机器阅读理解大赛总结

摘要

机器理解(MC)，即给定问题和上下文，找出能解决问题的上下文段内容，需要在上下文和查询之间建立复杂的交互。借由 2018 年阅读理解技术竞赛这一机会，我们构建了阅读理解系统，并在 1000 多支报名参赛团队中取得 BLEU-4 评分排名第 6，ROUGE-L 评分排名第 14 的成绩。在本文中，我们介绍了使用双向注意力流神经网络(BiDAF+Passage Self-Matching)和 em 算法构建的机器阅读理解系统。该系统先使用双向注意力流机制获得‘问题-文章’感知的张量表示。通过将该张量进行融合、自匹配、pointer net 解码等过程，我们从文章中截取出候选答案。由于问题下往往有多篇文章，每篇文章找出一个答案便构成了候选答案集。我们将从候选答案集中得到最佳答案这一任务，看作是分类问题。针对该分类任务，我们通过 em 算法进行候选答案间的信息交互，以获取候选答案作为正确答案的置信概率。最后，将神经网络的伴生向量和 em 的候选答案置信概率都输入到 xgboost，以标注出最优答案。

一、比赛介绍

机器阅读理解(Machine Reading Comprehension)是指让机器阅读文本，然后回答和阅读内容相关的问题。阅读理解是自然语言处理和人工智能领域的重要前沿课题，对于提升机器智能水平、使机器具有持续知识获取能力具有重要价值，近年来受到学术界和工业界的广泛关注。

“2018 机器阅读理解技术竞赛”由中国中文信息学会、中国计算机学会和百度公司联手举办，是目前国内该领域较为顶尖的赛事。竞赛提供了面向真实应用场景的大规模中文阅读理解数据集。

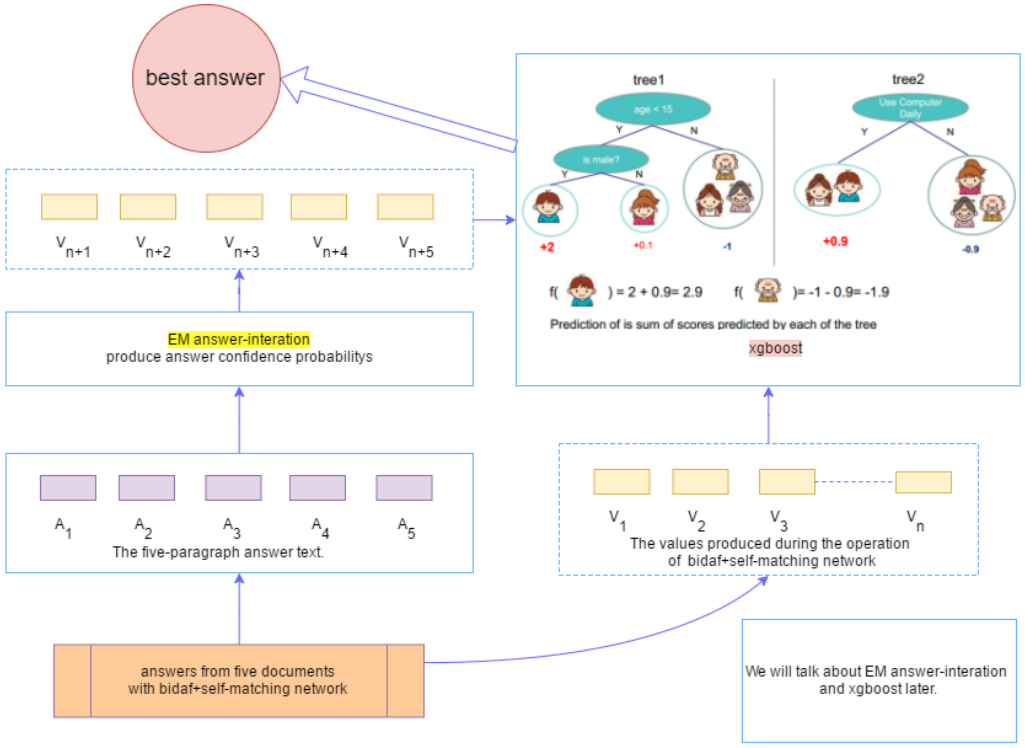
竞赛任务与数据说明

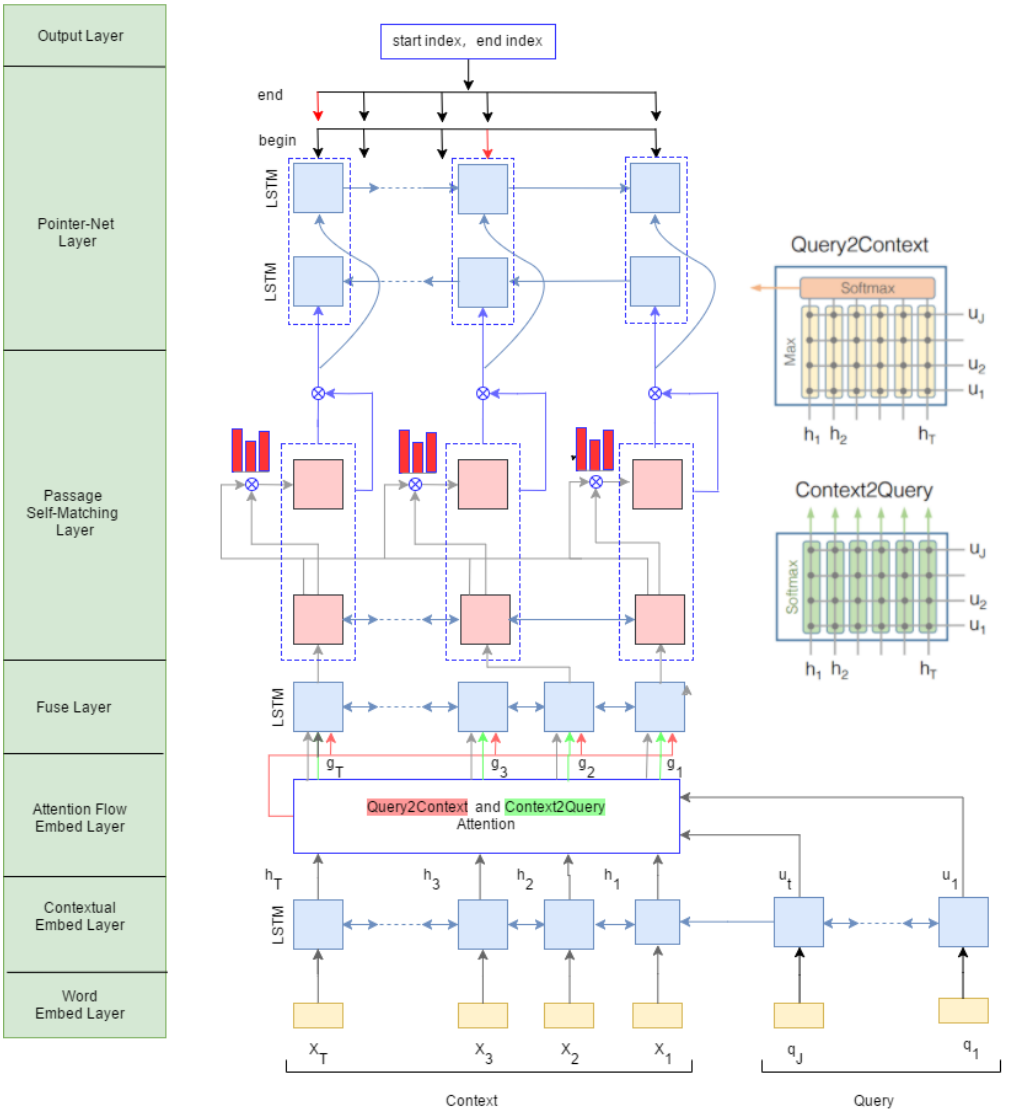
对于给定问题 q 及其对应的文本形式的候选文档集合 D=d1, d2, ..., dn，要求参评阅读理解系统自动对问题及候选文档进行分析，输出能够满足问题的文本答案 a。目标是 a 能够正确、完整、简洁地回答问题 q。

本次竞赛数据集来自搜索引擎真实应用场景，其中的问题为百度搜索用户的真实问题，每个问题对应 5 个候选文档文本及人工整理的优质答案。

二、系统架构

我们的模型采用先从每篇文章中独立抽取候选答案，再从候选答案集中抽取最佳答案的结构，以解决多答案致使神经网络难以学习的问题。架构的具体实现中，我们通过 BiDAF+ Passage Self-Matching 从单篇文章中抽取答案，再使用 em 和 xgboost 决策树从候选答案集中抽取最佳答案。


best answer
tree1
age < 15
Y
N
is male?
Y
N
+2
+0.1
-1
tree2
Use Computer Daily
Y
N
+0.9
-0.9
f( ) = 2 + 0.9= 2.9
f( )= -1 - 0.9= -1.9
Prediction of is sum of scores predicted by each of the tree
xgboost
V n+1 V n+2 V n+3 V n+4 V n+5
EM answer-interation
produce answer confidence probabilitys
A 1 A 2 A 3 A 4 A 5
The five-paragraph answer text.
V 1 V 2 V 3 V n
The values produced during the operation
of bidaf+self-matching network
answers from five documents
with bidaf+self-matching network
We will talk about EM answer-interation
and xgboost later.


Output Layer
Pointer-Net Layer
Passage Self-Matching Layer
Fuse Layer
Attention Flow Embed Layer
Contextual Embed Layer
Word Embed Layer
start index, end index
end
begin
LSTM
Query2Context
Softmax
Max
Context2Query
Softmax
Query2Context and Context2Query Attention
Context
Query

候选答案抽取层——BiDAF+Passage Self-Matching

针对一个问题，文档集里有多答案的情况非常普遍，一边提高某个答案作为答案的概率，另一边又降低其它答案作为答案的概率，这种数据特性严重降低了神经网络的学习效果。交流之中，更是发现其它团队截取的答案常常会出现不完整的情况，即只截断到正确答案的一小部分；而我们的模型却几乎没有出现这种状况，窃以为，这也是由多答案问题导致的结果之一。架构上，我们使用 BiDAF 从单篇文章抽取一个答案的方式，而不是从多篇文章中抽取一个答案，避免了多答案问题带来的弊端。

实现上，相比较一般的 BiDAF，我们的系统有以下两点变化：

1.编码层，问题的编码结果作为文章的上文流入到文章编码中。百度搜索中，问题对应的文章集，是根据检索词获取的；而百度知道中，问题对应的文章集则是根据问题作出回答而得到的。因此，问题其本身也是文章的上文。尤其针对下类：

Q：中国第一个乒乓球大满贯是谁？

A：刘国梁。

A：中国第一个乒乓球大满贯是刘国梁。

A：中国第一个乒乓球大满贯。

很可能第一个答案在 match 层就流失掉了，因为它没有很好匹配问题的每一个词，尤其在先验知识欠缺的情况(如果用先验知识将词向量训练得很好，或许可以匹配到)。但如果用上文流入到文章编码，它能强化文章对问题的匹配程度，而这种强化，也适合于百度搜索、百度知道数据集的数据特性。

2.在 fuse 层之后我们接入 Passage Self-Matching，强化同一篇文章内的信息交互。 Passage Self-Matching 是 R-net 的核心层，它将比较文章中具有相似含义的不同词，以便找到它们的差异之处。使用单纯的 RNN 是很难完成这个任务的，因为这些突出强调的词相距较远。

答案选择层——em 算法

从每一篇文章中都抽取出一个答案，构成了预备答案集。至此，我们面临了一个问题——哪一个答案才是最好的选择？

预备答案集没有被人工标注而无法进行监督训练，而无监督算法又很少能够完成这一步骤的。因此我们最初的想法就是编写新的 em 算法去“猜”。 交互答案之间的信息，来判断每个答案的置信概率。

显而易见，置信概率越高，则某个答案作为正确答案的可能性也就越高。围绕着这一目标，我们进行了设想、推测、证明、实验。先令 N 代表答案总数，n 代表某一答案内词语总数。

假设：

1.答案价值等价于该答案里所有词语的价值总和;

$$V_A = \sum_{j=1}^{n} V_{w_j}$$

2.词语价值等价于所有答案里该词语的贡献价值(即置信度*词语熵/答案总熵)总和;

$$V_w = \sum_{i=1}^{N} T_i * E_w / E_{A_i}$$

其中，IDF 是逆文本频率指数(Inverse Document Frequency)。E_W代表词语信息熵，E_A代表答案信息熵，T_i是答案信息熵归一化后的值。

$$E_W = \lg IDF_W$$

$$E_A = \sum_{j=1}^{n} \lg IDF_{w_j}$$

$$T_i = V_{A_i} / \sum_{k}^{N} v_{A_k}$$

3.答案的平均词语价值越高，则该答案越可信。答案的平均词语价值表示:

$$\overline{V_{w_i}} = T_i / n$$

归一化每个答案的平均词语价值，即为答案在候选答案集中的置信概率:

$$P_{A_i} = \overline{V_{w_i}} / \sum_{j=1}^{N} \overline{V_{w_j}}$$

“猜”的实际步骤：

1、根据信息量初始化答案价值与词语价值:

$$V_A = E_A = \sum_{j=1}^{n} \lg IDF_{w_j}$$

$$V_W = E_W = \lg IDF_W$$

2、E step：

多个答案出现同义词时，我们希望它能提高该答案的置信度；反之，当某个词语仅出现在一个答案中时，它无法为该答案带来置信度的提升。计算词语价值如下：

$$V_w = \sum_{i=1}^{N} T_i * E_w / E_{A_i}$$

E_w / E_{A_i}表示该词语于答案中的信息量占比。词语信息量越大，说明它对答案置信度的贡献就越大。由于常用词的信息量很小，因此，对答案的置信度影响较小。同时，由其带来的答案置信度提升，会被信息熵更大的词语给分摊掉价值。

3、M step：

根据词语价值更新答案价值，如式 1;

$$V_A = \sum_{j=1}^{n} V_{w_j}$$

归一化答案价值：

$$T_i = V_{A_i} / \sum_{k}^{N} v_{A_k}$$

4、总结每个答案的置信概率：

$$\overline{V_{w_i}} = T_i / n$$

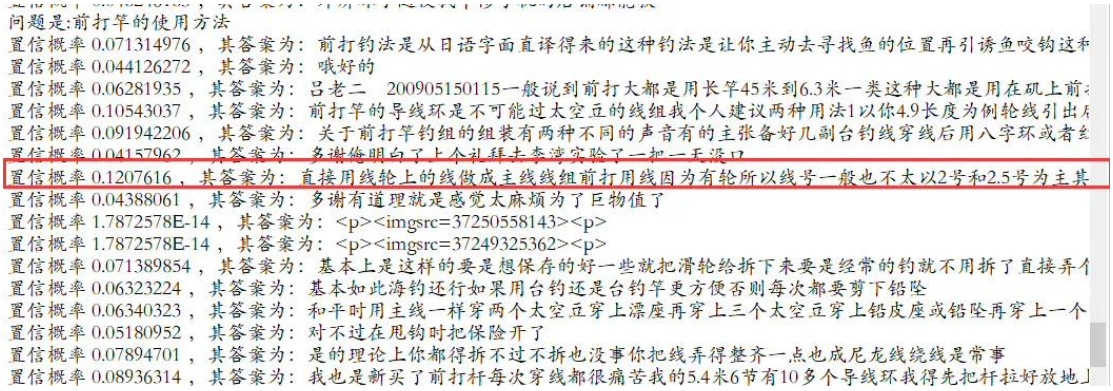
$$P_{A_i} = \overline{V_{w_i}} / \sum_{j=1}^{N} \overline{V_{w_j}}$$

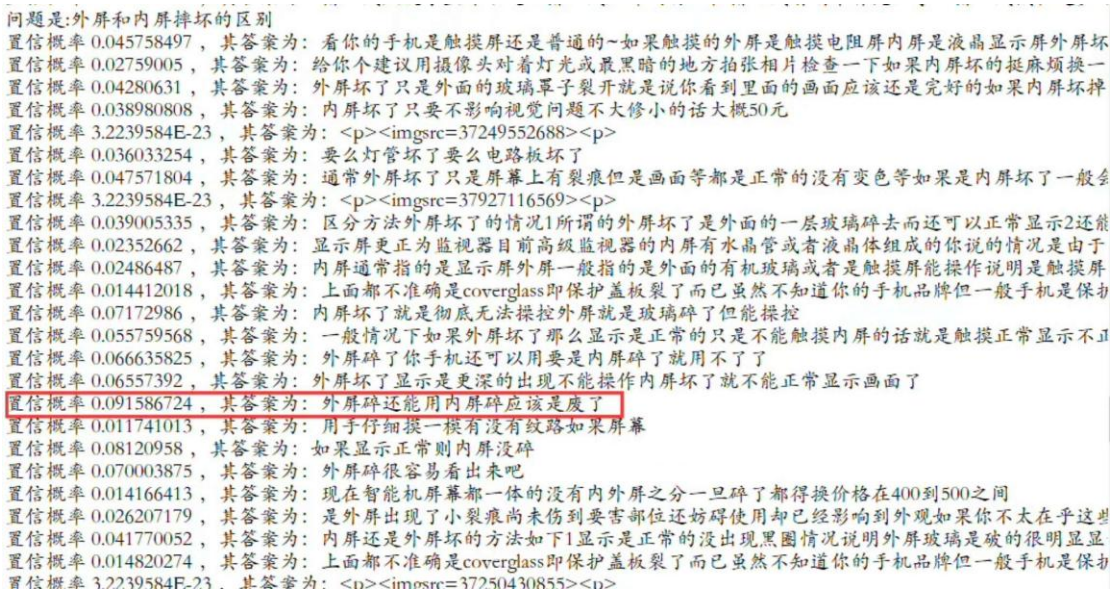
至此，P_{A_i}代表第 i 个答案作为正确答案的概率。

当然，我们还可以更深入地缩放该置信概率，使其避免由于与其它答案无同义词而陷入的无穷小值，或由于与某个别答案太过一致而导致的“垄断”局面（远比其它答案的置信概率要高）。解决方法很多，比如通过概率的概率密度函数来避免概率陷入无穷小或垄断。

考虑到这种缩放可以借由神经网络或 xgboost 来拟合，且比人为定义要更优。故而，暂未继续深入探讨无监督解决该问题。

该 em 算法效果如下：





总结：

对于该算法，答案越多，它能交互到的信息就越客观、越全面，因此效果就越好；经测试，抽取数百条百度知道有十条候选答案以上的问题，人眼观测其结果，往往置信概率最高的都是最优答案，而如果候选答案仅有两三条，那么效果就不太显著了。本次比赛的验证集和测试集普遍是五篇文章，我们的系统从中抽取出五个候选答案，该算法也取得了不错的效果。

实际上，该算法并不仅限于本次阅读理解，它可以用于关键词摘要、文本摘要、舆情分析等应用场景，此外，在各种机器学习算法中，它也可以作为特征输入到算法中，同种模型下，我们模型效果将比他人更好。因此，我认为该算法可作为一种常用型基础算法。

答案选择层——xgboost

em 算法分析了候选答案集之间的交互从而得到了每个答案的置信概率，但是由于**欠缺对‘问题-候选答案’感知的考虑，无法判断答案是否与问题匹配**。除了答案置信概率之外，我们迫切需要融入‘问题-候选答案’感知的特征及其它一些扩充特征(如问题长度、答案长度、问题-答案的编辑距离等)，并寻求一种高效模型来衡量各种特征对答案选择的影响，以期找到更佳的答案。

前文介绍了竞赛数据的情况，每个问题对应 5 个候选文档。我们将从候选答案集中得到最佳答案这一任务，看作是五分类问题。从分类问题出发，实验对比 svm、gdbt、xgboost 等主流模型之后，我们决定使用 xgboost。这是一种基于 boosting 学习方法的决策树模型,它用多个决策树来逐步缩小函数拟合的残差，达到渐进拟合函数的效果，同时它用一个与决策树群复杂程度有关的损失函数找到了较优树结构，避免了欠拟合和过拟合。

我们用训练好的 BiDAF+Passage Self-Matching 模型预测了 6w 组问题，根据候选答案与参考答案的 rouge-l 评分高低，又从每个问题对应的候选答案集标注出得分最高的候选答案作为 xgboost 训练数据的结果标注。预测过程中，我们从匹配层的 answer2question_attn 和 answer2context_attn、自匹配层的答案编码、解码层的 answer_start_prop 和 answer_end_prop 这几个向量分别取总和、均值、最大值、最小值作为 xgboost 的输入特征扩充，以表示‘问题-候选答案’感知。

嵌入‘问题-候选答案’感知，并使用 xgboost 有监督训练之后，我们的机器阅读**系统效果从 Rouge-l 评分 48 上升到 53**。但验证集表明，仅有 42%的概率从 5 个候选答案中选择到最佳答案。如果**全部选择最佳答案，不需要优化其它步骤，验证集 Rouge-l 就将达到 68 分，比本次比赛的第一名还要高出近 5 分**。由于 xgboost 使用的是候选答案抽取过程中的伴生向量,无法很好地代表实际的‘问题-候选答案’感知特征，因此，我认为用神经网络对候选答案集重新编码、匹配、融合、自匹配、解码，将能提高更多性能。

三、测验结果

比赛刚刚落幕，截止目前为止，在海内外以高校、科研院所和企业为主力的 800 多支报名参赛队伍中，G-scuter 的 ROUGE-L 评分排名第 14，BLEU-4 评分排名第 6。但是，我们还有太多想法未来得及付诸实现，因此，这还远不是我们的上限。接下来简单介绍下未来得及落实的优化点。

四、可优化点

正式进入比赛是在 4 月中旬，单次对模型进行的训练耗时四天左右，加上后续的分析调整，导致整个比赛期间仅能对模型进行了三次较大规模的修改。以下是后续进行继续改善的计划：

1. 我们**将训练好的词向量作为神经网络的输入**，取代现行的随机初始化再训练。预计可提高三个点。
2. 在单篇文章中**抽取答案之前，先通过一层自匹配层来交互其他文章的信息**，以提高准确抽取文章有效答案的概率。根据对得分低的预测答案的分析，很多时候是截取到了文章的无效信息，而先交互其它文章，能有效改善这种情况。因此我认为这一改动也将对整个系统的性能有较大提高，效果比词向量更甚。
3. 计划编写一层更适合本模型的神经网络，并将其置于 BiDAF+self-match 运行之后，以抽取正确答案（**取缔直接使用 xgboost 抽取正确答案的方式**）。事实证明使用 xgboost 实非良策，以 em 和 bidaf 的向量作为特征输入到 xgboost，效果也差强人意。
4. 增强数据预处理，**清洗掉 html 标签和爬虫时抓取的噪音信息**。在数据分析的时候，发现存在很多无效数据，如：问题的噪音文字以及文章的 html 标签、爬虫时抓取的噪音等。尤其是问题的噪音文字，我认为对结果影响颇大。
5. 问题中有无疑问代词对结果的确存在着影响，但目前验证来看，还未超过数据量所带来的价值。该部分我们可以继续细化，不再根据问题有无疑问代词拆分数据的同时，**提炼出不同的机制来针对这两种情况进行优化**。
6. **em 算法嵌入同义词典**。目前我们的 em 算法是根据同个词语进行信息交互的，而不是使用同义词。嵌入同义词，em 算法的效果将会提高。
7. **使用 dropout 和模型融合。**大规模的神经网络有两个缺点：费时、容易过拟合。而在训练过程中，使用 dropout 将会使模型按照一定的概率将神经网络单元暂时从网络中丢弃，这不仅减少了训练用时，还增强了模型的泛化能力。该改动预估可将每个评分提高 1.5 点以上。
8. **字符级别词嵌入**。扩充字符级别的词嵌入，能为模型提供更多的特征。

收到的点评及建议

1、 **LSTM 改用 GRU**。能有效减少耗时，同时可能提升效果。
2、 **利用其它方面的知识（语义网、知识图谱、规则）佐证答案的可信度。**这一步骤非常必要，而且也能带来很大提升。Watson。
3、 **信息量级越大，词向量越准确。采取字词联合训练。**几兆、几十兆、几百兆、几千兆的文本数据量，差异非常大。
4、 **训练数据越大，dropout 效果越好，否则可能起到反作用。**
5、 **Em 的同义词典要采用好的，不应采用同义词林。**
6、 **数据预处理，如清洗噪音等操作，或许会对效果起到非常大的帮助。**

五、经验总结与思考

知其所问，对症下药

本次机器阅读理解竞赛，走了不少弯路，但对我而言，依然是一笔宝贵的蕴含思考的财富。我认为，在基于知识图谱和规则的问答领域，它会比较适用。接下来就从知识图谱的结构体系对以下内容进行探讨。

刚接触阅读理解，我想的是先理解问题，再到文章里找答案（实际上目前的神经网络模型并没有真正理解问题）。于是，我开始不断总结它的问题，究竟有哪几种问法，问的是什么内容。总结下来，阅读理解无外乎三类，问属性、问行为、问状态。

一、缺乏疑问内容的弊端

本次比赛中缺乏疑问的内容约占五分之二的比例，就算是人类也是难以决定如何回答的。比如：白玉麻山药价格、优酷播放出现未知错误。因为：

1. 信息缺失。白玉麻山药价格贵吗、价格是多少？为什么优酷播放出现未知错误？优酷播放出现未知错误怎么办？ 对于没有疑问内容的问题，无以明确应该回答哪方面的内容。
2. 颗粒度缺失。白玉麻山药价格贵吗？问两面性，即‘是’与‘否’。白玉麻山药价格是多少？问属性，一个词组。为什么优酷播放出现未知错误？问原因，一段话。
3. 噪音效果。神经网络学习的过程中，有两份问题一致而答案不同的训练语料：

   问题:白玉麻山药价格;

   答案 1:很贵。

   答案 2:15 块。

   它们问题相同，却因提问人意向不一致而选择了不同答案，对神经网络的学习造成噪音。

   考虑另一情况，有两份问题不同而答案一致的训练语料：

   问题 1:白玉麻山药价格;

   问题 2:白玉麻山药价格是多少？

   答案:15 块。

   它们答案一致，但是问题信息不同，容易对‘问题-候选答案’感知造成负面影响。

综上，在基于知识图谱的问答体系中，应先补充疑问内容，明确提问者想了解什么，才方便‘对症下药’。

二、如何补充疑问内容

疑问方式有两种，一种是两面性问题，另一种是由疑问代词统领的问题。

两面性问题

这类问题的数据已经被百度数据划入 yes_no 任务类型，不需要我们再进行划分。但是想辨别两面性问题，我归纳了两种方式：1、[A 不 A]，比如:可不可以，能不能。2、无疑问代词，但是句尾有疑问符号‘?’或者语气词‘吗’之类的标记。

由疑问代词统领的问题

此类问题可细化为三类，问属性、问状态、问行为。

1. 问属性，做依存句法分析，若句法树的根是名词，那么就可以归纳为问属性这一类，属性有：年龄，性别，出生地等。比如：王者荣耀姜子牙大招距离。我们根据句法树自顶而下的寻找知识图谱，先找到【王者荣耀】，再从【王者荣耀】里找到英雄【姜子牙】，依此是【大招】→【距离】→【返回值】。梳理下如何根据三元组(sub,rel,obj)逐层找到查询属性，解决此类问题不难。
2. 问行为，行为有六要素：原因、过程、对象（主客体等）、时间、地点、结果。如果不是问属性的，那么绝大部分都是在问行为，陈述句居多。此类问题也是知识图谱较难解决的，作为整一个句子很难被吸纳到知识图谱体系中。我目前想到的方法，是建设行为图谱的语义网，逐层分类，之后通过某算法（或许可以用 word2vec 的思路，某行为的归属等于该句子的所有词语的加权（如

tfidf）总和再平均）把行为逐一挂靠到对应的类别下。问行为的此类问题，可以通过同一算法找到归属类，再对比该类下的所有行为，找出最相似的行为，最后基于问的是哪一要素进行作答。

3. 问状态，状态是指会因时间而变化的属性或者行为，比如：天气。有的知识是恒定的，不会改变或改变得非常缓慢，但有的知识却会不断改变，状态就属于此类。虽然它也是属性或行为，但拓展了时间维度，过于复杂，需要单独讨论。想要识别此类问题，需要强化问题中的各种条件，以期达到更好的检索效果。

词向量是先验知识

关于词嵌入。以下两种词嵌入有什么优劣，有什么差异？

1、采取随机初始化词向量，反向传播时更新;

2、使用为 w2v、词法特征（名词、动词等）、句法特征（主谓、动宾关系等）、其它特征（tf-idf、lda、空间距离）作为固定词向量。

我们的模型是随机初始化词向量的阅读理解系统。在测试模型的过程中，根据有无疑问代词将 zhidao 集和 search 集拆开，并没有起到想要的提升作用。明明是数据差异如此之大的两个集合，为什么拆分后反而没变化呢?我想，这是数据量增长所带来的价值超越了数据差异带来的影响。

自然语言理解 network，建立于词向量能够准确表达语义的基础上。而无论是 zhidao 集还是 search 集，想要覆盖人类知识都是不可能的，仅仅十来万的数据砸进去甚至翻不起水花，这就造成了随机初始化词向量的阅读理解具有很大局限性。也不是在否定随机初始化词向量的这种做法；毕竟，基于目标和模型得来的词向量，词语特征更能与模型更为相得益彰。甚至说，如果训练样本足够多，涵盖范围广、样本分布合理，直接用初始化词向量就能带来最好的效果。

与此相对应，如果用固定词向量，则它解答训练集所没有的知识的能力会有很大提升，此外固定词向量，能引入一些随机初始化词向量所无法弥补的特征（除非神经网络结构到了不可思议的地步），比如空间特征、词频特征等。

因此我想，将此次 mrc 比赛的数据拆分成四个子集，虽然效果有提升，但没有令我感到应有的效果，问题就在于每个数据集仅有几万的问题量。如 SQuAD 的 r-net，同样的神经网络，最高分能达到 80+，而开源的随机初始化仅 62。这种差异，我认为很大程度是因词向量引起的。

随机初始化词向量，用所有数据跑一次模型，训练出词向量，再拆分成四个子集分别训练（目的是得到更好的神经网络权重，**相当于词向量是知识，而神经网络是解题思路**）。后期考虑引入其它特征向量。

结语

非常感谢我的公司极天信息给我机会与时间参与本次比赛，本次项目开源也是在公司的促使之下。此次以母校‘华南理工大学’的名义参赛，另外非常感谢我的队友及校友吴潘安同学，只有大二，但已经帮到我不少，期待你的未来！

容福炬于 2018 年 8 月 7 日