基于深度学习的机器阅读理解技术研究综述

孙相会

学号：1971654

摘要：　本文主要针对机器阅读理解领域的经典数据集和模型做综述。机器阅读理解 (MRC) 是自然语言处理 (NLP) 领域非常具有挑战性的任务，给定一段文本和与文本有关的几个问题，要求模型能够根据这段文本回答这些问题，也就是让机器向人一样理解文章回答问题。近年来随着深度学习以及 NLP 领域预训练语言模型的发展，在 MRC 方向上的研究已经取得了很大的突破，在很多数据集上模型已经超过了人类的水平。本篇论文是对 MRC 领域做一个概述，主要涉及 (1) 机器阅读理解的任务定义以及不同形式的阅读之间的区别，(2) 一些经典的模型以及目前流行的基于预训练方式的模型 (3) 机器阅读理解领域未来的研究趋势。

关键词: 机器阅读理解；自然语言处理；预训练语言模型

中图分类号: 100　　**文献表示码**: A

A Survey in Machine Reading Comprehension

Xianghui Sun

id: 1971654 (NEU)

Abstract: This article summarizes recent and classic dataset and model in machine reading comprehension.

Key words: machine reading comprehension; what `https://blog.csdn.net/`

1　引言

机器阅读理解 (MRC) 是自然语言处理领域十分重要也是具有挑战性的研究方向，这项任务的目的是衡量计算机理解自然语言文本的能力。具体的就是给定一篇文章和相关的问题，要求计算机通过阅读理解这篇文章后能够正确的回答这些问题。早期的 MRC 系统主要是基于规则和模式匹配的方法，而且数据集规模比较小，系统难以获得期望的性能也不能实际的应用。随着深度学习的兴起，词嵌入技术的发展，注意力机制应用在 NLP 领域 [7] 以及大规模阅读理解数据集如（CNN/Daily Mail [11]，SQuAD [15]，RACE [25]，MS MARCO [30]，CoQA [29] 等）的发布，这些推动了 MRC 领域的发展，越来越多的学者采用神经网络构建 MRC 模型，也叫神经机器阅读理解，效果上显著的优于传统的机器学习方法并且在 SQuAD [15] 数据集上逐渐的接近人类的阅读理解水平。

自 2018 年，随着 ELMo [34]、GPT [35]、BERT [36] 等预训练语言模型的出现，再一次提升了机器阅读理解的水平，特别是 BERT [36] 在 SQuAD 数据集上首次超过了人类的表现。

本篇论文主要从具体任务，数据集，经典的神经机器阅读理解模型，目前流行的基于预训练的模型，对于不同任务的不同的评估指标以及 MRC 领域目前新的趋势几方面对机器阅读理解领域做阐述。

2 机器阅读理解任务概述

机器阅读理解任务是为了使得计算机具有对自然语言文本理解的能力，像人类一样阅读并且理解一篇文章。具体的就是给定一篇文章 P 和一些与文章 P 相关的问题 Q，要求模型通过阅读 P 之后给出 Q 的正确答案 A，即建模给定 P 和 Q 的条件下预测 A 的概率：

$$P(A|P,Q) \tag{1}$$

根据答案形式的不同，任务也是多种多样的，大致可以概括为 4 类：完形填空、多项选择、片段选择和自由回答。下面对这四种任务分别进行叙述并介绍相关的数据集。

2.1 完形填空

完型填空型阅读理解是指给定一篇文章 P 和一个与文章相关的问题 Q，Q 是通过删除掉句子中某一个单词构成，要求模型根据 P 能够正确的填写出 Q 缺失的单词。CNN 和 Daily Mail 数据集 [11] 是由 Google DeepMind 和牛津大学发布于 2015 年，这是第一个较大规模的阅读理解型数据集。从 CNN[1] 中收集 93k 篇文章，从 Daily Mail[2] 上收集 220k 篇文章。每一篇文章的作者都为这篇总结出一些具有概括性的句子，这些句子涵盖了这篇文章的要点。于是把这些概括性的句子删去其中的一个单词，以此作为问题，构建了 (文章-问题-答案) 的三元组形式的语料库作为填空式的阅读理解任务。

2.2 多项选择

多项选择型这类问答任务是对于给定的文 P，以及和 P 相关的问题 Q 和多个候选答案 $A = \{A_1, A_2, \cdots, A_n\}$，从中选择正确的答案，即 $P(A_i|P,Q)$，其中 $A_i \in A$。相关的数据集如 RACE[3]，这个数据集是从中国中学生的英语考试题中建立的数据集。共有将近 28000 篇文章以及 100000 个问题，答案并不是简单的限制于文章中的单词，而且答案和问题中单词可能从没有在文中出现过，因此简单的利用单词匹配方式并不能达到很好的效果。这些问题和候选的答案都是由出题专家生成的，因此更加的接近真实世界的语义。另外 RACE 数据集中文章主题的覆盖度比其它的数据集更广泛，比如 CNN/Daily [11] 所有文章全都是来源于 CNN 新闻，SQuAD [15] 数据集所有的文章全都是来源于维基百科。而 RACE 数据集涵盖多个领域如新闻、故事、广告、传记等等。由于其类型的多样性因此可以更好的评估机器的阅读理解能力。

2.3 片段选择

这类问答任务是 MRC 领域较为流行的研究方向，给出 P 和问题 Q，问题的答案是 P 中的一段连续的单词构成，答案的长度不固定，可以表示为 $P(A|P,Q)$，其中 $A = \{t_i, t_{i+1}, \cdots, t_{i+k}\}(1 \leq i \leq i+k \leq n)$，$n$ 代表 P 中单词的个数。抽取式问答任务最为广泛使用的数据集是 SQuAD 1.1 [15] 和 SQuAD 2.0 [26]。其中 SQuAD 1.1 是 Stanford 问答数据集的第一个版本，由众包工人在维基百科上面的文章中给出问题，答案来源于文章中某段连续的单词。SQuAD 1.1 含有 536 篇文章，总计 107785 个问题-答案对。在 SQuAD 2.0 中又在原有的数据集中加入了 50000 多个没有答案的问题，准确的说这些问题的答案不在相应的文章中。

2.4 自由回答

简单的从文章中摘取一段文本可能并不能回答问题需要的答案，自由回答型任务的答案是自由形式的，不局限于文章中的某些单词，语法上往往是更加的灵活。可以表示为 $P(A|P,Q)$，其中 $A \subseteq P$ 或 $A \nsubseteq P$。从文章中概括提炼出问题的答案也是更加的符合人类的阅读方式的，基于这些原因，自由回答式问答的数据集也因此公布出来并且受到广泛的关注，相关的数据集如 MS MARCO [30]。MS MARCO 是由微软通过在必应搜索引擎的日志上收集用户提出的问题，对于文本段落是来源于必应搜索引擎的返回的搜索结果。具体的就是对于每一个问题，给出 10 个最相关的查询结果的文本段落，然后由标注人员从这 10 个文本段落中找出那些与这个问题有关

[1] www.cnn.com

[2] www.dailymail.co.uk

[3] www.cs.cmu.edu/glail/data/race/

的文本段落，然后人工的从这些选择出来的段落中概括提炼出答案，同时对于选出来的段落要标记为 $is\_select = 1$，表示这个段落和答案相关，从而可以训练模型。

因此可以看到这个数据集与前面的数据集很大的不同之处就是答案是人工生成的，不局限于文本段落中固定的一段单词，因此更加的接近现实世界中的人类阅读理解，对模型的推理能力要求也更高。同时如果不能从给出的 10 个段落中推理出答案，那么这个问题就标记为不可回答的问题，同样要保留在数据集中，目的就是让模型能够判别出问题是否具有答案。

2.5 评估方法

对于不同的 MRC 任务有不同的评估指标。对于填空型任务与多项选择型任务都是属于客观题型，用准确率就可以衡量模型的性能。片段选择型任务属于半客观题型，通常用精确匹配 EM（Exact Match）和 F1 值来评估模型，F1 值是精确率和召回率之间的调和平均数。对于自由回答式任务的答案，一般采用单词水平的匹配率作为评分标准，常用标准有 ROUGE [3]。下面详细介绍这几种评估指标如何评估不同的 MRC 任务。

2.5.1 准确率

准确率可以用来评估完形填空和多项选择这两种类型的任务。对于测试集合中的所有问题 $Q = \{Q_1, Q_2, \cdots, Q_m\}$，其中 m 代表问题的个数。如果模型预测出来的 m 个答案中有 n 个是正确的，那么模型的准确率自然是 n/m。精确匹配 EM 评估指标可以看做是准确率的扩展，就片段选择型任务来讲，EM 要求预测出来的所有单词要和标准答案的所有单词要精确匹配下，EM 值才为 1，否则为 0。因此最后模型在测试集上的 EM 值也就是 n/m。

2.5.2 F1

F1 分数是最为普遍使用的一种评估标准，不仅仅局限于 MRC 的各种任务。F1 值是精确率（precision）和召回率（recall）之间的调和平均数。具体计算公式如下：

$$\mathrm{F1} = \frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{2}$$

精确率是指模型预测的答案中有多大比例的单词是标准答案中的单词。召回率是指标准答案中的单词有多大比例在预测答案中出现。

2.5.3 ROUGE

ROUGE（Recall-Oriented Understudy for Gisting Evaluation）最初是用来评估生成文本摘要的一种方法，因为可以 ROUGE 的计算机制来评估 MRC 领域中自由答案型任务。ROUGE 的评分有多种，在 MRC 领域较为常用的是 ROUGE-L，ROUGE-L 用来计算标准答案和预测答案的最长公共子序列（Longest Common Subsequence，LCS），ROUGE-L 的计算公式如下：

$$R_{LCS} = \frac{LCS(X,Y)}{m} \tag{3}$$

$$P_{LCS} = \frac{LCS(X,Y)}{n} \tag{4}$$

$$F_{LCS} = \frac{(1+\beta)^2 R_{LCS} P_{LCS}}{R_{LCS} + \beta^2 P_{LCS}} \tag{5}$$

其中 $LCS(X,Y)$ 表示标准答案与预测答案之间的最长公共子序列，m 和 n 分别代表标准答案和预测答案中单词的个数，β 是 ROUGE-L 的的参数，用来控制精确率和召回率的重要程度。

3 神经机器阅读理解模型

随着大规模机器阅读理解数据集如 CNN&Daily Mail [11]，SQuAD 1.1 [15] 等的发布以及深度学习技术的发展，神经机器阅读理解模型的性能显著的超过传统的基于规则和特征的模型，随着 NLP 领域预训练模型的发展，基于预训练模型来做 MRC 任务的模型性能再一次的提升。用于 MRC 任务的深度学习模型的整体框架主要包括如下几个层：词嵌入层、语义编码层、语义交互层、答案预测层。

1）词嵌入层：如何将文本有效的表示成计算机可以处理的形式同时可以有效地利用单词之间的语义一直的 NLP 领域的重点问题。分布式表示是将单词用一个低维度的稠密向量表示，即将单词嵌入到一个

低维稠密空间中，因此这种表示方式也叫词嵌入。从早期的 one-hot 形式编码到词袋模型再到分布式表示技术最后到基于上下文的词嵌入技术，每一种技术的出现都证明了一个好的文本表示方法可以极大地提升模型的性能。

2）语义编码层：这一层的目的是在词嵌入层的基础上通过对词嵌入层的输入文本做特征提取，进一步获得句子层面的语义信息。常用的特征提取器有基于 RNN 的变体如 LSTM [1] 和 GRU [?] 等。但由于梯度消失问题不能解决长距离依赖问题，使得其特征提取能力始终受限。Transformer [32] 通过利用自注意力机制取代 RNN 那种序列式的计算方式，通过自注意力机制不仅可以做到单词之间的全局交互同时其并行计算使得模型训练时间大幅减少。实验证明在大规模数据集上 transformer 的特征提取能力要强于基于 RNN 的编码器，目前几乎所有的 NLP 预训练模型都是利用 transformer 作为特征提取器。

3）语义交互层：在预测答案时需要将问题的语义信息与文章的语义信息关联，这样模型在预测答案时才能知道文章中哪一部分是问题的答案。通常利用注意力机制实现这一目标，注意力机制就是让模型关注到重点的部分，不同的注意力计算方式很大程度上的影响模型性能，后面将详细介绍基于注意力机制的模型以及它们不同的计算方式。

4）答案预测层：这是整个模型架构的最后一层，用来输出预测的答案。如前面所提到的 MRC 任务大致分成四类，因此这一层的设计需要考虑到答案形式。对于填空型任务，答案的输出是文章中的一个单词。对于多项选择任务，答案的输出是从多个候选答案中选择出正确的选项。对于片段选择型任务，答案的输出是文章中某段连续的文本。对于自由答案型任务，答案的输出不限固定的文本，而是根据文章中的单词生成的答案。此外还有不可回答的问题，此时模型的输出还要考虑到问题是否可以回答。

下面介绍神经机器阅读理解模型中基于注意力机制的模型和基于预训练模型的模型。

3.1 基于注意力机制的模型

交互层是整个网络模型中关键的一层，前面的编码层输出的是问题和文章中每个单词的上下文语义编码，每个单词关注了自己所在句子的上下文单词，但是却并没有关注对应的句子。而我们在做阅读理解问题时，通常是带着问题去文章中找答案，我们要知道文章中每一个单词和问题之间的相关度。因此交互层的目的就是让文章的语义信息与问题的语义信息融合，以此达到对文章更深层次的理解，而交互层中最常用的方法就是注意力机制。

注意力机制简单的来说就是为序列的每一个位置生成一个权重值，这个值代表当前位置的单词对预测结果的重要性。文献 [7] 最早将注意力机制应用在机器翻译领域，获得了极大的反响。这种注意力的思想就是在解码端每次生成一个单词时要对编码端的句子做关注，也就是注意力的计算，具体的就是利用解码端的单词与编码端序列计算出一个权值分布，这个权值表示的是编码端每一个单词对解码端单词的重要程度，然后利用权值分布对编码端序列加权求和得到一个固定长度的向量，这个向量就是注意力的计算结果，并且作为下一个单词输入的一部分，这属于一种动态式的计算方式。而这种在将注意力计算结果送入到循环神经网络中并且对于循环神经网络的隐藏状态同时参与注意力计算的网络也叫基于注意力的循环神经网络。以下将首先介绍 MRC 任务中注意力机制之间的差异，然后分析各个模型将注意力机制应用在 MRC 任务上以及它们之间的区别与联系。

3.1.1 MRC 中的注意力

不同于机器翻译任务，只能将 encoder 端的单词语义信息融入到 decoder 端。在机器阅读理解问题上，文章和问题都可以作为 encoder 或者 decoder。这种情况下做注意力运算有两个方向，即从问题到文章，从文章到问题这两个方向。从问题到文章的注意力是指利用注意力权值对文章的语义表示加权求和，将文章的语义信息融合到问题上。从文章到问题注意力是指将问题的语义信息融合到文章中，即文章的每一个单词都会关注问题。以问题到文章的注意力计算过程为例，$P = [p_1, p_2, \cdots, p_n]$ 代表文章的语义信息，其中的每一个 p_i 代表文章中单词的语义向量表示，Q 代表整个问题的语义信息。问题到文章的注意力就是利用问题的语义信息和文章中每一个单词的向量表示计算它们之间的相关性，最后得到一组注意力权重 $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_n]$ 表示文章与问题之间的相关程度，利用 α 对文章加权求和就可以提取出来文章中

与问题最相关的单词，这些单词对于回答问题是至关重要的。具体计算过程如下：

$$\alpha_i = \text{softmax}(s(p_i, Q))$$
$$C = \sum_{i=1}^{n} \alpha_i p_i \tag{6}$$

其中 $s(p_i, Q)$ 就是计算问题的语义信息和文章中每一个单词的向量表示它们之间的相关性的一个函数，常用的计算方式有点积、双线性项、加法模型，分别见如下公式：

$$s(p_i, Q) = p_i^T Q \tag{7}$$
$$s(p_i, Q) = p_i^T W Q \tag{8}$$
$$s(p_i, Q) = v^T \tanh(W p_i + U Q) \tag{9}$$

上面的式子是将问题压缩成一个固定维度的向量，得到的注意力权重也是一维的，因此也成为一维注意力，与其相对应的是二维注意力，即对于问题中的每一个单词都会和文章做注意力计算，得到的注意力权重是二维向量。

如果按照注意力的计算形式上区分，又可以分为 one-hop 和 multi-hop 形式。one-hop，也叫“单跳结构”是指仅仅通过一次计算得到注意力权值然后加权求和得到注意力结果，这也是一种静态的计算形式。与之对应的是 multi-hop，也叫“多跳结构”。one-hop 形式下仅仅只做一次交互计算，而注意力机制虽然可以提取相关的重要信息，但是它仍然是基于浅层语义信息的相似度计算。在机器阅读理解任务中，对于复杂的问题通常是不能在一个句子中找出答案，需要多步推理才能寻找答案，如下图中的例子我们可以看到想要得到最终的答案需要进行三次推理过程，在每一个推理过程中都会变换注意力关注的对象。实现多步推理这种机制通常有三种方式：1）基于之前时间步所计算得到的关注问题的文章语义信息计算下一时间步的文档和问题交互，如 Impatient Reader [11]。2）利用 RNNs 这种基于上一时刻隐藏状态更新下一时刻隐藏状态的循环特性来达到多步推理，如 Match-LSTM [?]。3）引入额外的记忆单元存储语义信息，目的是希望解决 RNN 中不能够长期依赖导致信息丢失的问题，典型的如文献 [**?**] 提出的记忆网络 (memory networks)。

此外还有自注意力机制，自注意力机制是文本的向量表示和自身做交互，即文本中所有单词之间都会计算注意力，与单词的位置顺序无关，这样对于两个距离较远的单词仍然可以交互，循环神经网络这种序列式的计算机制非常适合处理文本句子这种序列式数据，但是也正因如此，网络的训练是非常耗时的同时存在梯度消失的问题使得模型不能捕获长距离单词之间的关联信息。自注意力机制（self attention）主要的思想就是撇弃这种序列式的计算方式，对于一个句子中的两个单词不考虑单词之间顺序的关系，直接计算它们之间的相关度，例如计算两个单词向量表示的内积。Self attention 可以捕获句子中长距离依赖的特征关系，而且这种计算方式是并行的，极大地加快了模型的训练速度。解决了循环神经网络固有的序列式传递信息导致后面的单词与前面的单词之间达不到有效的信息传递问题。

下表详细的列举了本文介绍的所有模型的注意力机制。

模型	注意力方向	注意力维度	推理模式
Attentive Reader [11]	问题到文章	一维注意力	单跳结构
Impatient Reader [11]	问题到文章	二维注意力	多跳结构
Standford Reader [12]	问题到文章	一维注意力	单跳结构
AS Reader [13]	问题到文章	一维注意力	单跳结构
IA Reader [14]	问题到文章	一维注意力	多跳结构
GA Reader [22]	文章到问题	一维注意力	多跳结构

3.1.2 相关模型

文献 [11] 最早利用神经网络模型并且融入注意力机制做 MRC 任务。文中提出两种不同的单向注意力机制 Attentive Reader 和 Impatient Reader，均是计算问题到文章的注意力。Attentive Reader 是将问题表示为一个固定长度的向量然后与文章中每一个单词做注意力计算，然后利用注意力权重对文章中的单词的向量表示加权求和得到一个固定长度的向量即为注意力运算后的结果，然后与问题联合预测答案，可见这种注意力的计算方式仅仅只计算一次，因此也叫 one-hop。这种方式类似于人在阅读时带着问题去文章中找答案。Impatient Reader 计算注意力的机制类似于文献 [7] 的方式，也属于一种 multi-hop 的计算。并不是将问题表示为一个固定长度的向量，

而是对于问题中的每一个单词都要和整个文章做注意力计算，而且计算的结果要和下一个单词以及文章共同做注意力计算，最后一个单词的注意力结果作为整个 Impatient Reader 计算注意力过程的结果。这种方式类似于人在阅读过程中不断的在问题和文章之间做关注。文献 [12] 在 Attentive Reader 的基础上利用双线性项（公式 8）取代原有的利用 tanh 函数的加法模型（公式 9）并且直接将对文章加权求和后得到的向量作为预测答案的输入而不是联合问题 Q 的语义信息，实验证明这种简化反而提高了模型的准确度。

文献 [8] 提出的记忆网络并不能端到端的训练。为了处理这个问题，文献 [10] 提出一种端到端的记忆网络（MemN2N）。利用记忆槽存储文档中每一个句子的嵌入矩阵，记忆槽的状态以及问题的语义信息会随着与文档的多次交互不断更新。

文献 [14] 提出 Iterative Attention Reader（IA Reader）模型，利用 BiGRU [?] 存储每一次迭代计算得到的问题和文章的交互信息。在每一时间步上，首先利用上一次的 BiGRU 的状态与问题做一维注意力匹配提取出问题的语义信息，然后再结合上一次的 BiGRU 的状态与文章再做一维注意力匹配从而提取出文章的语义信息。将问题与文章的语义信息通过各自的门控单元作为 BiGRU 当前时刻的输入，其中门控单元采用前馈神经网络用来解决当前时间步下问题和文章的语义信息提取不充分的问题。

文献 [22] 提出 Gated Attention Reader（GA Reader）模型，类似于 IA Reader [14] 模型同样采用 BiGRU 作为编码模块实现多跳结构。在每一步的推理过程中，首先通过 BiGRU 得到问题的语义信息然后对文章的每一个单词做注意力的计算，同时采用点乘计算的门控机制建模当前时间步的推理过程下关注了文章第 i 个单词的问题语义信息与文章第 i 个单词的语义信息影响，从而更新文章的语义表示。这种处理过程类比于带着问题反复的阅读文章，每一次都加深对文章的语义理解。

文献 [21] 提出一种动态决定推理次数的模型 ReasonNet，不像之前的模型如 GA Reader [22]，IA Reader [14] 等在整个推理过程中有着固定的推理次数。这种固定推理次数的缺点就是不考虑问题的复杂性，对于复杂的问题往往需要模型多次的推理，因此不同题目难度需要不同的推理次数，应当让模型学会什么时候终止推理。为了达到这一目的，ReasoNet 模型利用一个终止门产生二元值输出来动态的决定是否继续推理。Reasonet 模型大致分为外部记忆单元模块、内部控制器模块、终止门模块以及答案输出模块。具体的，将文章和问题通过 Bi-GRU 编码后的语义表示作为外部的记忆单元 M，利用内部控制器（采用 GRU）当前时刻的状态 s_t 与 M 做二维注意力匹配，得到注意力结果输入到内部控制器中更新内部控制器的状态。终止门模块以当前时刻内部注意力的状态作为输入来判断是否需要继续推理。由于产生了二元离散输出值，使得模型用梯度下降法训练，因此模型引入强化学习机制训练。

文献 [23] 提出一种 Match-LSTM 的交互机制，利用 RNNs 的循环机制达到多跳结构。与之前模型如 Impatinent Reader [11] 不同，虽然 Match-LSTM 也是多跳结构，并且也是一维注意力，但是 Match-LSTM 计算的方向是文章到问题的注意力，也就是对问题的文本表示加权求和，将问题的语义信息融入到 Match-LSTM 中。具体的就是对于文章中当前时刻单词的上下文表示，以及 Match-LSTM 前一时刻的隐藏状态与问题的语义编码做注意力的计算，计算方式和 Bahdanau [7] 一样采用激活函数为 tanh 的全连接层以及输出维度是 1 的全连接层，最后利用 softmax 将结果以概率形式归一化得到注意力权重。具体计算过程如下：

$$
\begin{aligned}
s_t &= v^T \tanh(W^Q H^Q + W^P h_t^P + W_r h_{t-1}^r) \\
\alpha_t &= \mathrm{softmax}(s_t)
\end{aligned} \tag{10}
$$

其中 H^Q 是问题通过编码层的输出，h_t^P 是文章的第 t 个单词通过编码层的输出，h_{t-1}^r 是 Match-LSTM 上一时刻的隐藏状态。α_t 是文章的第 t 个单词与问题的每一个单词之间的注意力权重。计算得到的注意力权重对问题的语义编码加权求和，此时得到的是关注问题的向量表示，然后与文章当前时刻单词的上下文表示拼接作为 Match-LSTM 当前时刻的输入。

$$
\begin{aligned}
z_t &= [h_t^p; \alpha_t H^q] \\
h_t^r &= \mathrm{LSTM}(z_t, h_{t-1}^r)
\end{aligned} \tag{11}
$$

此外为了使得文章从后向前的对问题做关注，将文章序列翻转再次按照上述方式计算，最后将两个方向的

计算结果拼接作为交互层的输出。这种 multi-hop 的计算方式比较耗时。

文献 [18] 提出一种 Dynamic Co-attention Network（DCN）模型，在交互层中采用协同注意力 [38] 机制，这种机制首先在视觉问答领域提出，目的是将图片到问题，以及问题到图片两个方向的注意力以某种形式融合。定义 $D = [d_1, d_2, \cdots, d_m, d_\phi] \in R^{l\times(m+1)}$，$Q^{'} = [q_1, q_2, \cdots, q_n, q_\phi] \in R^{l\times(n+1)}$。，其中 d_ϕ 和 p_ϕ 是一个监哨向量 [16]，在做注意力交互机制计算时，对于那些某些在文章和问题中不相关的单词，模型将这些单词映射为这个监哨向量，使模型并不会关注到这些不相关的单词。为了使得使得问题编码空间和段落文本编码空间有可变化的余地，在问题编码向量 $Q^{'}$ 上再利用一个非线性层把 $Q^{'}$ 转换为 $Q = \tanh(W^{(Q)}Q^{'} + b^{(Q)}) \in R^{l\times(n+1)}$，$Q$ 作为问题的最终表示。协同注意力网络同步的计算文章对问题的注意力以及问题对文章的注意力。具体的，首先计算关联矩阵 L 然后利用关联矩阵得到文章端的注意力权重 A^Q 以及问题端的注意力权重 A^D:

$$
\begin{aligned}
L &= D^T Q \in R^{(m+1)\times(n+1)} \\
A^Q &= \text{softmax}(L) \in R^{(m+1)\times(n+1)} \\
A^D &= \text{softmax}(L^T) \in R^{(n+1)\times(m+1)}
\end{aligned}
$$

A^Q 的每一列表示的是问题的一个单词对文章所有单词的相关度，它的编码空间是在文章端，同理 A^D 的编码空间是在问题端。因此接下来利用 A^Q 计算问题对文章的注意力 $C^Q = DA^Q \in R^{l\times(n+1)}$。对于文章对问题的注意力，定义

$$
C^D = [Q; C^Q]A^D \in R^{2l\times(m+1)}
$$

其中 $C^Q A^D$ 的含义是将问题端的编码信息转换到文章端，其中 [;] 表示向量在行的维度上连接，C^D 是问题和文章的协同依赖的注意力矩阵，它的每一个值都是既关注到了文章的语义信息，也关注到了问题的语义信息。最后融合 C^D 和文章的向量表示 D 作为交互层的输出。可以看出 DCN 模型中注意力仅仅只计算一次，而且两个方向的注意力可以并行计算，这属于 one-hot 形式的注意力机制，速度要显著快于 Match-LSTM。

文献 [19] 提出 Bidirectionl Attention Flow（BiDAF）模型。对于之前的模型如 Attentive Reader [11]，Impatient Reader [11]，Match-LSTM [23] 等，这些模型计算注意力的方式都是将注意力结果问文章的语义信息融合作为交互层的输出，而 BiDAF 额外将注意力结果也作为交互层输出的一部分，这在一定程度上避免了过早的对问题语义信息概括而导致信息的损失，使得交互层计算得到的注意力结果仍然参与后面网络层的计算。BiDAF 的计算方式如下:

$$
\begin{aligned}
S &= W^T[C; Q; C \circ Q] \in R^{m\times n} \\
\widehat{Q} &= \text{softmax}(S)Q \ \in R^{m\times d} \\
\alpha &= \text{softmax}(\max_{col}(S)) \in R^m \\
\widehat{c} &= \sum_{i=1}^{m} \alpha_t C_{:i} \in R^d
\end{aligned} \tag{12}
$$

其中 C, Q 分别表示文章和问题的语义向量表示，m, n 分别表示文章和问题的长度，S 表示文章和问题之间的相似度矩阵，$\circ$ 表示点乘运算。$\widehat{Q}$ 表示的就是文章对问题的注意力，即代表着问题中哪个单词与文章最相关。α 的意义是将关于问题最相关的文章中的单词提取出来，这些单词是作为答案的关键因素。$\widehat{c}$ 就是将文章中与问题最相关的单词的语义向量加权求和，将其重复 m 次得到问题对文章的注意力 $\widehat{C}$。最后以如下形式作为 BiDAF 交互层的输出:

$$
[C; \widehat{Q}; C \circ \widehat{Q}; C \circ \widehat{C}] \tag{13}
$$

此外实验结果表明去掉文章到问题的注意力后模型效果显著下降。

文献 [20] 提出一种带有门控机制的注意力循环神经网络以及自注意力机制联合的交互层设计模型，也叫 RNet。RNet 在交互层的设计分为两部分。第一部分是带有门控机制的注意力循环神经网络，整体计算方式类似于 Match-LSTM [23]，不过采用的是 RNN 不是 LSTM 而且额外加入了门控机制使得模型可以有选择的输出语义信息。具体的，在公式（2）中的 z_t 上添加一个门控单元:

$$
\begin{aligned}
g_t &= \text{sigmoid}(W_g z_t) \\
z_t^* &= g_t \odot z_t
\end{aligned} \tag{14}
$$

其中 $\odot$ 表示元素之间的点乘。由于 $z_t = [h_t^p; \alpha_t H^q]$，$h_t^p$ 表示的是文章的第 t 个单词的语义表示，$\alpha_t H^q$ 表

示的是对问题语义表示的融合，因此通过添加门控单元使得模型可以有选择的决定哪部分作为重要的语义信息输出。这种机制类似于人在阅读过程中要忽略文章中那些与问题无关的信息，凸显出重要的信息才能更加准确的找到答案。第二部分是利用自注意机制对文章的语义信息再次交互建模。基于注意力机制的循环神经网络的输出对关注了问题的文章语义表示与原始文章语义表示建模后的输出，而这种计算机制的问题之一是在它输出的整个句子的语义信息中前面的单词没有关注到后面的单词而两个距离较远的单词之间交互信息由于链式求导等原因会变得很弱。因此通过自注意机制可以使得文章中每一个单词关注到其余所有的单词，使得模型对文章达到更深层次的理解。

文献 [28] 提出一种网络模型 QANet，不像之前的那些模型几乎都是用 RNN 的变体来做编码器，QANet 提出一种新颖的编码结构，利用卷积结合 transformer [32] 中的 multi-head 注意力结构，实验结果表明这种架构不仅加快训练速度同时在 SQuAD 数据集上模型性能优于那些利用 RNN 作为编码器的模型。QANet 交互层的计算方式类似于 BiDAF [19] 以及 DCN [18]。具体的：

$$
\begin{aligned}
S &= W[Q, C, Q \odot C] \in R^{m \times n} \\
\bar{S} &= \text{softmax(S)} \in R^{m \times n} \\
\widehat{S} &= \text{softmax}(S^T) \in R^{n \times m} \\
A &= \bar{S} Q^T \in R^{m \times d} \\
B &= \bar{S} \widehat{S} C^T
\end{aligned} \tag{15}
$$

其中 $Q \in R^{d \times n}$ $C \in R^{d \times m}$ 分别问题和文章通过编码层的语义表示，m n d 分别表示文章长度和问题长度以及输出维度。$\odot$ 表示点乘，W 是一个训练参数，A 表示就是文章到问题的注意力，B 表示的是问题到文章的协同注意力。其中通过 $\bar{S}$ 将问题的语义编码空间转换到文章中，类似于 DCN [18] 的计算方式。交互层的输出采用与 BiDAF [19] 一样的输出形式，见公式（4）。

3.2 基于预训练模型的模型

预训练模型近年来 NLP 领域获得了极大的关注度，基于预训练模型的方法在 NLP 几乎所有的任务上都要优于之前的模型。预训练方式源自于迁移学习的概念：首先在其它相关任务上预训练模型，使得模型已经学习到一些知识，然后在目标任务上做进一步优化，实现模型所学知识的迁移。对于 NLP 领域来讲，预训练过程就是在大量的文本数据上学习到通用的语言表示。在应用到下游任务时，预训练所学习到的知识提供了一个很好的初始化点，从而加快模型的收敛并且提高模型的性能。此外预训练也对模型起到正则化的作用，使得模型避免在数据不充分的数据集上过拟合。目前最流行的几个预训练模型如 ELMo [34]，GPT [35]，BERT [36] 以及基于 BERT 改进的预训练模型 UNILM [39]，[37] 等等全都是基于语言模型做预训练，因此也叫预训练语言模型。预训练的模型根据迁移方式的不同可以分为两种方式基于特征的方式（Feature-based）和基于微调的方式（Fine-tuning）。基于特征的方式是指对于预训练好的模型，应用到具体任务时固定模型的参数，将具体任务的数据通过预训练的模型得到数据的表示特征然后输入到根据具体任务设计的具体模型中。基于微调的方式是广泛采用的一种预训练模型的使用方式如计算机视觉领域的经典模型 VGG [?],ResNet [?]，具体的做法是在预训练模型的基础上加上少量的网络层，然后利用具体任务的标注数据训练整个网络模型。本节将主要概述 NLP 领域一些流行的预训练语言模型以及它们在 MRC 任务上的应用和效果对比。

3.2.1 预训练模型

ELMo [34] 是较早提出的一种预训练语言模型。传统的词嵌入模型如 word2vec [5],GloVe [6] 属于静态的词向量，训练好模型后一个单词的表示向量就是固定的，没有考虑上下文的信息，因此无法解决多义词问题。ELMo 提出一个三层网络的模型，第一层就是词嵌入层用来提取单词特征，随后是两层 BiLSTM 网络分别提取单词的词性特征和语义特征。前向 LSTM 的目标是根据前 $k-1$ 个词预测第 k 个词，从而计算出一个句子的概率，如公式（1）。反向 LSTM 的目标是根据最后的单词直到第 $k+1$ 个

单词预测第 k 个单词，具体如公式（2）。

$$p(t_1, t_2, \cdots, t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, \cdots, t_{k-1}) \quad (16)$$

$$p(t_1, t_2, \cdots, t_N) = \prod_{k=1}^{N} p(t_k | t_{k+1}, t_{k+2}, \cdots, t_N) \quad (17)$$

最后的目标函数是最大化联合的前向和后向最大似然：

$$\begin{aligned} L(\Theta) = &\sum_{k=1}^{N} (\log p(t_1, \cdots, t_{k-1}; \Theta)) \\ &+ \log p(t_{k+1}, \cdots, t_N; \Theta) \end{aligned} \quad (18)$$

在做阅读理解任务时，将文章和问题输入到模型中，每一层都会得到句子的语义表示，然后将每一层的特征加权求和作为 ELMo 的输出，此时得到的每一个单词的向量表示都是考虑了上下文的，将其作为下游模型嵌入层的输入。利用 ELMo+BiDAF [19] 结构超过之前单模型 8.3 个百分点。可以看出 ELMo 属于自回归语言模型，模型的迁移方式是基于特征的方式。

GPT [35] 首先提出两阶段的训练方式，先利用大规模无监督语料库训练语言模型，然后在下游任务的少量监督数据集上微调模型，因此 GPT 也是一种半监督学习方式。由于 Transformer [32] 在处理长期依赖性方面比 LSTM 的性能更强同时并行计算的快于 LSTM，因此 GPT 包括后来的大规模预训练模型都采用 Transformer 作为特征提取器。由于采用单向语言模型作为训练的任务，因此没有用完整的 Transformer 结构而且利用掩码机制约束模型。在预训练阶段的目标函数就是标准的单向语言模型的目标函数：

$$L(\Theta) = \sum_{k=1}^{N} (\log p(t_1, \cdots, t_{k-1}; \Theta)) \quad (19)$$

GPT 与 ELMo 的共同点是它们都属于自回归语言模型，而最大的不同之处在于迁移的设计上并不是像 ELMo 那种基于特征方式的迁移。GPT 有统一的输入数据表示形式，而且在输入层并不需要继续设计复杂的网络模型而仅仅只需要简单的结构，在训练时整个网络模型的参数共同训练。这种基于微调的迁移方式被后续的其它预训练模型广泛采用。利用 GPT 在 RACE [25] 数据集上微调后达到的效果比之前最好的模型提高了 5.7 个百分点。GPT 是 NLP 领域首先提出的一种基于 fine-tune 的通用式网络结构，不仅仅在 MRC 领域，在很多其它的 NLP 领域都取得了很大的进步。

BERT [36] 与 ELMo 和 GPT 最本质的不同在于预训练方式上采用的自编码语言模型。自回归语言模型的缺点就是由于自回归的性质使得它不能同时利用一个单词的上下文信息预测这个单词，ELMo 虽然利用双向 LSTM 来预测单词但是这也只是两个单向的语言模型的拼接并不能当做双向语言模型。BERT 在整个预训练的流程上采用 GPT 的方式，即预训练然后微调。但是 BERT 采用降噪自编码（DAE）的方式训练，具体的就是在输入数据中加入噪声，也就是随机掩盖掉一些单词，让模型根据掩盖掉单词的上下文预测这个单词，这种训练方式也叫掩码语言模型（MLM）。对比 ELMo 和 GPT 的目标函数，BERT 的掩码语言模型的目标函数为：

$$L(\Theta) = \sum_{i=1}^{N} \log P(t_k | t_1, t_2, \cdots, t_{k-1}, t_{k+1}, \cdots, t_N) \quad (20)$$

此外利用下一个句子预测（NSP）任务使得模型在诸如文本蕴含、问答这类需要判断两个句子关系的下游任务表现更好。BERT 的预训练过程实质上是一个多任务学习的过程，通过同时训练掩码语言模型和下一个句子预测两个提高了预训练模型的语义表达能力。BERT 在 SQuADv1.1 [15] 数据集上的效果超过了人类的水平，在其它的 NLP 任务上也都有提升。

UNILM [39] 扩展了 BERT 预训练的任务，由于双向语言模型的性质使得 BERT 在生成任务上效果不好，因此 UNILM 利用掩码机制同时训练单向语言模型（包括从左到右和从右到左）、双向语言模型以及 Seq2Seq 语言模型，使用掩码机制来解决不同的语言模型约束问题。虽然是三个不同的语言模型作为训练任务但是共享同一个网络结构，也就是利用三个任务来联合的优化模型的参数。这种多任务学习的方式缓解了模型在某一个单一任务上容易出现过拟合的问题，使得预训练后的模型不仅在原有的自然语言理解任务上效果进一步提升，同时在自然语言生成任务上也达到了很好的效果。在微调阶段，对于自然语言理解任务（如文本分类、抽取式问答等）同 BERT 的微调方式一样，对于自然语言生成任务（如自动化摘要、生成式问答等）采用与预训练阶段 Seq2Seq 语

表 1: 预训练语言模型对比

模型	任务	模型结构	介绍
ELMo	单向 LM	LSTM	拼接两个单向语言模型的语义信息， 基于特征形式迁移
GPT	单向 LM	Transformer	首次采用预训练＋微调形式
BERT	双向 LM + NSP	Transformer	利用掩码语言模型和预测下一个句子共同作为训练任务
UNILM	单向 LM+ 双向 LM+Seq2SeqLM	Transformer	同时训练三种语言模型， 采用掩码机制解决不同语言模型的约束问题
ALBERT	双向 LM+SOP	Transformer	对比 BERT 采用矩阵分解和共享参数减少模型的参数量， 同时用句子顺序预测取代下一个句子预测

言模型类似的方式在目标序列中随机的掩盖掉一些单词从而让模型根据源序列生成目标序列的单词达到生成任务的目的。UNILM 不仅在抽取式 QA 数据集（如 SQuAD）上超过 BERT，在生成式 QA 数据集（如 CoQA）上的表现远超过最初的基准模型。

ALBERT [37] 通过在 BERT 的基础上进行了如下几点改进:

1. BERT 的词嵌入矩阵的大小是 $V \times H$，其中 V 代表词汇表的词汇数量，主要的缺陷是由于输入的词是以独热形式编码的，因此词嵌入矩阵会有大量的参数在模型训练时不会被更新，此外词嵌入矩阵中的参数仅仅表达了单词自身词向量的特征，没有上下文语义，因此通过矩阵分解的方式将 $V \times H$ 分解为 $V \times E + E \times H$，其中 $E << H$，这样便减少了模型在词嵌入层的参数。

2. 虽然 BERT 利用 12 层的 transformer 结构，但是由实验观察得到，这 12 层的所学习到的参数内容很相似，因此 ALBERT 采用参数共享方式，使得 12 层的 transformer 共享同一组参数，这极大地减少了模型参数的数量。

3. BERT 的 NSP 任务包含了两个子任务，来源于同一篇文章的两个连续的句子判别为正例，来源于不同文章的句子判别为负例，这使得模型不能集中于判断两个句子之间的顺序反而更加关注句子所表达的主题是否一致。因此 ALBERT 中用 SOP(sentence-order prediction) 取代 NSP 任务，SOP 是指句子顺序预测，两个句子都是来源于同一篇文章的连续的句子，调换顺序后便是负例，这使得模型集中于预测句子之间的顺序，实验表明 SOP 任务使得模型的效果提升一个百分点。

ALBERT 的改进机制使得模型的预训练后的效果更好，在 RACE [25]，SQuAD 2.0 [26] 等机器阅读理解数据集上的准确率超过 BERT 以及其它的预训练模型。

3.2.2 基于预训练模型的相关模型

3.3 答案预测层的设计

如前所述，答案预测层的设计要依据答案的形式而设计，下面介绍各个模型在四类不同的 MRC 任务上的输出层设计。

1）填空型：这类任务答案的形式是预测问题中缺失的单词，而且缺失的答案来源于文章中。文献 [11] 最早提出将问题的语义向量与关注了问题的文章的语义向量拼接成一个向量然后映射到整个词典中预测那个缺失的单词。这种方法存在的一个问题就是不能够确保预测的单词一定是文章中的词汇，因为它是将最后的输出向量映射到整个词典中，这就使得模型的预测准确率受到影响。指针网络（Pointer networks [24]）模型由 seq2seq 模型演变而来，主要就是为了解决输出源自于输入的问题，实现方式是利用计算的注意力的权重分布直接输出预测结果，而这种机制正适合填空型任务以及片段选择型任务。文

献 [?] 提出 AS Reader 模型正是受到指针网络的启发，对于计算得到的注意力权重分布，将其中相同单词的注意力权值相加，最后输出具有最大权值的单词最为答案。填空型任务模型的损失函数可以写为 $L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\log P_{y_i}$。其中 θ 为模型参数，N 代表样本数目，y_i 表示文章中第 i 个样本的文章中标准答案的位置。

2）多项选择型：这类任务是从多个候选答案选项中选择正确的选项。处理这种任务最简单的一种方式就是计算模型输出后的文章的语义信息和选项之间的相似程度，相似程度最高的作为预测的选项，从而将问题变化为句子之间的语义匹配问题。文献 [40] 提出将问题、文章、选项一起方法模型中做交互计算输出一个向量作为输出层的输入，输出层采用简单的输出维度是 1 的全连接层，输出的值代表模型对这个选型的打分值，其它的选项类似的处理，值最高的选项作为预测的答案。最后对所有选项的打分值做归一化作为模型的损失函数。多项选择型任务模型的损失函数可以写为 $L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{m}\log P_{y_j^i}$。其中 θ 为模型参数，N 代表样本数目，m 代表选项个数，y_j^i 表示第 i 个样本中第 j 个选项是正确答案。

3）片段选择型：这类任务是从文章中提取出来一段连续的单词作为答案，虽然类似于填空型任务输出来源自输入的性质，但是不像填空型任务仅仅只是预测一个单词。因此填空型任务答案输出层的设计不能直接用来作为片段选择型任务答案预测层。由于提取文本的长度不固定，使得这一任务更具有挑战性。文献 [23] 受到指针网络的启发提出了两种基于指针网络的输出模型，第一种是序列式模型，利用指针网络以一种序列式的形式生成答案的每一个位置，处理过程类似于 seq2seq 模型的解码过程，这种模型下答案的每一个单词可能出现在文本段落的任何一个位置，这是因为指针网络并没有要求从输入中选择的输出具有连续性。由于答案的长度不固定，因此在段落中设置一个特殊的位置表示答案的终止点，当预测到这个位置时终止答案的生成。第二种是边界式模型，不同于序列式模型那样序列的生成答案的每一个位置，由于要预测的答案是一段连续的单词的组合，因此可以利用指针网络仅仅预测答案的起始位置和终止位置。所预测答案的概率是预测这两个位置概率的乘积，这种方式相比于第一种更加的简单而且测试结果表明更加高效。

边界式模型的这种设计思想也被后来很多 MRC 模型采纳。如 RNet [20] 采用几乎一样的边界式模型，只不过解码端的初始状态采用的是对问题的语义向量加权求和。尽管边界式模型简单有效，但是在文章中可能有些文本片段与标准答案相似，比如初始位置一样，那么边界式模型有可能陷入极值的情况从而提取错误的文本片段。为了处理这个问题，DCN [18] 模型提出一种动态迭代的指针网络作为解码端，利用上一次预测的答案的起始位置和终止位置以及解码端当前的状态来重新评估下一次预测答案的起始位置和终止位置。多次迭代后选取所有迭代次数中概率最大的情形作为预测答案。片段选择任务模型的损失函数可以写为 $L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\log P_{y_i^s}^S + \log P_{y_i^e}^E$。其中 θ 为模型参数，N 代表样本数目，y_i^s 表示第 i 个样本中标准答案的起始位置在文章中的位置，y_i^e 表示第 i 个样本中标准答案的终止位置在文章中的位置。如果考虑到不可回答的问题，如数据集 SQuAD 2.0 [26]，还需要额外在输出层加上一个输出维度是 1 的全连接层。此时的损失函数可以写为 $L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\log P_{y_i^s}^S + \log P_{y_i^e}^E + \log P_{y_i^u}^U$。其中 y_i^u 表示第 i 个样本中的问题是不可回答的问题

4）自由答案型：这类任务的答案形式已经不再是原文中某段文本，而是需要根据文章和问题生成符号语法规范的答案语句。这类任务对答案生成模块的能力要求较高。文献 [29] 采用三种模型在在 CoQA 数据集上进行实验。第一种是传统的 seq2seq 模型，第二种是 Pointer-Generator Network [41]，简称 PGNet。这个模型最早提出来用在文本摘要领域，模型结合了 seq2seq 的生成机制以及指针网络的拷贝机制，使得模型既能生成单词又能在原文中拷贝单词，实验结果表明该模型的效果优于传统的 seq2seq 模型。第三种是 DrQA+PGNet 模型，其中 DrQA [43] 是类似于上述 BiDAF [19]，RNet [20] 等模型的片段抽取模块。整个模型的思想是先利用片段提取模块从文章中提取中与问题最相关的一段文本，然后利用答案生成模块在这个被抽取出来的文本上生成答案。

这种（提取模块 + 生成模块）模型在其它的生成模 模型的效果是最好的。
型中如 SNet [42] 等广泛使用。实验结果也表明这种

参考文献

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8): 1735–1780, 1997.

[2] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for au- tomatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pages 311–318. Association for Computational Linguistics, 2002.

[3] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. Text Summariza- tion Branches Out, 2004.

[4] Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. arXiv preprint arXiv:1302.4389 (2013)

[5] Mikolov T，Sutskever I，Chen K，et al.Distributed represen- tations of words and phrases and their compositionality[C]// Advances in Neural Information Processing Systems，2013：3111-3119.

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word//w representation. In Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, 2014.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learn- ing to align and translate. ICLR, 2015.

[8] J. Weston, S. Chopra, and A. Bordes. Memory networks. In International Conference on Learning Representations (ICLR), 2015.

[9] Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. arXiv preprint arXiv:1505.00387 (2015)

[10] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In Advances in neural information processing systems, pages 2440–2448, 2015.

[11] Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blun- som, P.: Teaching machines to read and comprehend. In: Advances in Neural Information Processing Systems, pp. 1693–1701 (2015)

[12] Danqi Chen, Jason Bolton, and Christopher D Manning. A thorough examination of the cnn/daily mail reading comprehension task. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 2358–2367, 2016.

[13] Rudolf Kadlec, Martin Schmid, Ondřej Bajgar, and Jan Kleindienst. Text understanding with the attention sum reader network. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 908–918, 2016.

[14] Alessandro Sordoni, Philip Bachman, Adam Trischler, and Yoshua Bengio. Iterative alternat- ing neural attention for machine reading. arXiv preprint arXiv:1606.02245, 2016.

[15] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2016.

[16] Merity, S., Xiong, C., Bradbury, J., Socher, R.: Pointer sentinel mixture models. arXiv preprint arXiv:1609.07843 (2016)

[17] Wang, S., Jiang, J.: Machine comprehension using match-lstm and answer pointer. arXiv preprint arXiv:1608.07905 (2016)

[18] Xiong, C., Zhong, V., Socher, R.: Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604 (2016)

[19] Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 (2016)

[20] Wang, W., Yang, N., Wei, F., Chang, B., Zhou, M.: Gated self-matching networks for reading comprehension and question answering. In: Proceedings of the 55th Annual Meet- ing of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 189–198 (2017)

[21] Shen, Y., Huang, P.S., Gao, J., Chen, W.: Reasonet: Learning to stop reading in machine comprehension. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1047–1055. ACM (2017)

[22] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1832–1846, 2017.

[23] Shuohang Wang and Jing Jiang. Learning natural language inference with LSTM. In Proceedings of the Conference on the North American Chapter of the Association for Computational Linguistics, 2016.

[24] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In Proceedings of the Conference on Advances in Neural Information Processing Systems, 2015.

[25] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 785–794, 2017.

[26] Rajpurkar, P., Jia, R., Liang, P.: Know what you don' t know: Unanswerable questions for squad. arXiv preprint arXiv:1806.03822 (2018)

[27] Hill, F., Bordes, A., Chopra, S., Weston, J.: The goldilocks principle: Reading children' s books with explicit memory representations. arXiv preprint arXiv:1511.02301 (2015)

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

[29] Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. arXiv preprint arXiv:1808.07042, 2018.

[30] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)

[31] Kočiskỳ, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K.M., Melis, G., Grefenstette, E.: The narrativeqa reading comprehension challenge. Transactions of the Association of Computational Linguistics 6, 317–328 (2018)

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Neural Information Processing Systems, 2017b.

[33] François Chollet. Xception: Deep learning with depthwise separable convolutions. abs/1610.02357, 2016.

[34] Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)

[35] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understand- ing with unsupervised learning. Tech. rep., Technical report, OpenAI (2018)

[36] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)

[37] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In ICLR.

[38] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. arXiv preprint arXiv:1606.00061, 2016.

[39] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language un- derstanding and generation. In NeurIPS, pages 13042–13054, 2019.

[40] Shuohang Wang, Mo Yu, Jing Jiang, and Shiyu Chang. 2018. A Co-Matching Model for Multi-choice Reading Comprehension. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 746–751. Association for Computational Linguistics.

[41] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summa- rization with pointer-generator networks. In Annual Meeting of the Association for Compu- tational Linguistics (ACL), pages 1073–1083. Vancouver, Canada.

[42] Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. S-net: From answer extraction to answer synthesis for machine reading comprehension. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[43] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In Asso- ciation for Computational Linguistics (ACL), pages 1870–1879. Vancouver, Canada.