesis for machine reading comprehension [J]. arXiv preprint arXiv: 1706.04815.2017.

刘家骅(1990—)，博士研究生，主要研究领域为语音识别、机器阅读理解。
E-mail: alphaf52@gmail.com



陈灏(1992—)，学士，主要研究领域为中文分词、词向量表示。
E-mail: hao.chen@naturali.io

韦琬(1993—)，学士，主要研究领域为语音识别、机器阅读理解。
E-mail: wan.wei@naturali.io