多跳阅读理解

Multi-hop Reading Comprehension

汇报人：王佳安

2020.12.08

目录

Part 1 概述

1.1 什么是多跳阅读理解？

多跳问题：需要在多个事实片段之间进行推理才能得到正确答案的问题。

例如：《爱情公寓》的哪位主演做客了LOL S10的评论席？

P1：《爱情公寓》是上海电影集团公司出品，上海高格影视制作有限公司承制，由汪远、邹杰编剧 、韦正执导，**娄艺潇、陈赫、孙艺洲、李金铭、王传君、邓家佳、金世佳、赵霁、李佳航、成果、万籽麟、张一铎、赵文琪领衔主演的都市青春情景喜剧**。讲述了住在爱情公寓里的一群租户之间发生的乐趣十足的日常故事 。

P2：2020全球总决赛半决赛将于10月24日、25日举行，**“十年老粉”明星嘉宾赵品霖和李佳航将现身评论席**,让我们拭目以待! 距离上次“张伟”现身解说席已经是2017年的事情。

Supporting Sentence/Fact

Answer：李佳航

1.2 多跳问答都有哪些形式？

根据推理所需的数据类型将多跳问答分为三类

本次汇报关注的范围

推理数据所需类型	数据集
多跳阅读理解：仅从纯文本数据上进行推理	HotpotQA, Wikihop, ROPES, DROP, Dream, MultiRC, QASC等
多跳知识图谱问答：仅从知识图谱上进行推理	MetaQA等
两者结合：从文本与KG上进行推理	OpenBookQA等

note：分类方式依据的是推理所需的数据类型，而不是问题类型，例如Wikihop中的问题是三元组形式，但该数据集还是在文本数据上进行推理，所以仍属于多跳阅读理解。

1.3 MhRC数据集概述

数据集	规模	开放域	答案类型	备注	发表会议/期刊	引用量
段落级的推理：大部分研究都在HotpotQA与Wikihop上						
HotpotQA	113K	y&n	抽取式	在Open设定中：只给问题和一个5M的语料库。 在non-Open设定中：每个问题可由2个段落推理得到答案，于此同时还会有8个干扰段落。	EMNLP 2018	316
Qangaroo	51K &3K	n	多选式	每个QA对有多篇对应的段落（平均14，最多64）	TACL 2018	209
数学的推理						
DROP	97K	n	演算式	答案不一定出现在原文中 需要通过计算、计数等操作得到	NAACL 2019	141
MultiRC	6K	n	多选式	多个sentences以及答案候选集，根据sentences去选择正确的答案	NAACL 2018	114
QASC	10K	y	多选式	含有9980个八项选择题，每个问题都被标注了两个fact sentences用来推理出最终的答案。还提供了一个包含了17M个句子的语料库，所有的fact sentences都在里面	AAAI 2020	29
句子级的推理						
ROPES	14K	n	抽取式	每个QA对就给两个对应的文档，从中推理出答案，没有干扰文档。	MRQA@EMNLP 2019	24

没啥人用的推理：难度与数据量均低于HotpotQA

1.3.1 段落级的推理

Paragraph A: LostAlone were a British rock band ... consisted of *Steven Battelle, Alan Williamson, and Mark Gibson*...
Paragraph B: Guster is an American alternative rock band ... Founding members *Adam Gardner, Ryan Miller, and Brian Rosenworcel* began...
Q: Did LostAlone and Guster have the same number of members? (**yes**)

HotpotQA数据集

Wikihop数据集

The Hanging Gardens, in **[Mumbai]**, also known as Pherozeshah Mehta Gardens, are terraced gardens … They provide sunset views over the **[Arabian Sea]** …

Mumbai (also known as Bombay, the official name until 1995) is the capital city of the Indian state of Maharashtra. It is the most populous city in **India** …

The **Arabian Sea** is a region of the northern Indian Ocean bounded on the north by **Pakistan** and **Iran**, on the west by northeastern **Somalia** and the Arabian Peninsula, and on the east by **India** …

Q: (Hanging gardens of Mumbai, country, ?)
Options: {Iran, **India**, Pakistan, Somalia, …}

1.3.2 句子级的推理

S3: Hearing noises in the garage, Mary Murdock finds a bleeding man, mangled and impaled on her jeep's bumper.
S5: Panicked, she hits him with a golf club.
S10: Later the news reveals the missing man is kindergarten teacher, Timothy Emser.
S12: It transpires that Rick, her boyfriend, gets involved in the cover up and goes to retrieve incriminatory evidence off the corpse, but is killed, replaced in Emser's grave.
S13: It becomes clear Emser survived.
S15: He stalks Mary many ways.

Who is stalking Mary?

A)* Timothy	D) Rick
B) Timothy's girlfriend	E) Murdock
C)* The man she hit	F) Her Boyfriend

MultiRC数据集

1.3.3 数学的推理

Question	Passage	Answer
What is the second longest field goal made?	... The Seahawks immediately trailed on a scoring rally by the Raiders with kicker *Sebastian Janikowski nailing a 31-yard field goal* ... Then in the third quarter *Janikowski made a 36-yard field goal*. Then *he made a 22-yard field goal* in the fourth quarter to put the Raiders up 16-0 ... The Seahawks would make their only score of the game with kicker *Olindo Mare hitting a 47-yard field goal*. However, they continued to trail as *Janikowski made a 49-yard field goal*, followed by RB Michael Bush making a 4-yard TD run.	47-yard

DROP数据集

1.4 评价指标

对于span预测式的数据集（HotpotQA、ROPES）

- EM：模型只有输出与标准答案一致时得1分，其余均得0分。
- F1：基于模型的输出以及标准答案的重复度来计算F1值。

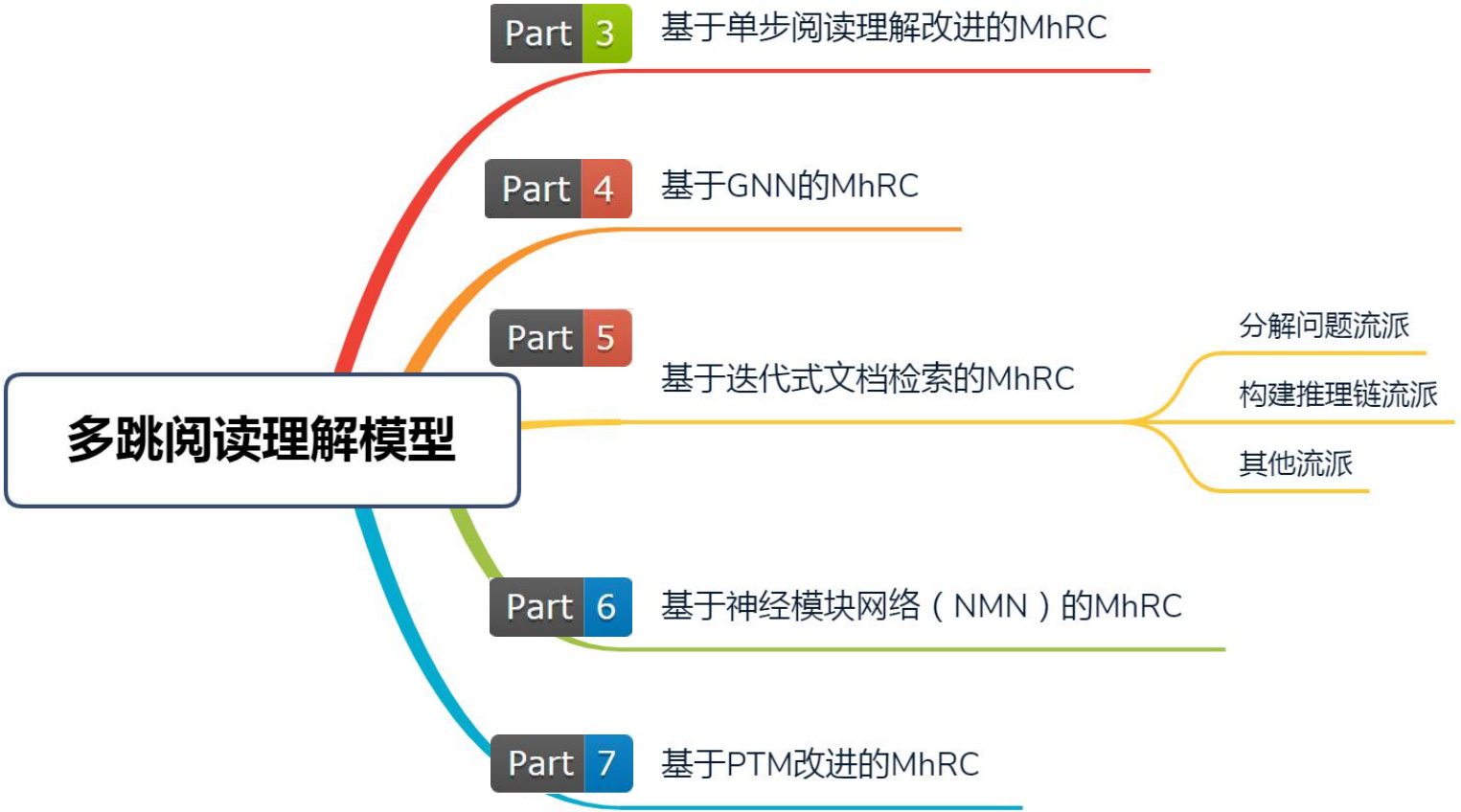
$$F_1 = \frac{2 \times P \times R}{P + R}, P = \frac{|\text{重复单词}|}{|\text{模型答案单词}|}, R = \frac{|\text{重复单词}|}{|\text{标准答案单词}|}$$

对于多选式的数据集（Wikihop、QASC、MultiRC）

- Accuracy

Part 2 MhRC方法总览

MhRC方法总览：百家争鸣



MhRC方法总览：百家争鸣

Leaderboard of HotpotQA

1. ANS (distractor setting)

模型	对应论文	所属类别	EM(Test)	F1(Test)	EM(dev)	F1(dev)	TOP
QUARK (arXiv 2020)	2.2	(1)	-	-	67.75	81.21	TOP-3
DFGN (ACL 2019)	3.2	(2)	56.31	69.69	-	-	
KGNN (arXiv 2019)	3.5	(2)	50.81	65.75	-	-	
HGN (EMNLP 2020)	3.6	(2)	69.22	82.19	-	-	TOP-1
C2F Reader (EMNLP 2020)	3.7	(2)	67.98	81.24	-	-	TOP-2
SAE (AAAI 2020)	3.8	(2)	66.92	79.62	67.70	80.75	TOP-4
ICLR 2020	5.7	(4)	-	-	81.2	68.0	
DECOMPRC (ACL 2019)	6.2	(5)	-	70.57	-	-	
Unsupervised (EMNLP 2020)	6.5	(5)	66.33	79.34	-	80.1	TOP-5
TheirNMN (EMNLP 2019)	7.1	(6)	49.58	62.71	50.67	63.35	
MODULARQA (arXiv 2020)	7.4	(6)	-	-	-	61.8	

类别信息

（1）基于单步阅读理解模型改进

（2）基于GNN

（3）迭代式文档检索

（4）推理链

（5）分解问题

（6）NMN

（7）PTM

https://github.com/krystalan/Multi-hopRC#leaderboard-of-hotpotqa

MhRC方法总览：百家争鸣

2. ANS (fullwiki setting)

模型	对应论文	所属类别	EM(Test)	F1(Test)	TOP
QUARK (arXiv 2020)	2.2	(1)	55.50	67.51	TOP-4
KGNN (arXiv 2019)	3.5	(2)	27.65	37.19	
HGN (EMNLP 2020)	3.6	(2)	59.74	71.41	TOP-3
MUPPET (ACL 2019)	4.2	(3)	31.07	40.42	
whole pip (EMNLP 2019)	4.3	(3)	45.30	57.30	
DDRQA (arXiv 2020)	4.6	(3)	62.9	76.9	TOP-1
CogQA (ACL 2019)	5.2	(4)	37.1	48.9	
GOLDEN (EMNLP 2019)	5.6	(4)	37.92	48.58	
ICLR 2020	5.7	(4)	60.0	73.0	TOP-2
Transformer-XH (ICLR 2020)	8.1	(7)	51.6	64.1	TOP-5

类别信息

（1）基于单步阅读理解模型改进

（2）基于GNN

（3）迭代式文档检索

（4）推理链

（5）分解问题

（6）NMN

（7）PTM

https://github.com/krystalan/Multi-hopRC#leaderboard-of-hotpotqa

MhRC方法总览：百家争鸣

3. 所属类别及入选TOP数量：

（1）基于单步阅读理解模型改进：1（其中一个同时入选两个setting）
（2）基于GNN：3（其中一个同时入选两个setting）
（3）迭代式文档检索：1
（4）推理链：1
（5）分解问题：1
（6）NMN：0
（7）PTM：1

Label 略

Part 3 基于单步阅读理解改进的MhRC

3 QUARK (2020 arXiv)

A Simple Yet Strong Pipeline for HotpotQA

Dirk Groeneveld† and **Tushar Khot**† and **Mausam**‡ and **Ashish Sabharwal**†

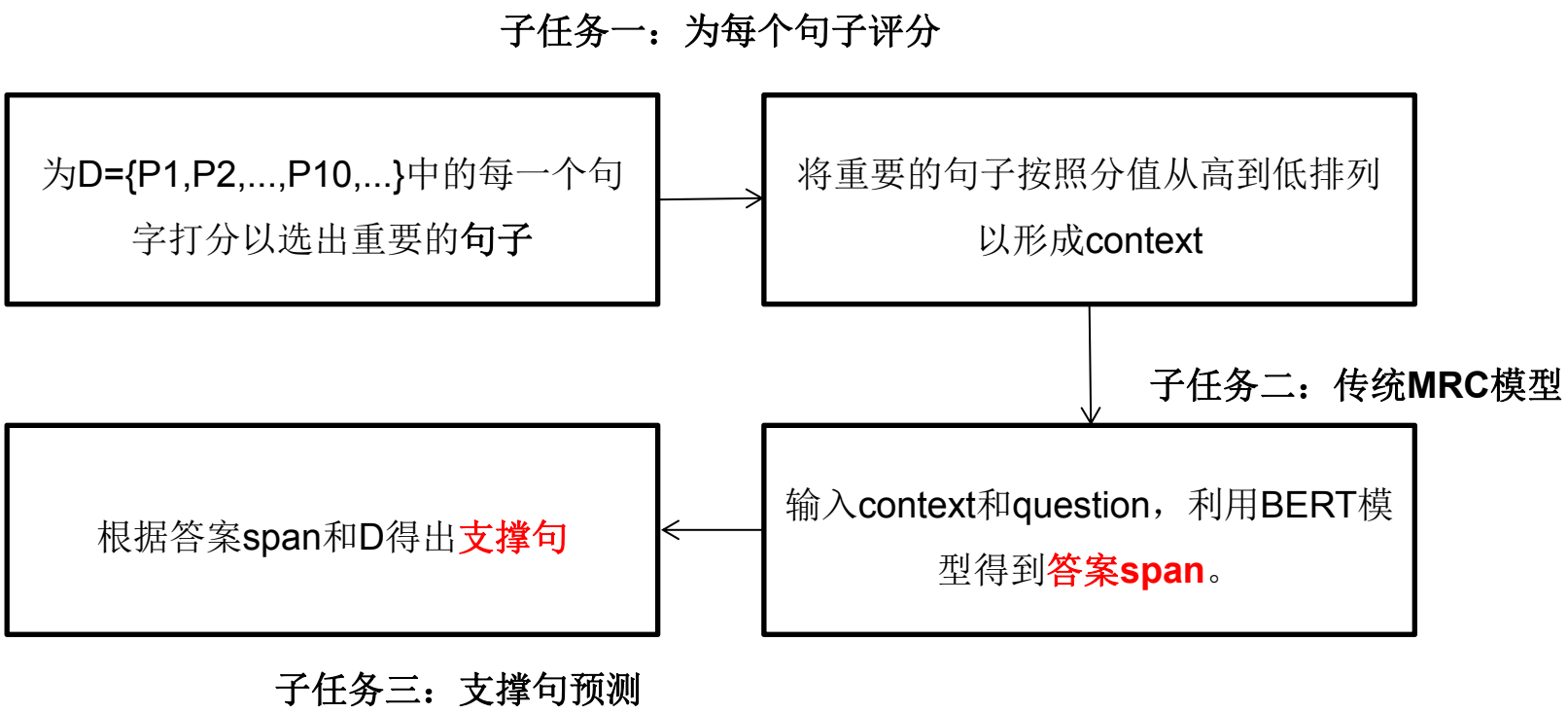
† Allen Institute for AI, Seattle, WA, U.S.A.

dirkg,tushark,ashishs@allenai.org

‡ Indian Institute of Technology, Delhi, India

mausam@cse.iitd.ac.in

3.1 概要


子任务一：为每个句子评分
为D={P1,P2,...,P10,...}中的每一个句子打分以选出重要的句子
将重要的句子按照分值从高到低排列以形成context
子任务二：传统MRC模型
输入context和question，利用BERT模型得到答案span。
根据答案span和D得出支撑句
子任务三：支撑句预测

3.2 模型-句子评分

- 对于HotpotQA(distractor setting)来说，给D={p1,p2,...,p10}中的每个句子打分。
- 对于HotpotQA(fullwiki setting)来说，利用SR-MRS[1](EMNLP 2019)检索相关文档，再对相关文档中的每个句子进行打分。

$$[CLS]\ question\ [SEP]\ paragraph\ [SEP]\ answer\ [SEP]$$

BERT ↓

利用[CLS]的表示进行二分类从而得到句子的重要程度

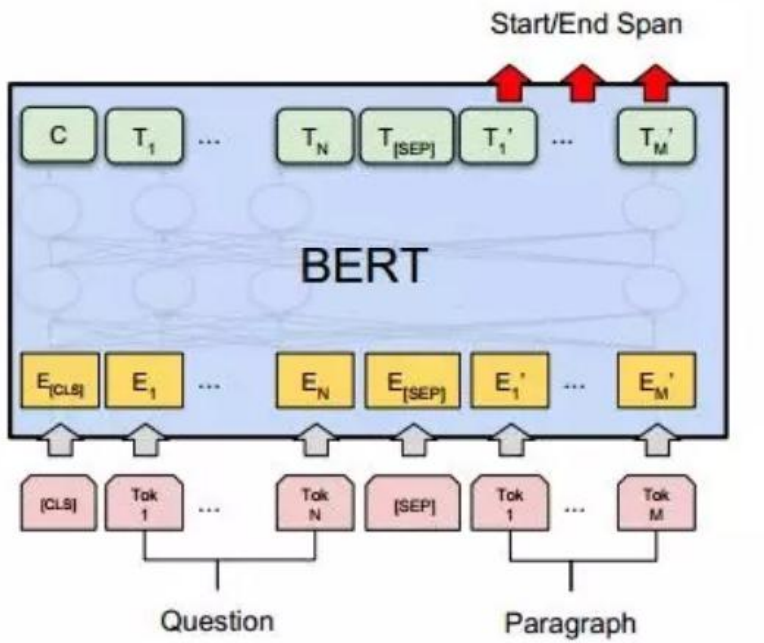
- Q1：answer字段当前明明没有答案？怎么还输入进来？
- Q2：不是对一个句子打分么，为啥输入的是paragraph？

- A1：当前没有答案，输入的是[MASK]。
- A2：将当前预测的句子的segment id设置为1，其余设置为0，就可以让模型区分当前预测的是哪句了。

1.Revealing the Importance of Semantic Retrieval for Machine Reading at Scale

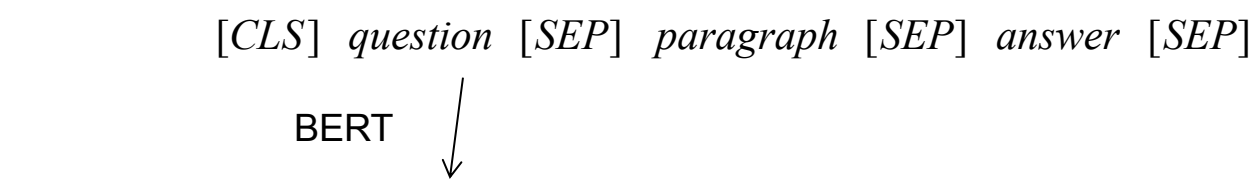
3.2 模型-答案span预测

将上述步骤中的句子按照评分从高到低排序，依次添加至context中，其中这些context中这些句子token化后的长度不应超过508（含[CLS]以及[SEP]），因为后面还有添加特殊token：yes, no, noans, [SEP]。



接着输入question和context，利用BERT模型预测出**答案span**。（类似于SQuAD）

3.3 模型-支撑句预测



利用[CLS]的表示进行二分类从而得到句子的重要程度

支撑句其实都是来源于两个文档中的。所以我们会根据不同文档中的所有句子的分数来选取得分最高的两个文档，再从这两个文档中选取得分较高的句子成为最终的支撑句。

3.4 实验结果：NO.3 in distractor

QA Model	Answer		Support		Joint	
	EM	F1	EM	F1	EM	F1
Single-paragraph (Min et al., 2019a)	–	67.08	–	–	–	–
QFE (Nishida et al., 2019)	53.70	68.70	58.80	84.70	35.40	60.60
DFGN (Xiao et al., 2019)	55.66	69.34	53.10	82.24	33.68	59.86
SAE (Tu et al., 2019)	61.32	74.81	58.06	85.27	39.89	66.45
HGN (Fang et al., 2019)	–	79.69	–	**87.38**	–	71.45
QUARK (Ours)	**67.75**	**81.21**	**60.72**	86.97	**44.35**	**72.26**
SAE (RoBERTa) (Tu et al., 2019)	67.70	80.75	**63.30**	87.38	**46.81**	72.75
HGN (RoBERTa) (Fang et al., 2019)	–	81.00	–	**87.93**	–	**73.01**

Table 1: HotpotQA's distractor setting, Dev set. The bottom two models use larger language models than QUARK.

QA Model	Answer		Support		Joint	
	EM	F1	EM	F1	EM	F1
QFE (Nishida et al., 2019)	28.66	38.06	14.20	44.35	8.69	23.10
SR-MRS (Nie et al., 2019)	45.32	57.34	38.67	70.83	25.14	47.60
QUARK + SR-MRS (Ours)	**55.50**	**67.51**	**45.64**	**72.95**	**32.89**	**56.23**
HGN (RoBERTa) + SR-MRS (Fang et al., 2019)	**56.71**	**69.16**	**49.97**	**76.39**	**35.36**	**59.86**

Table 2: HotpotQA's fullwiki setting, Test set. The bottom-most model uses a larger language model than QUARK.

Label 重点

Part 4 基于GNN的MhRC

- 4.1 SAE (AAAI 2020)
- 4.2 HGN (EMNLP 2020)

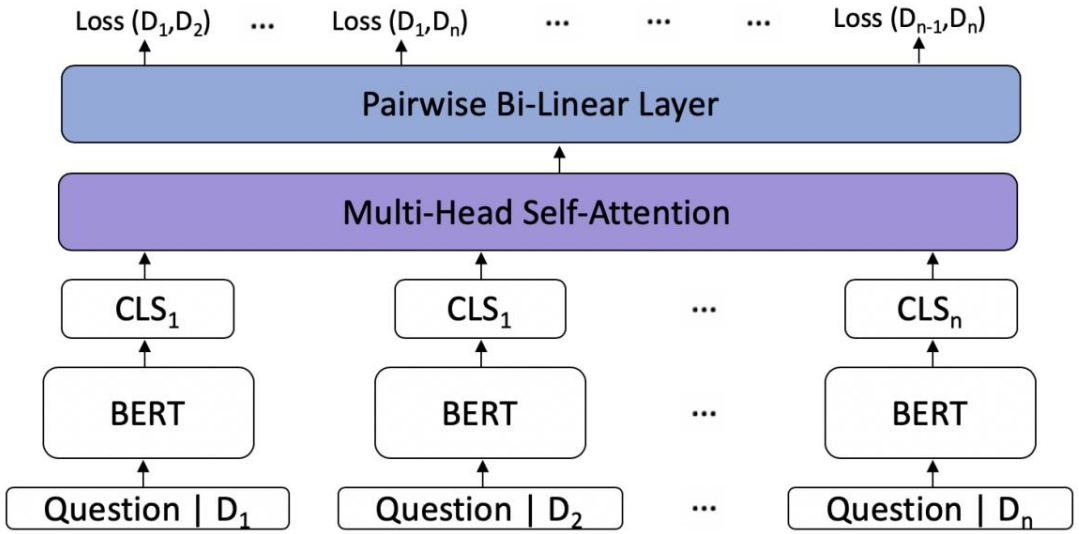
4.1 SAE (AAAI 2020)

Select, Answer and Explain: Interpretable Multi-hop Reading Comprehension over Multiple Documents

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, Bowen Zhou
JD AI Research
{ming.tu, kevin.huang3, guangtao.wang, jing.huang, xiaodong.he, bowen.zhou}@jd.com

- Select：从10篇文档中选择2篇作为后续步骤的输入。
- Answer and Explain：多任务学习（答案预测、支撑句预测）

4.1.1 选择文档部分



$$R_i = \sum_j^n F(P(D_i, D_j))$$

$$F(x) = \begin{cases} 1 & x > 0.5 \\ 0 & x \le 0.5 \end{cases}$$

1. 将每个文档以“[CLS] question [SEP] ducument [SEP]”的形式输入至BERT中，经过BERT后的 [CLS] token embedding表示整个文档的语义。

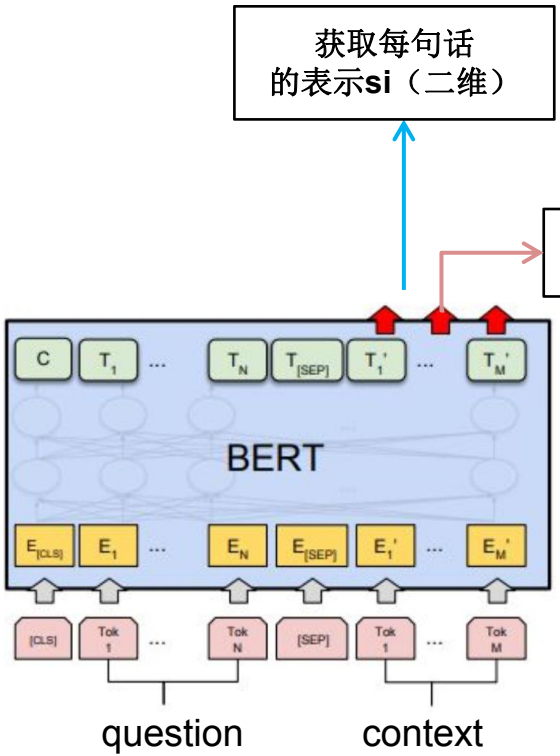
2. 在文档的表示间（各个[CLS]）搞一层多头注意力机制。

3. 基于每两个文档Di和Dj的表示，预测Di比Dj更靠谱的概率P(Di,Dj)。

4. 对于每个文档Di，计算其得分Ri，得分最高的两个文档将被选中进入下一轮。

4.1.2 答案预测与支撑句预测

基于每个token的表示，为开始/结尾的概率，计算一个权重，再根据权重将每个token embedding相加。



获取每句话的表示si（二维）

将si降至一维（attention）

2层MLP（768→2）
答案区间预测

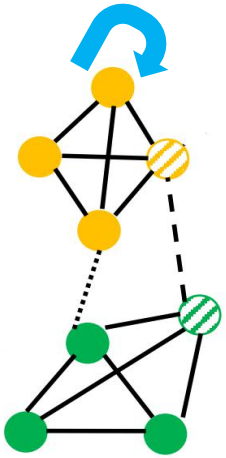
context-level

loss

$$L^{span} = \frac{1}{2}(CE(\hat{\mathbf{Y}}[:,0], \mathbf{y}^{start}) + CE(\hat{\mathbf{Y}}[:,1], \mathbf{y}^{end}))$$

bp

图注意力



1）同文档中不同句子结点相连。

2）不同文档但含有相同实体或名词短语的句子结点相连。

3）不同文档但均含有问题中的答案或名词短语的句子结点相连。

支撑句预测

sentence-level

loss

$$BCE(\hat{\mathbf{y}}^{sp}, \mathbf{y}^{sp})$$

bp

答案类型预测

3-class (Yes, No, Span)

context-level

loss

$$BCE(\hat{\mathbf{y}}^{sp}, \mathbf{y}^{sp})$$

bp

4.1.3 实验结果：NO.4 in distractor

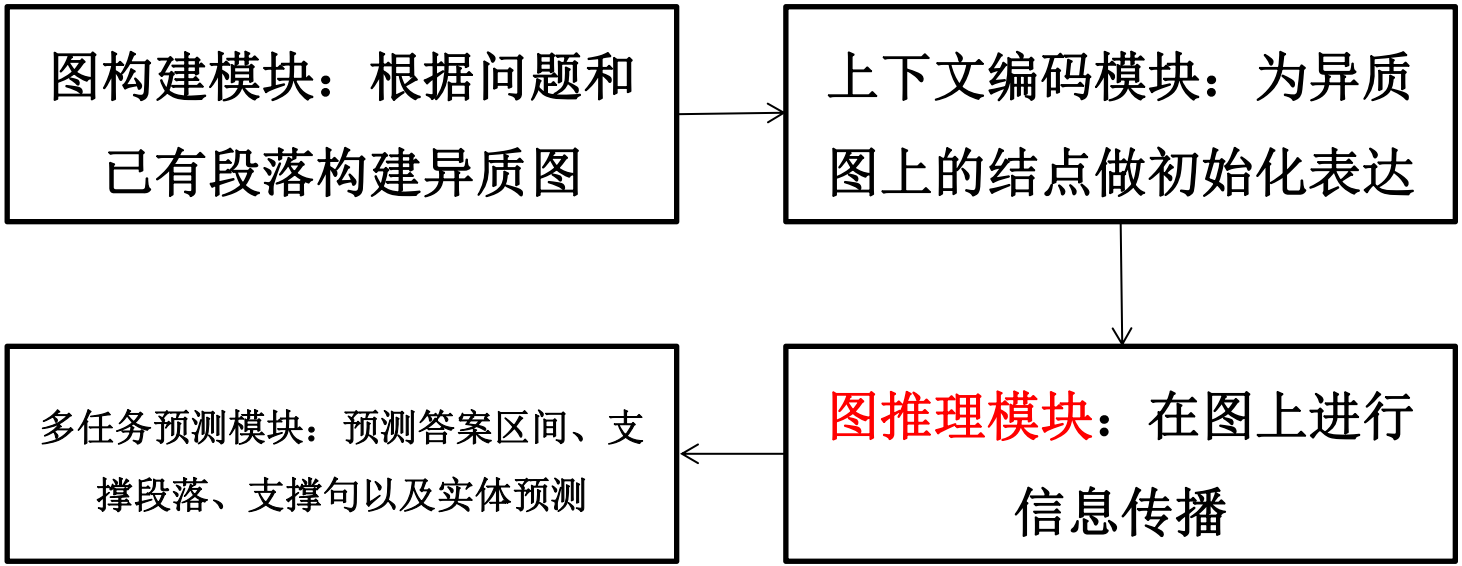
	Model	Ans		Sup		Joint	
		EM	F_1	EM	F_1	EM	F_1
Dev	Baseline(Yang et al. 2018)	44.44	58.28	21.95	66.66	11.56	40.86
	QFE(Nishida et al. 2019)	53.70	68.70	58.80	84.70	35.40	60.60
	DFGN(Xiao et al. 2019)	55.66	69.34	53.10	82.24	33.68	59.86
	SAE(ours)	61.32	74.81	58.06	85.27	39.89	66.45
	SAE-oracle(ours)	63.48	77.16	62.80	89.29	42.77	70.13
	SAE-large(ours)	67.70	80.75	63.30	87.38	46.81	72.75
Test	Baseline(Yang et al. 2018)	45.46	58.99	22.24	66.62	12.04	41.37
	QFE(Nishida et al. 2019)	53.86	68.06	57.75	84.49	34.63	59.61
	DFGN(Xiao et al. 2019)	56.31	69.69	51.50	81.62	33.62	59.82
	SAE(ours)	60.36	73.58	56.93	84.63	38.81	64.96
	SAE-large(ours)	66.92	79.62	**61.53**	86.86	**45.36**	71.45
	C2F Reader*	67.98	81.24	60.81	87.63	44.67	72.73

4.2 HGN (EMNLP 2020)

Hierarchical Graph Network for Multi-hop Question Answering

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, Jingjing Liu
Microsoft Dynamics 365 AI Research
{yuwfan,siqi.sun,zhe.gan,rohit.pillai,shuohang.wang,jingjl}@microsoft.com

4.2 HGN


图构建模块：根据问题和已有段落构建异质图
上下文编码模块：为异质图上的结点做初始化表达
图推理模块：在图上进行信息传播
多任务预测模块：预测答案区间、支撑段落、支撑句以及实体预测

4.2.1 HGN-图构建模块

Now ⇨ step 1：检索相关文档（First-hop&Second-hop）

step 2：利用相关文档构建异质图

First-hop中利用三种检索方法：

（1）标题匹配：若某一段落的标题中包含问题的任意一个短语，则该段落被匹配。

（2）实体匹配：若某一段落的标题中包含问题的任意一个实体，则该段落被匹配。

（3）基于RoBERTa的段落排序模型：输入问题和段落，利用RoBERTa进行编码，输出段落中包含支撑句的概率。

Second-hop：利用一跳检索到的文档中的超链接得到二跳文档集合。

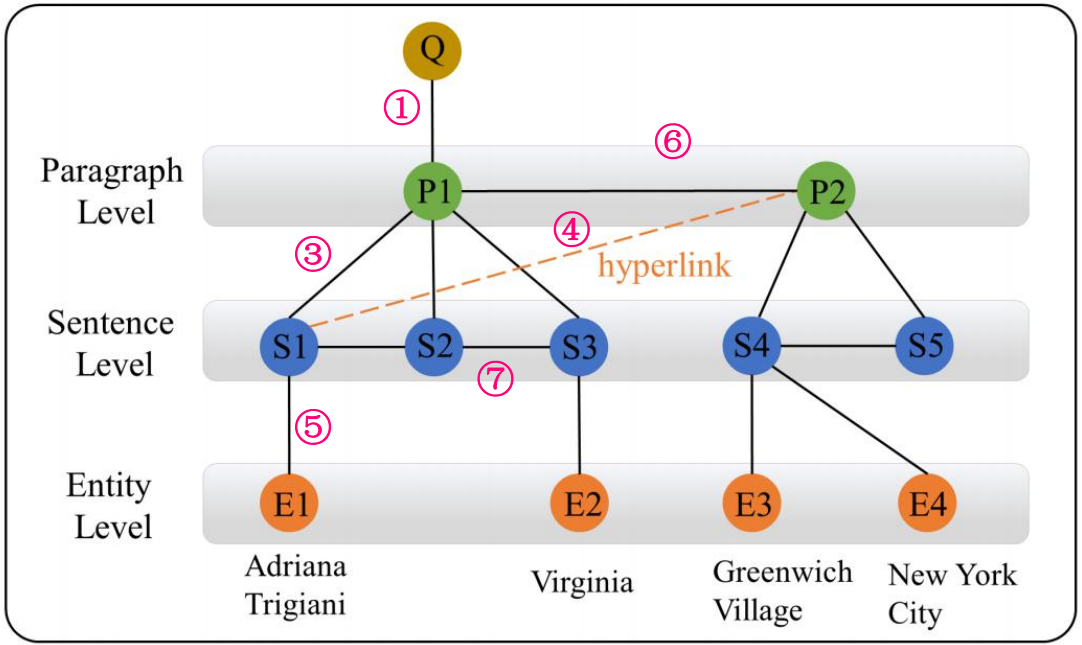
最终检索到的相关文档为一跳+二跳得到的文档集合中的**Top-N**个文档（利用RoBERTa进行计算）。

4.2.1 HGN-图构建模块

step 1：检索相关文档（First-hop&Second-hop）

Now ⇨ step 2：利用相关文档构建异质图

Graph Construction Module



四类结点（Q,P,S,E）、七类边：

①Q（问题结点）-P（段落结点）

②Q（问题结点）-E（实体结点）

③P（段落结点）-S（句子结点）：包含

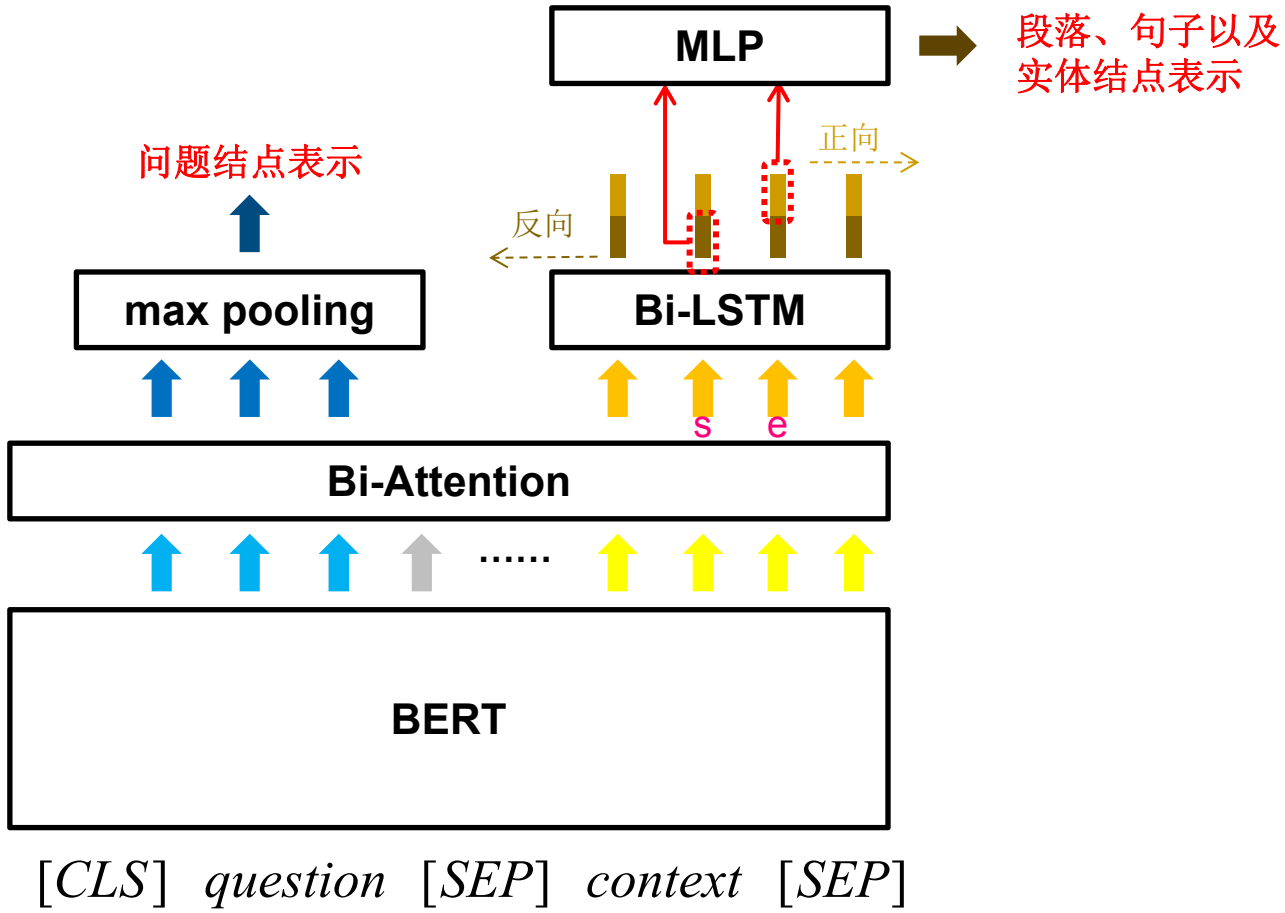
④S（句子结点）-P（段落结点）：链接

⑤S（句子结点）-E（实体结点）

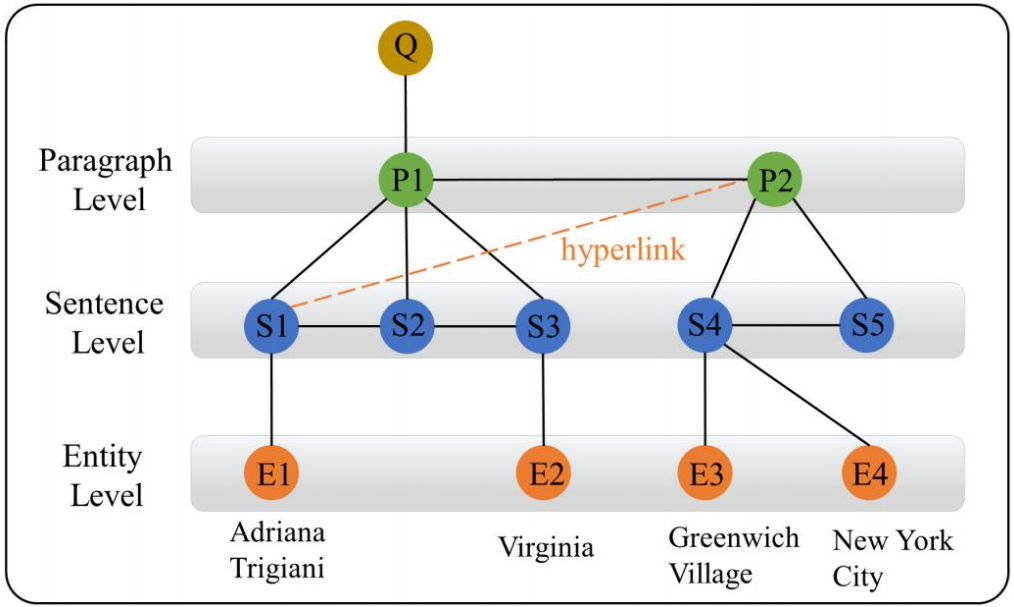
⑥段落间的边（Optionally）

⑦句子间的边（Optionally）

4.2.2 HGN-上下文编码模块

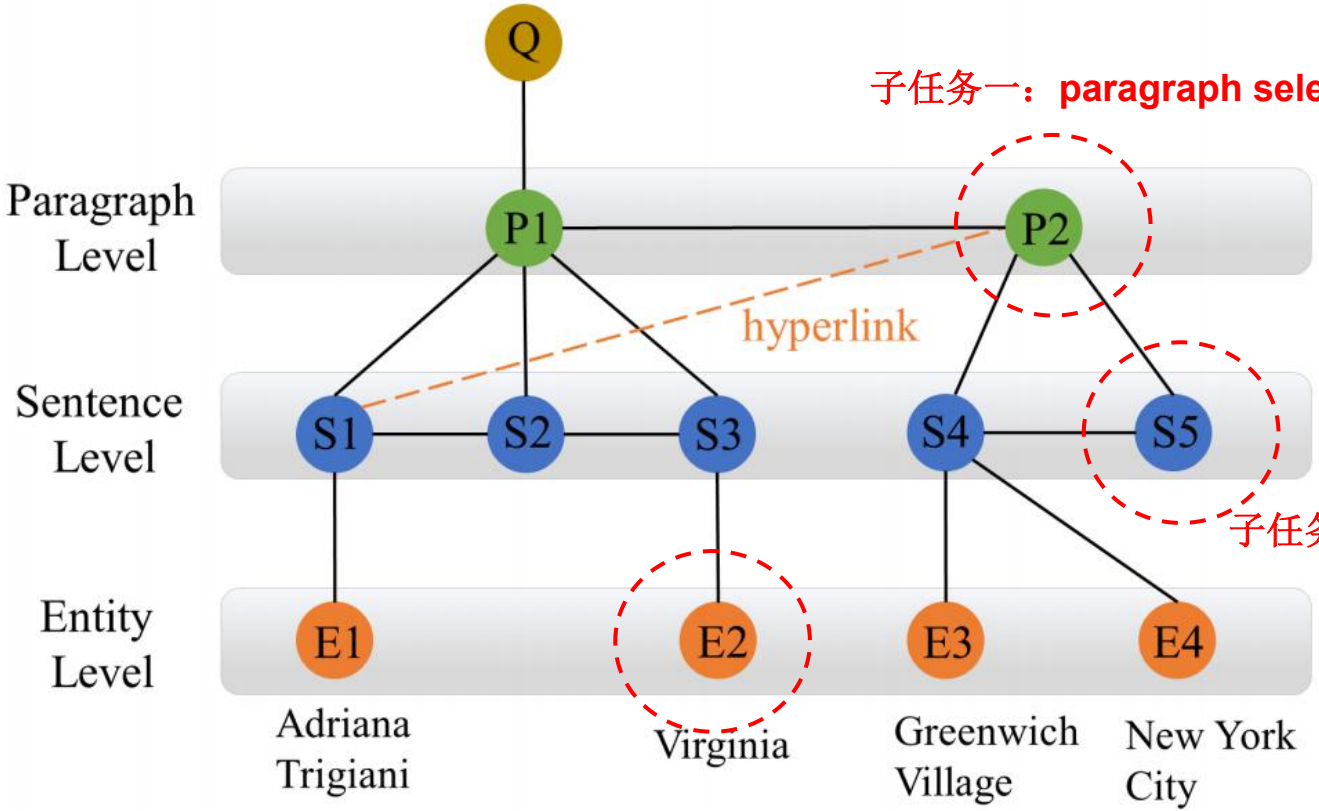


4.2.3 HGN-图推理模块



使用**Graph Attention Network**进行结点表示更新（推理）。

4.2.4 HGN-多任务预测模块



子任务一：**paragraph selection**

子任务二：**sentence selection**

子任务三：**answer prediction**
基于实体结点表示和上下文表示共同预测答案区间

子任务四：基于上下文表示进行问题类型预测。

4.2.5 实验结果：NO.1 in distractor ; No.3 in fullwiki

Model	Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1
DecompRC (Min et al., 2019b)	55.20	69.63	-	-	-	-
ChainEx (Chen et al., 2019)	61.20	74.11	-	-	-	-
Baseline Model (Yang et al., 2018)	45.60	59.02	20.32	64.49	10.83	40.16
QFE (Nishida et al., 2019)	53.86	68.06	57.75	84.49	34.63	59.61
DFGN (Xiao et al., 2019)	56.31	69.69	51.50	81.62	33.62	59.82
LQR-Net (Grail et al., 2020)	60.20	73.78	56.21	84.09	36.56	63.68
P-BERT†	61.18	74.16	51.38	82.76	35.42	63.79
TAP2 (Glass et al., 2019)	64.99	78.59	55.47	85.57	39.77	69.12
EPS+BERT†	65.79	79.05	58.50	86.26	42.47	70.48
SAE-large (Tu et al., 2020)	66.92	79.62	61.53	86.86	45.36	71.45
C2F Reader(Shao et al., 2020)	67.98	81.24	60.81	87.63	44.67	72.73
Longformer* (Beltagy et al., 2020)	68.00	81.25	63.09	88.34	45.91	73.16
ETC-large* (Zaheer et al., 2020)	68.12	81.18	**63.25**	**89.09**	46.40	73.62
HGN (ours)	**69.22**	**82.19**	62.76	88.47	**47.11**	**74.21**

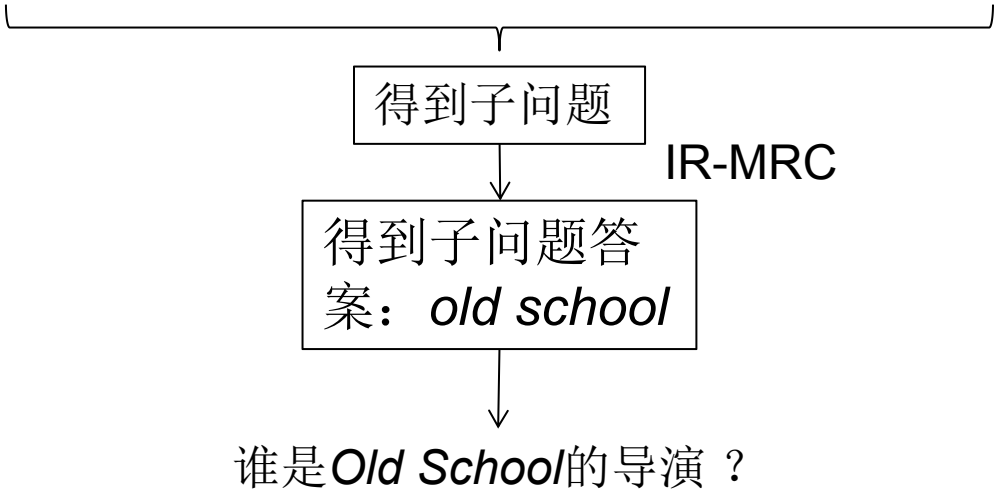
Label 重点

Part 5 基于迭代式文档检索的MhRC

- 5.1 分解问题流派
- 5.2 构建推理链流派
- 5.3 其他方法

5.1 分解问题流派

问题：谁是那场2013年在洛杉矶的皇家咖啡厅里取过景的电影的导演？

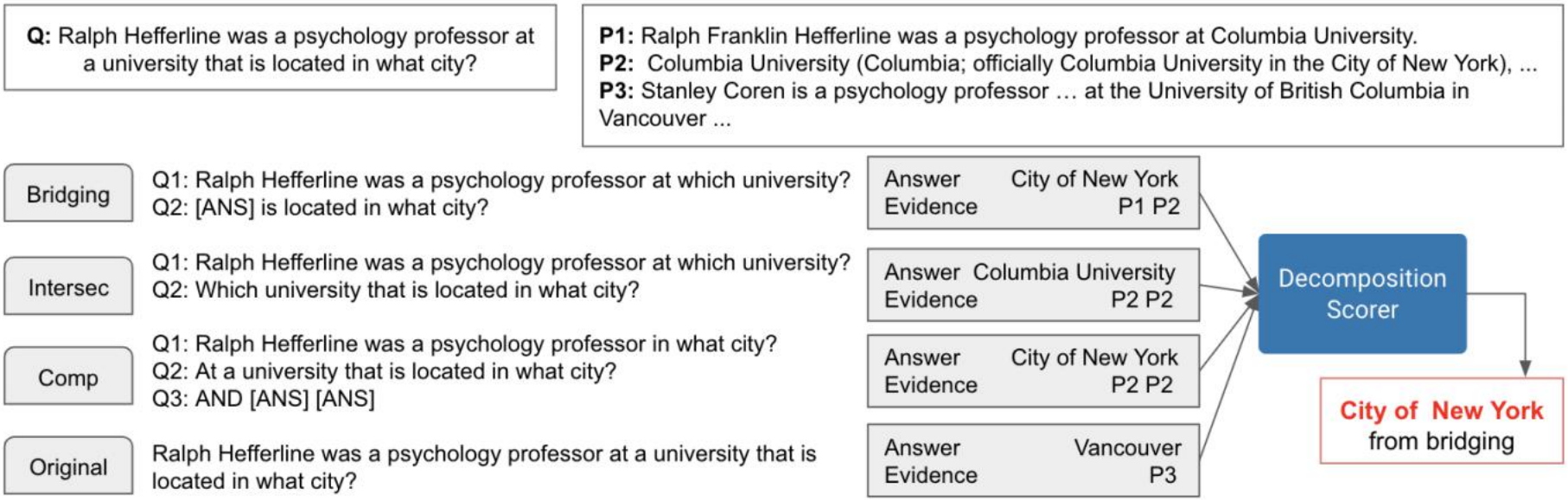


5.1.1 DecompRC (ACL 2019)

Multi-hop Reading Comprehension through Question Decomposition and Rescoring

Sewon Min[1], **Victor Zhong**[1], **Luke Zettlemoyer**[1], **Hannaneh Hajishirzi**[1,2]
[1]University of Washington
[2]Allen Institute for Artificial Intelligence
{sewon,vzhong,lsz,hannaneh}@cs.washington.edu

5.1.1 DecompRC总览



- step1：将一个问题分解成三种类型（Bridging类型、Intersection类型以及Compare类型）进行问题分解
- step2：对于每一种类型的分解，利用单跳阅读理解模型逐步计算答案并得到原始多跳问题的答案
- step3：将所有类型的答案分别打分，选择分值最高的答案作为最终预测结果。

5.1.1 DecompRC：问题分解

目的：将一个多跳问题分解成多个单跳问题。

分析：（1）使用seq2seq的方式产生子问题：太难了，需要大量的问题分解标注，标注成本过于高。而且seq2seq产生的结果不可控。

（2）基于Span预测和规则结合的方法产生子问题：相对容易，训练数据样本数量需要不算太高，但基于规则结合的方法可能对个别问题来说并不能够完美覆盖。

5.1.1 DecompRC：问题分解

对于Bridging类型的问题

“谁是那场2013年在洛杉矶的皇家咖啡厅里取过景的电影的导演？”

Q1：“那场2013年在洛杉矶的皇家咖啡厅里取过景的电影？”

Q2：谁是[ANS]的导演

“RH心理学教授所任教的大学坐落于哪个城市？”

Q1：“RH心理学教授所任教的大学”

Q2：“[ANS]坐落于哪个城市？”

对于Bridging问题而言，从原始多跳问题中寻找两个断点，然后利用规则形成两个单跳子问题。

5.1.1 DecompRC：问题分解

Algorithm 1 Sub-questions generation using Pointer_c.[2]

procedure GENERATESUBQ(Q : question, Pointer_c)

/* Find q_1^b and q_2^b for Bridging */
$\text{ind}_1, \text{ind}_2, \text{ind}_3 \leftarrow \text{Pointer}_3(Q)$
$q_1^b \leftarrow Q_{\text{ind}_1:\text{ind}_3}$
$q_2^b \leftarrow Q_{:\text{ind}_1}$: ANS : $Q_{\text{ind}_3:}$
article in $Q_{\text{ind}_2-5:\text{ind}_2} \leftarrow$ 'which'

/* Find q_1^i and q_2^i for Intersecion */
$\text{ind}_1, \text{ind}_2 \leftarrow \text{Pointer}_2(Q)$
$s_1, s_2, s_3 \leftarrow Q_{:\text{ind}_1}, Q_{\text{ind}_1:\text{ind}_2}, Q_{\text{ind}_2:}$
if s_2 starts with wh-word **then**
  $q_1^i \leftarrow s_1 : s_2,\ q_2^i \leftarrow s_2 : s_3$
else
  $q_1^i \leftarrow s_1 : s_2,\ q_2^i \leftarrow s_1 : s_3$

/* Find q_1^c, q_2^c and q_3^c for Comparison */
$\text{ind}_1, \text{ind}_2, \text{ind}_3, \text{ind}_4 \leftarrow \text{Pointer}_4(Q)$
$\text{ent}_1, \text{ent}_2 \leftarrow Q_{\text{ind}_1:\text{ind}_2}, Q_{\text{ind3:ind}_4}$
$op \leftarrow \text{find\_op}(Q, \text{ent}_1, \text{ent}_2)$
$q_1^c, q_2^c \leftarrow \text{form\_subq}(Q, \text{ent}_1, \text{ent}_2, op)$
$q_3^c \leftarrow op\ (\text{ent}_1, \text{ANS})\ (\text{ent}_2, \text{ANS})$

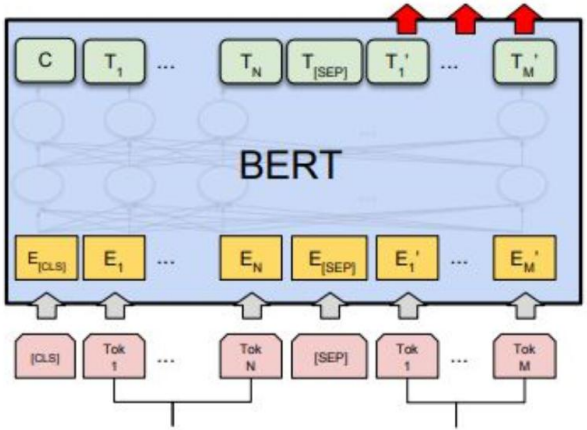
根据bridging类型产生分解问题的结果

根据intersection类型产生分解问题的结果

根据compare类型产生分解问题的结果

5.1.1 DecompRC：回答单跳子问题

根据当前需要回答的单跳问题，先利用 Paragraph Selection[1]检索到相关段落，$P_1, P_2, ..., P_N$ 再通过BERT模型预测出答案span。



单跳子问题 P_i

（1）任务类型预测

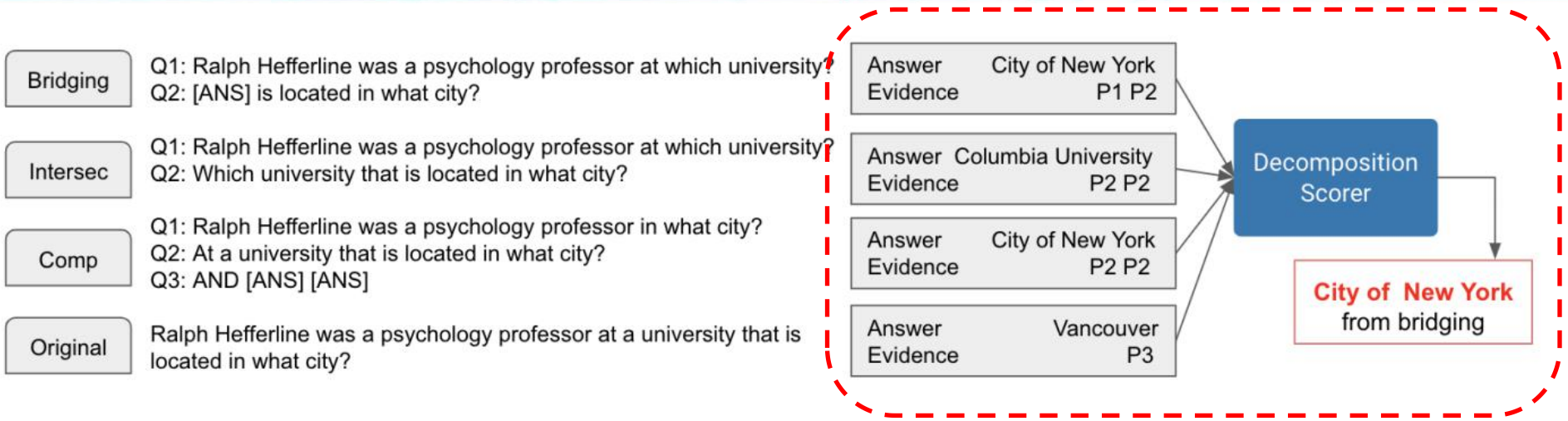
$$[y_i^{\text{span}}; y_i^{\text{yes}}; y_i^{\text{no}}; y_i^{\text{none}}] = \max(U_i)W_1 \in \mathbb{R}^4$$

（2）答案span预测

最终单跳问题的答案是y_none最小的段落中计算出的答案。

1.Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In ACL.

5.1.1 DecompRC：预测最终答案



step1：给BERT输入question和推理类型（concat），得到二维表示$U_t \in R^{n \times h}$

step2：计算该推理类型的概率 $p_t = \text{sigmoid}(W_2^T \max(U_t)) \in \mathbb{R}$

step3：将概率最高的推理类型对应的答案视为该多跳问题的**最终答案**。

5.1.2 ONUS (EMNLP 2020)

Unsupervised Question Decomposition for Question Answering

Ethan Perez[1 2] **Patrick Lewis**[1 3]
Wen-tau Yih[1] **Kyunghyun Cho**[2 4*] **Douwe Kiela**[1]
[1]Facebook AI Research, [2]New York University,
[3]University College London, [4]CIFAR Azrieli Global Scholar
perez@nyu.edu

ONUS：**O**ne-to-**N** **U**nsupervised **S**equence transduction

5.1.2 ONUS：动机

DecompRC的缺点

- 使用的方法非常基于人工设定的规则。只针对于HotpotQA数据集，out-of-domain上的效果肯定非常差。
- 一个复杂问题分解后的子问题所使用的词汇，一定属于原问题当中使用的词汇集合么？

目标

- 不要人工设计那种模板来搞，希望搞点不仅适用于HotpotQA的，在真实场景也能起点作用的。
- 希望在分解问题的时候，不要局限于原文本。
- 同时不用标注数据。

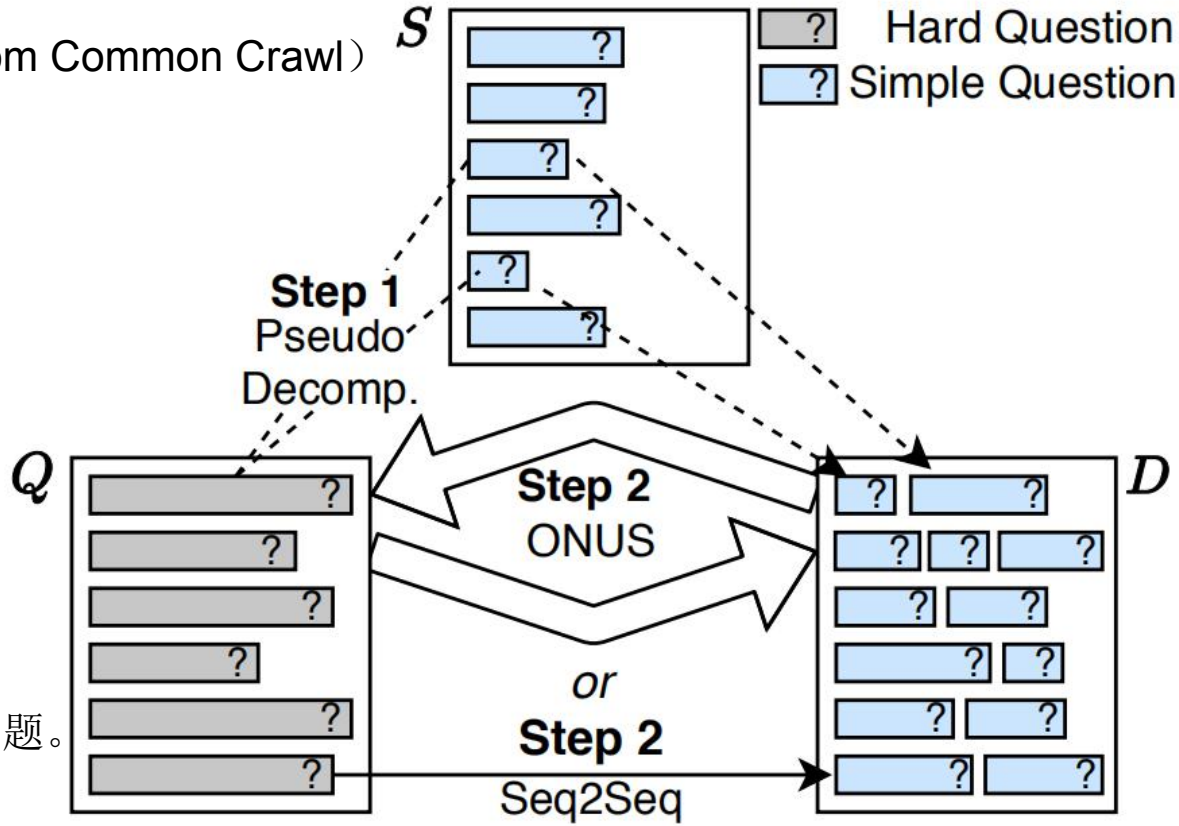
5.1.2 ONUS：动机

在机器翻译（NMT）领域，机器翻译的效果对训练数据的规模和质量十分敏感。所以NMT中有一个大的研究方向就是：**如何构建一个质量高并且数量多的平行语料库**。

BUCC (Building and Using Comparable Corpora) share task：在训练部分给了ground truth平行语料，然后还给了大量的不同语言的句子，需要设计模型去对齐这些句子以构成新的平行语料。

5.1.2 ONUS：总览

S：简单问题语料库（10M sub-question from Common Crawl）



Q：复杂问题集合，例如HotpotQA中的所有复杂问题。

5.1.2 ONUS-step1：伪分解数据集构建

对于一个给定的复杂问题q。从S（简单问题语料库）中检索到N个简单子问题：$d^{'} = \{s_1, s_2, ..., s_N\}$

具体检索方法是：

$$\underset{d' \subset S}{\operatorname{argmax}} \sum_{s_i \in d'} f(q, s_i) - \sum_{s_i, s_j \in d', i \neq j} f(s_i, s_j)$$

每一个简单问题都与原问题有比较高的关联

不同的简单问题之间的关联尽量少一些。

后处理：对一个简单子问题si中所包含的每一个没有出现在q中的实体，替换成q中出现过的同类型实体。

5.1.2 ONUS-step2：训练模型分解

approach 1：PseudoD

对于测试集中的一个复杂问题，直接用刚才所讲的步骤从S（简单问题语料库）中检索到两个子问题。之后对子问题进行后处理。

approach 2：Seq2seq

利用上一步构建的伪分解数据集训练一个seq2seq模型，对于测试集中的一个复杂问题，利用训练好的seq2seq模型预测其分解后的简单问题。

approach 3：ONUS

在训练seq2seq分解问题的同时还会利用回译与去噪任务对模型进行辅助训练。

回译：输入多个单跳子问题，输出多跳问题

去噪：输入加噪（随机mask，丢弃，局部打乱等）之后的多跳问题或单跳问题，输出原始问题。

5.1.2 ONUS-step3：复杂问题答案

- 给定一个复杂问题，利用预训练好的ONUS模型得到多个分解后的简单子问题。
- 接着对每个简单子问题，得到对应的子问题答案（利用已有单跳QA模型）
- 给BERT输入[CLS] question [SEP] sub-question1 sub-answer1 [SEP] sub-question2 sub-answer2[SEP]，预测最终答案区间。

5.1.2 ONUS 实验结果：NO.5 in distractor

Decomp. Method	Pseudo-Decomps.	HotpotQA Dev F1 Orig	Multi	OOD
✗	✗ (1hop)	66.7	63.7	66.5
✗	✗ (Baseline)	$77.0_{\pm.2}$	$65.2_{\pm.2}$	$67.1_{\pm.5}$
PseudoD	Random	$78.4_{\pm.2}$	$70.9_{\pm.2}$	$70.7_{\pm.4}$
	FastText	$78.9_{\pm.2}$	$72.4_{\pm.1}$	$72.0_{\pm.1}$
Seq2Seq	Random	$77.7_{\pm.2}$	$69.4_{\pm.3}$	$70.0_{\pm.7}$
	FastText	$78.9_{\pm.2}$	$73.1_{\pm.2}$	$73.0_{\pm.3}$
ONUS	Random	$79.8_{\pm.1}$	$76.0_{\pm.2}$	$76.5_{\pm.2}$
	FastText	$\mathbf{80.1}_{\pm.2}$	$\mathbf{76.2}_{\pm.1}$	$\mathbf{77.1}_{\pm.1}$
DecompRC*		$79.8_{\pm.2}$	$76.3_{\pm.4}$	$77.7_{\pm.2}$
SAE (Tu et al., 2020) †		80.2	61.1	62.6
HGN (Fang et al., 2019) †		82.2	78.9‡	76.1‡

	Ours	SAE†	HGN†
Test (EM/F1)	66.33/79.34	66.92/79.62	69.22/82.19

5.1.3 BREAK（TACL 2020）

BREAK It Down: A Question Understanding Benchmark

Tomer Wolfson[1,3], Mor Geva[1,3], Ankit Gupta[1],
Matt Gardner[3], Yoav Goldberg[2,3], Daniel Deutch[1], Jonathan Berant[1,3]

[1]Tel Aviv University [2]Bar-Ilan University [3]Allen Institute for AI
{tomerwol,morgeva}@mail.tau.ac.il, ankitgupta.iitkanpur@gmail.com,
mattg@allenai.org, yoav.goldberg@gmail.com,
danielde@post.tau.ac.il, joberant@cs.tau.ac.il

从10个复杂问题的QA数据集中提取问题并标注分解，一共定义了13个基本分解操作，以及3个高级分解。最终构建了一个包含了83978个复杂问题及其对应的分解的数据集BREAK。

5.1.3 BREAK

Operator	Template / Signature	Question	Decomposition
Select	Return [entities] $w \rightarrow S_e$	How many touchdowns were scored overall?	1. **Return touchdowns** 2. Return the number of #1
Filter	Return [ref] [condition] $S_o, w \rightarrow S_o$	I would like a flight from Toronto to San Diego please.	1. Return flights 2. **Return #1 from Toronto** 3. **Return #2 to San Diego**
Project	Return [relation] of [ref] $w, S_e \rightarrow S_o$	Who is the head coach of the Los Angeles Lakers?	1. Return the Los Angeles Lakers 2. **Return the head coach of #1**
Aggregate	Return [aggregate] of [ref] $w_{agg}, S_o \rightarrow n$	How many states border Colorado?	1. Return Colorado 2. Return border states of #1 3. **Return the number of #2**
Group	Return [aggregate] [ref1] for each [ref2] $w_{agg}, S_o, S_e \rightarrow S_n$	How many female students are there in each club?	1. Return clubs 2. Return female students of #1 3. **Return the number of #2 for each #1**
Superlative	Return [ref1] where [ref2] is [highest / lowest] $S_e, S_n, w_{sup} \rightarrow S_e$	What is the keyword, which has been contained by the most number of papers?	1. Return papers 2. Return keywords of #1 3. Return the number of #1 for each #2 4. **Return #2 where #3 is highest**
Comparative	Return [ref1] where [ref2] [comparison] [number] $S_e, S_n, w_{com}, n \rightarrow S_e$	Who are the authors who have more than 500 papers?	1. Return authors 2. Return papers of #1 3. Return the number of #2 for each of #1 4. **Return #1 where #3 is more than 500**
Union	Return [ref1] , [ref2] $S_o, S_o \rightarrow S_o$	Tell me who the president and vice-president are?	1. Return the president 2. Return the vice-president 3. **Return #1 , #2**
Intersection	Return [relation] in both [ref1] and [ref2] $w, S_e, S_e \rightarrow S_o$	Show the parties that have representatives in both New York state and representatives in Pennsylvania state.	1. Return representatives 2. Return #1 in New York state 3. Return #1 in Pennsylvania state 4. **Return parties in both #2 and #3**

5.1.3 BREAK

Algorithm 1 BREAKRC

```
1: procedure BREAKRC(s : QDMR)
2:     ansrs ← []
3:     for s^i in s = ⟨s^1, ..., s^n⟩ do
4:         op ← OPTYPE(s^i)
5:         refs ← REFERENCEDSTEPS(s^i)
6:         if op is SELECT then
7:             ans ← ANSWER(s^i)
8:         else if op is FILTER then
9:             ŝ^i ← EXTRACTQUESTION(s^i)
10:            ans_tmp ← ANSWER(ŝ^i)
11:            ans ← INTERSECT(ans_tmp, ansrs[refs[0]])
12:        else if op is COMPARISON then
13:            ans ← COMPARESTEPS(refs,s)
14:        else                                  ▷ op is PROJECT
15:            ŝ^i ← SUBSTITUTEREF(s^i, ansrs[refs[0]])
16:            ans ← ANSWER(ŝ^i)
17:        ansrs[i] ← ans
18:    return ansrs[n]
```

5.2 构建推理链流派

LEARNING TO RETRIEVE REASONING PATHS OVER WIKIPEDIA GRAPH FOR QUESTION ANSWERING

Akari Asai[*†], **Kazuma Hashimoto**[‡], **Hannaneh Hajishirzi**[†§], **Richard Socher**[‡] **& Caiming Xiong**[‡]
[†]University of Washington [‡]Salesforce Research [§]Allen Institute for Artificial Intelligence
{akari,hannaneh}@cs.washington.edu
{k.hashimoto,rsocher,cxiong}@salesforce.com

ICLR 2020

5.2.1 动机

第二个Supporting Paragraph并不能通过问题直接检索得到（因为词汇相似度低）

5.2.2 总览

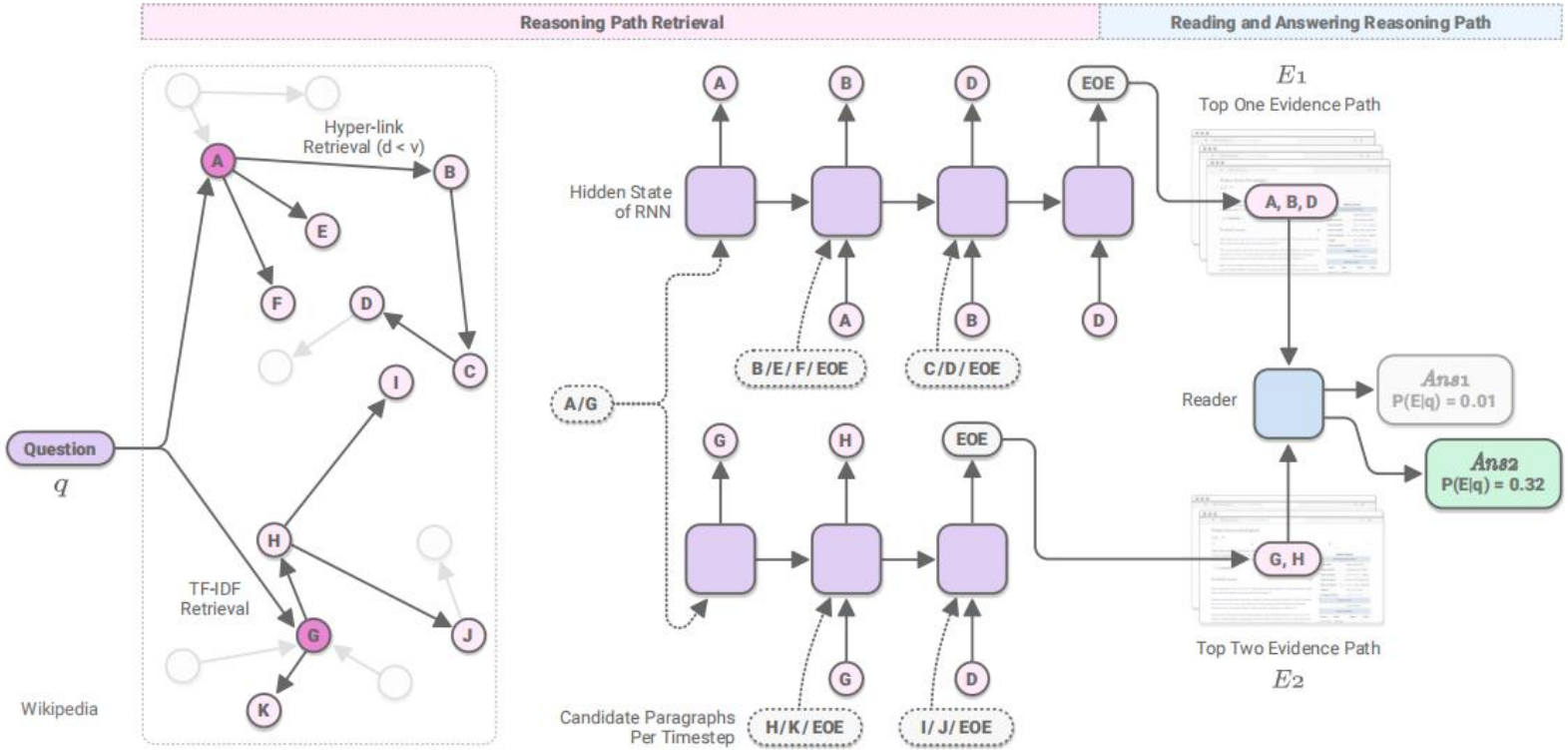
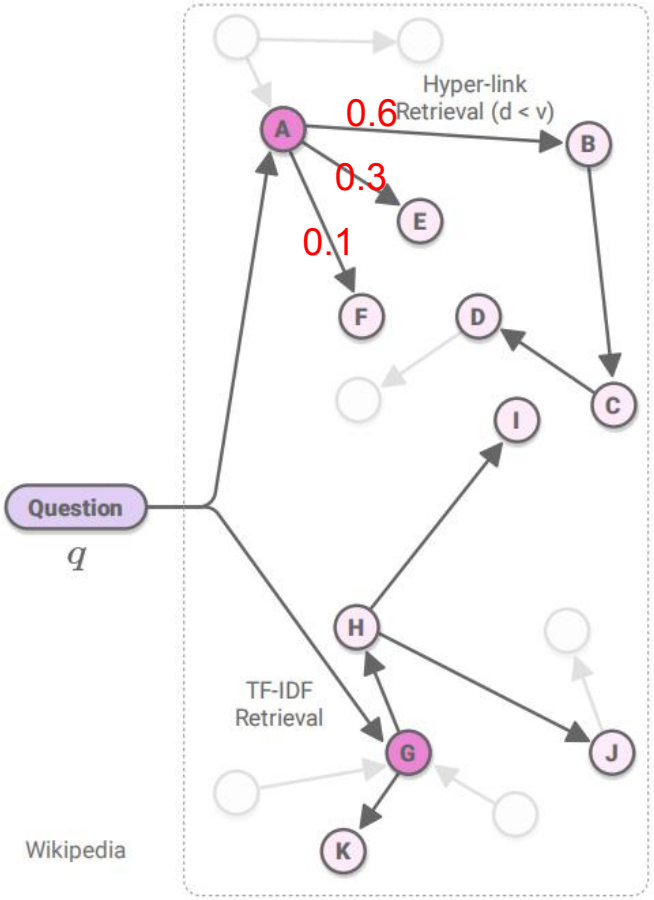


Figure 2: Overview of our framework.

5.2.3 Reasoning Path Retrieval



使用RNN保存已经检索到的路径的全部信息（就将已检索到的所有文档用RNN进行编码，那么RNN的隐层向量就包含了所有已检索到的文档的信息）

$$w_i = \mathrm{BERT}_{[\mathrm{CLS}]}(q, p_i) \in \mathbb{R}^d,$$
$$P(p_i|h_t) = \sigma(w_i \cdot h_t + b),$$
$$h_{t+1} = \mathrm{RNN}(h_t, w_i) \in \mathbb{R}^d,$$

为了让推理链终止，有一个特殊的符号[EOE]。

5.2.4 实验结果：No.2 in fullwiki

Models	full wiki				distractor			
	QA		SP		QA		SP	
	F1	EM	F1	EM	F1	EM	F1	EM
Semantic Retrieval (Nie et al., 2019)	58.8	46.5	71.5	39.9	–	–	–	–
GoldEn Retriever (Qi et al., 2019)	49.8	–	64.6	–	–	–	–	–
Cognitive Graph (Ding et al., 2019)	49.4	37.6	58.5	23.1	–	–	–	–
DecompRC (Min et al., 2019c)	43.3	–	–	–	70.6	–	–	–
MUPPET (Feldman & El-Yaniv, 2019)	40.4	31.1	47.7	17.0	–	–	–	–
DFGN (Xiao et al., 2019)	–	–	–	–	69.2	55.4	–	–
QFE (Nishida et al., 2019)	–	–	–	–	68.7	53.7	84.7	**58.8**
Baseline (Yang et al., 2018)	34.4	24.7	41.0	5.3	58.3	44.4	66.7	22.0
Transformer-XH (Zhao et al., 2020)	62.4	50.2	71.6	42.2	–	–	–	–
Ours (Reader: BERT wwm)	**73.3**	**60.5**	**76.1**	**49.3**	**81.2**	**68.0**	**85.2**	58.6
Ours (Reader: BERT base)	65.8	52.7	75.0	47.9	73.3	59.4	84.6	57.4

5.3 其他方法

DDRQA: Dynamic Document Reranking for Open-domain Multi-hop Question Answering

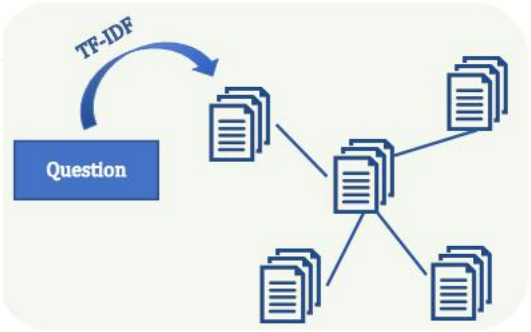
Yuyu Zhang[*,1] **Ping Nie**[*,2], **Arun Ramamurthy**[3], **Le Song**[1]
[1]Georgia Institute of Technology
[2]Peking University
[3]Siemens Corporate Technology
yuyu@gatech.edu, ping.nie@pku.edu.cn
arun.ramamurthy@siemens.com, lsong@gatech.edu

arXiv 2020

5.3.1 DDRQA动机

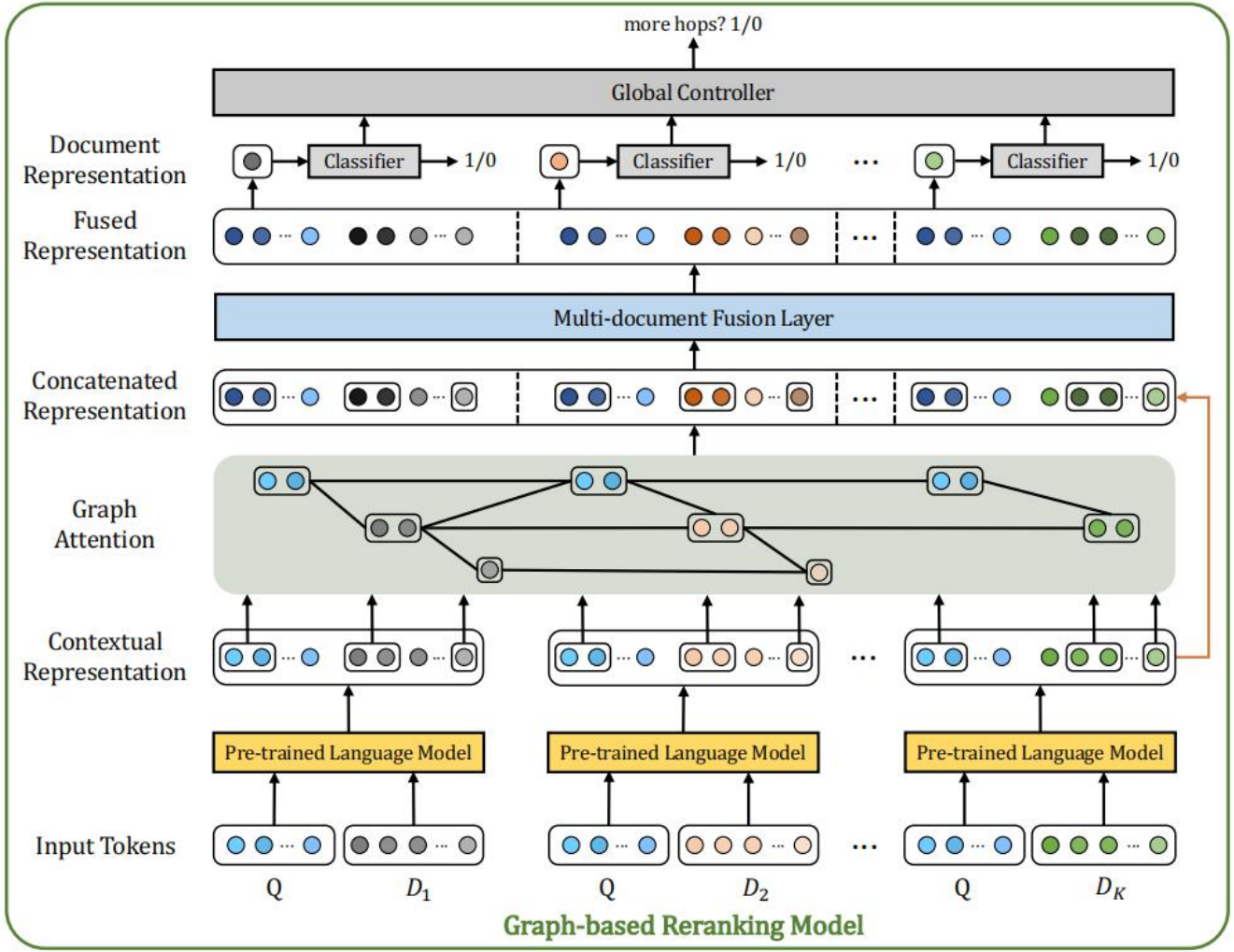
- 在Open Domain QA的场景下，**检索非常重要**，尤其是对多跳问题而言。
- 在该场景下，最经典的框架就是**Retriever and Reader**。
- Retriever负责检索与复杂问题相关的文档，Reader阅读检索到的文档并预测答案。
- 本文的工作集中在Retriever上，**提出了一种动态迭代检索的思路**，而在Reader上只采用了非常标准的span预测方法。

5.3.2 流程



- 先利用TF-IDF检索top N个文档
- 利用这N个文档建立一个图结构：每个结点代表一个文档。如果两个文档存在共享实体，那么这两个文档相连。接着对这个文档图进行**基于图结构的排序**模块算法，为每个文档打分（0~1），只保留topK个文档，其余全部去除。
- **判读是否继续进行下一轮检索：如果positive文档数大于某阈值则继续检索**
- 在已经检索到的文档上进行**span预测**，将span预测的结果作为新query继续利用基于TF-IDF的方法去检索更多的文档，并将新检索到的文档补充到图中。
- 一旦检索过程结束，最终得分最高的k个文档进行Reader阶段以预测最终答案。

5.3.3 基于图结构的排序算法



5.3.4 产生新query

基于**问题**以及**当前检索到的文档**，在当前检索到的文档上**预测一个span**，该span用于下一次检索。

文本在产生新query时借鉴了GoldEn（EMNLP 2019），GoldEn在产生新query时用到了DrQA（ACL 2017）的Reader部分。

预测span训练集的产生：当前已得到的文档和我们期望检索文档之间具有最高重叠率的连续跨度（计算重叠率的时候测试了多种方法，例如最长公共子序列、最长公共子串、前两者的结合）。

Part 6 基于神经模块网络的MhRC

6.1 NMN（神经模块网络）介绍

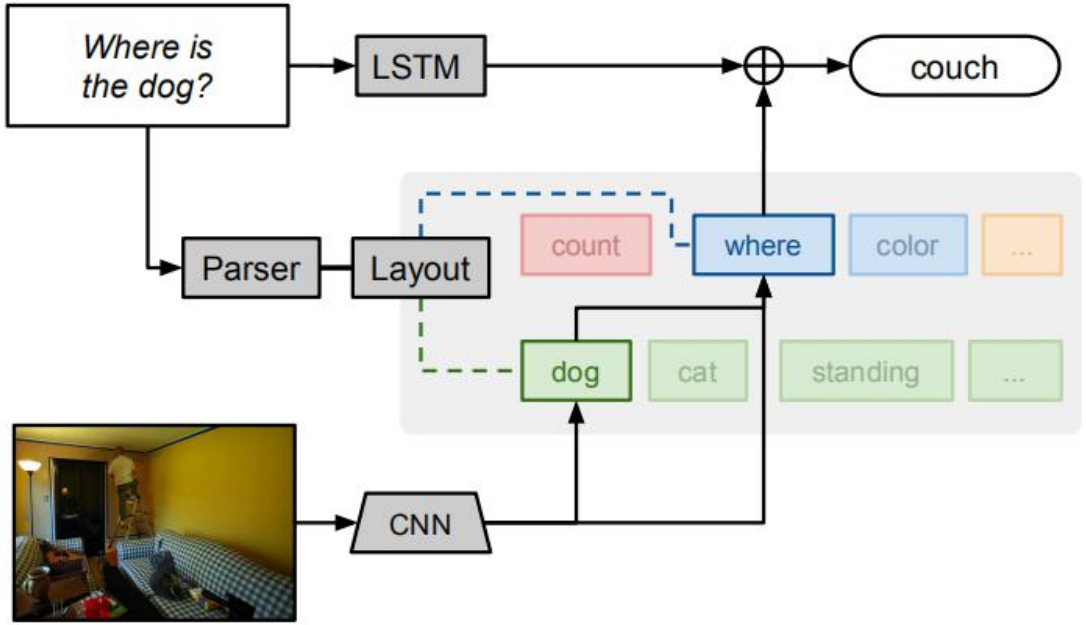
Deep Compositional Question Answering with Neural Module Networks

Jacob Andreas Marcus Rohrbach Trevor Darrell Dan Klein
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
{jda,rohrbach,trevor,klein}@{cs,eecs,eecs,cs}.berkeley.edu

神经模块网络在CVPR 2016的一篇论文中提出，用于解决VQA问题，后被引入多跳阅读理解。

其核心思想在于：对于每一种类型的操作都定义一个基于神经网络的模块来完成，对于一个复杂问题，会根据问题实例，预测出需要哪些模块来共同解决这个问题，每个模块完成这个问题实例中的一小步。

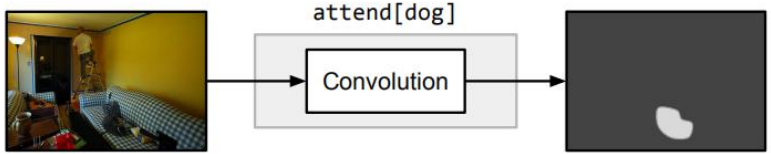
6.1 NMN for VQA



6.1 NMN for VQA

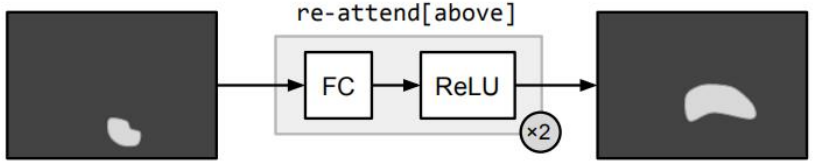
Attention

attend : $Image \rightarrow Attention$



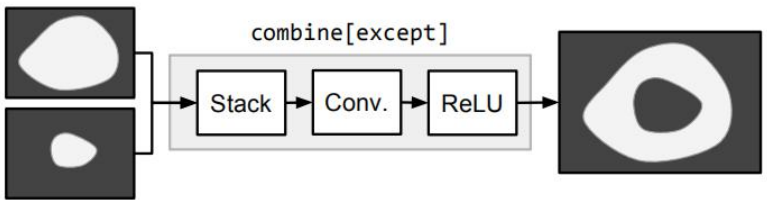
Re-attention

re-attend : $Attention \rightarrow Attention$



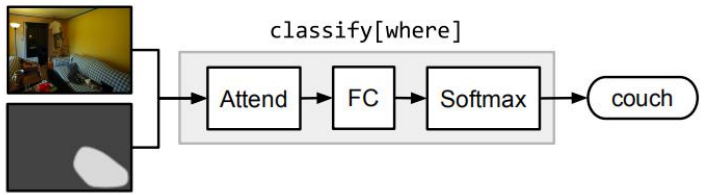
Combination

combine : $Attention \times Attention \rightarrow Attention$



Classification

classify : $Image \times Attention \rightarrow Label$



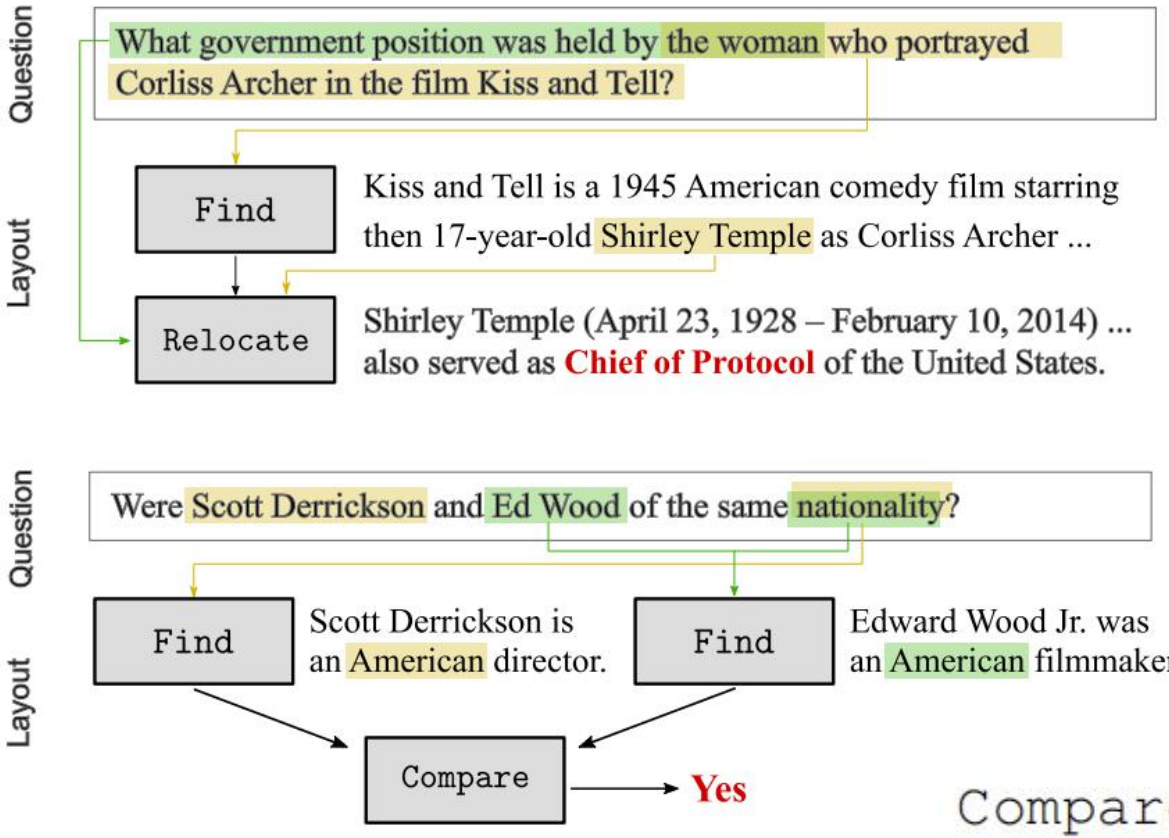
6.2 NMN for MhRC

Self-Assembling Modular Networks for Interpretable Multi-Hop Reasoning

Yichen Jiang and **Mohit Bansal**
UNC Chapel Hill
{yichenj, mbansal}@cs.unc.edu

EMNLP 2019

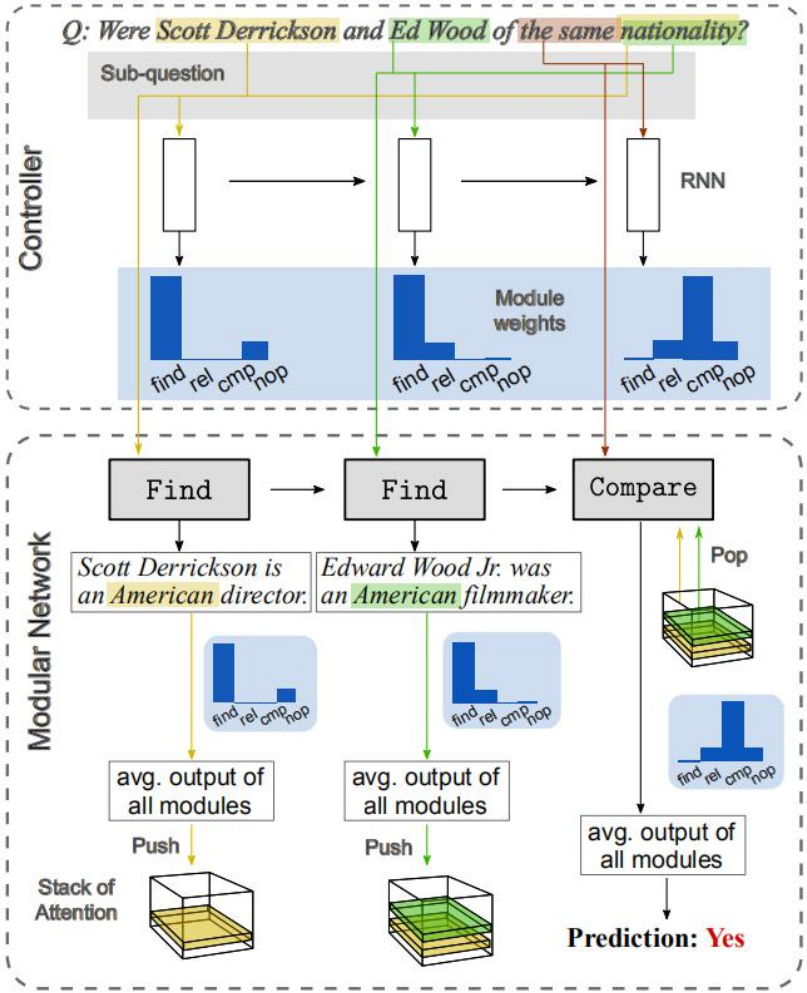
6.2.1 总览




```
Relocate(Find())
```

```
Compare(Find(), Find())
```

6.2.2 模型：Controller

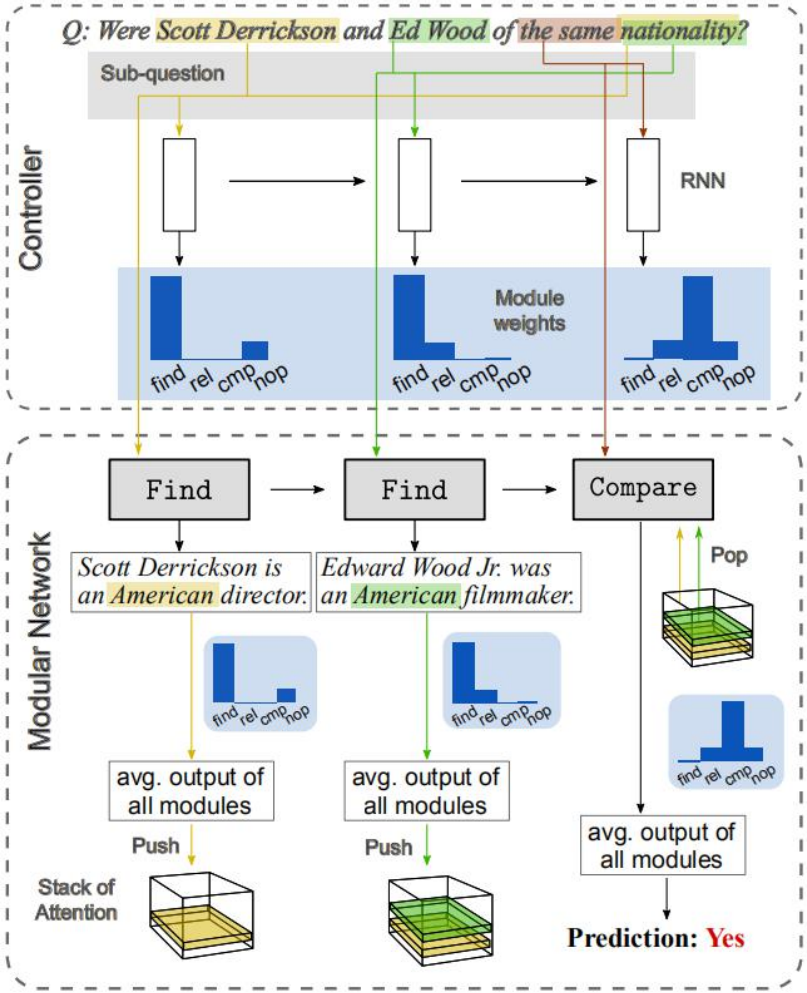


Controller：给定一个复杂问题，预测出需要的模组序列，并告诉每一步模组对应的子问题（以向量的形式）。

在第t步时，根据复杂问题的表示和第t-1步Controller的隐层向量来预测第t步需要哪个模组（Find、Relocate、Comp、NoOp）。

还会根据<复杂问题的表示，第t-1步控制器的隐层向量>以及上下文表示，对上下文的每一个token计算权重。按照权重将所有token表示相加得到向量Ct。

6.2.2 模型：模块



Find

功能：给定一个子问题，在上下文中寻找答案。

输入：子问题表示（Ct），原问题表示，上下文表示。

输出：一个在上下文上的attention map（入栈）。

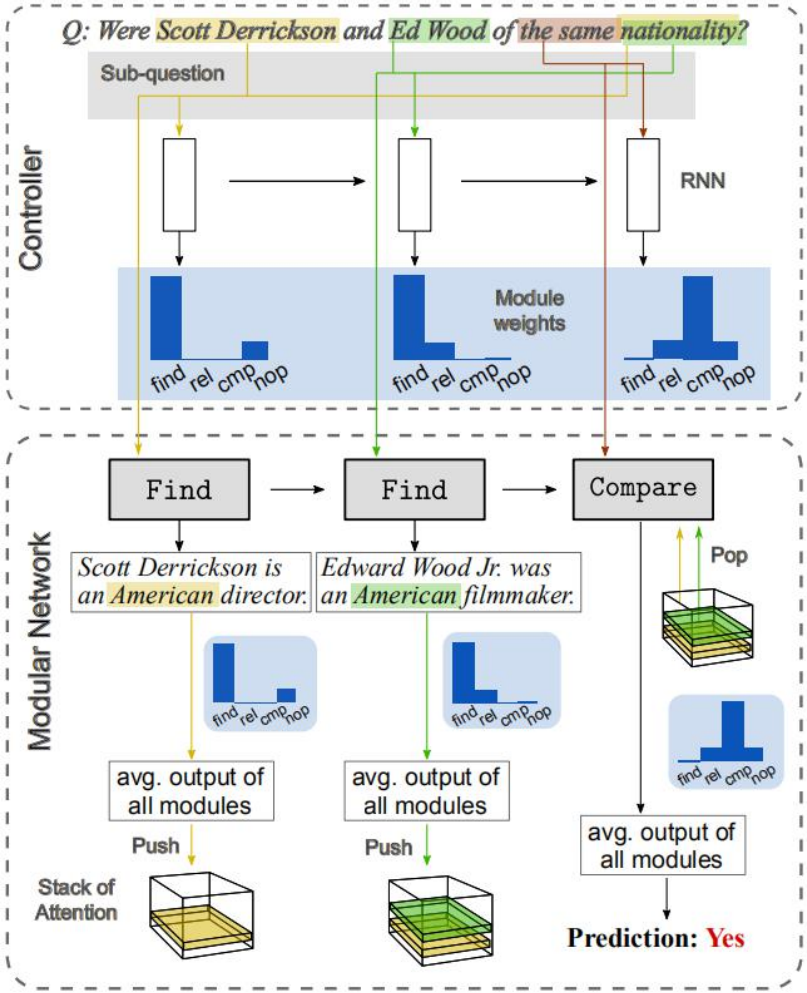
Relocate

功能：给定上一个子问题的答案（桥梁实体），化简原问题，并找到最终答案。

输入：attention map（蕴含桥梁实体）、原问题表示、上下文表示。

输出：一个在上下文上的attention map（入栈）。

6.2.2 模型：模块



Compare

功能：对比某对实体或事件的某一属性。

输入：两个attention map，上下文表示。

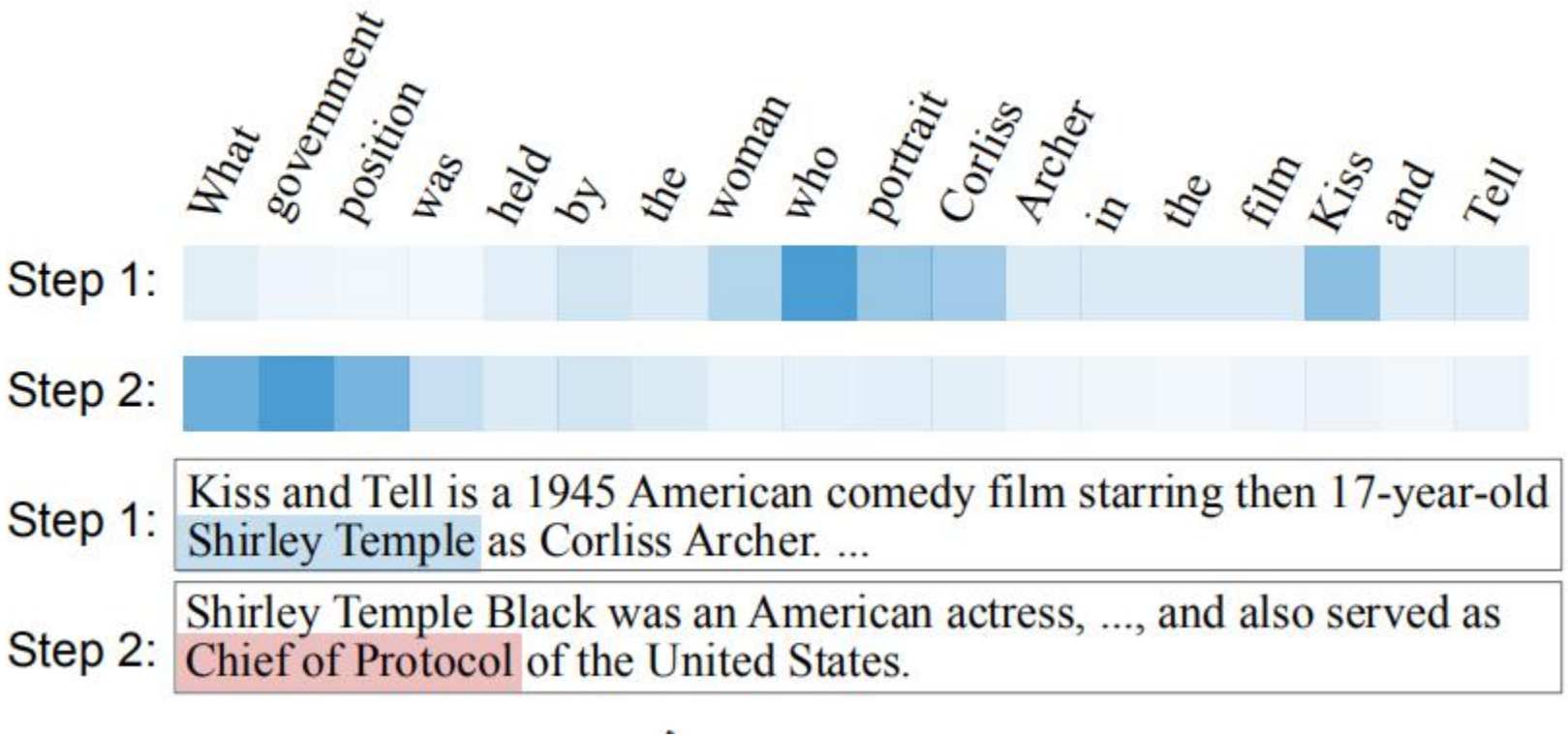
输出：一个向量（用于之后进行二分类）。

```
Relocate(Find())
Compare(Find(), Find())
```

（1）对于bridge类型的问题，最后剩余一张attention map。利用该attention map在上下文上做span预测得到最终答案。

（2）对于Compare类型问题，最后剩下一个向量，接着利用二分类器预测yes or no。

6.2.3 分析


What
government
position
was
held
by
the
woman
who
portrait
Corliss
Archer
in
the
film
Kiss
and
Tell
Step 1:
Step 2:
Step 1:
Kiss and Tell is a 1945 American comedy film starring then 17-year-old Shirley Temple as Corliss Archer. ...
Step 2:
Shirley Temple Black was an American actress, ..., and also served as Chief of Protocol of the United States.

Part 7 如何赋予PTM多跳能力

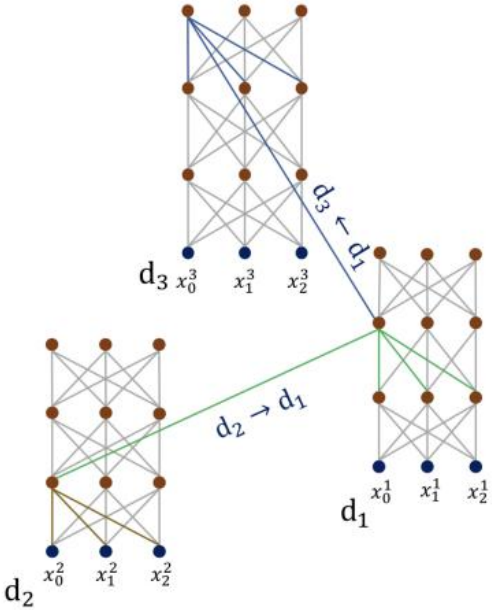
Transformer-XH (ICLR 2020)

TRANSFORMER-XH: MULTI-EVIDENCE REASONING WITH EXTRA HOP ATTENTION

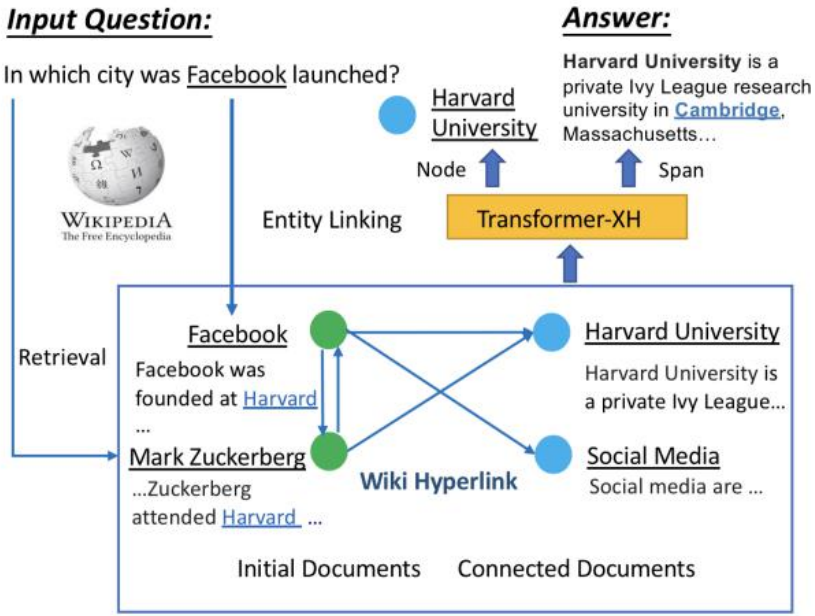
Chen Zhao*
University of Maryland, College Park
chenz@cs.umd.edu

Chenyan Xiong, Corby Rosset, Xia Song,
Paul Bennett, and Saurabh Tiwary
Microsoft AI & Research
cxiong, corosset, xiaso,
pauben, satiwary@microsoft.com

7.1 动机

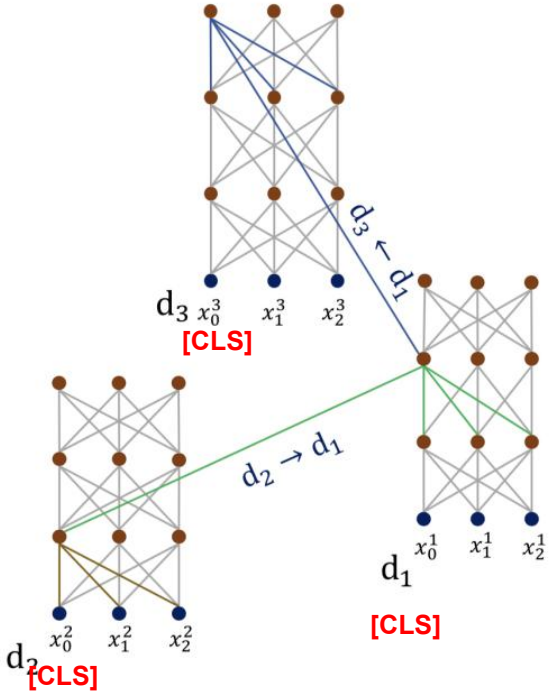


(a) Hop attention on the path $d_2 \rightarrow d_1 \rightarrow d_3$.



(b) Transformer-XH in Multi-hop QA

7.2 模型



(a) Hop attention on the path $d_2 \rightarrow d_1 \rightarrow d_3$.

step1：每个文档内的所有token按照vanilla transformer进行表示传递。

step2：相连接的文档中的[CLS] token通过self-attention进行表示传递。

7.3 应用于HotpotQA（fullwiki）

- 给定一个多跳问题q，结合已有的检索工具以及一些规则，得到相关文档集合{D1,D2,D3,...,Dn}。
- 在文档间建立连线：作者尝试了两种方法，（1）超链接文档连线（2）全连接
- 根据文档以及图结构通过transformer-XH得到所有文档中每个token的表示。
- 先预测那个文档包含答案（二分类，基于[CLS] token的表示）
- 上一步中概率最高的文档，继续进行span预测，得到最终答案。

7.4 实验结果：NO.5 in fullwiki

	Dev						Test					
	Ans		Supp		Joint		Ans		Supp		Joint	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Official Baseline (Yang et al., 2018)	23.9	32.9	5.1	40.9	47.2	40.8	24.0	32.9	3.9	37.7	1.9	16.2
DecompRC (Min et al., 2019a)	-	43.3	-	-	-	-	30.0	40.7	-	-	-	-
QFE (Nishida et al., 2019)	-	-	-	-	-	-	28.7	38.1	14.2	44.4	8.7	23.1
MUPPET (Feldman & El-Yaniv, 2019)	31.1	40.4	17.0	47.7	11.8	27.6	30.6	40.3	16.7	47.3	10.9	27.0
CogQA (Ding et al., 2019)	37.6	49.4	23.1	58.5	12.2	35.3	37.1	48.9	22.8	57.7	12.4	34.9
SR-MRS* (Nie et al., 2019)	46.5	58.8	39.9	71.5	26.6	49.2	45.3	57.3	38.7	70.8	25.1	47.6
CogQA (w. BERT IR) [Ours]	44.8	57.7	29.2	62.8	18.5	43.4	-	-	-	-	-	-
Transformer-XH	**54.0**	**66.2**	**41.7**	**72.1**	**27.7**	**52.9**	**51.6**	**64.1**	**40.9**	**71.4**	**26.1**	**51.3**

Table 1: Results (%) on HotpotQA FullWiki Setting. Dev results of previous methods are reported in their papers. Test results are from the leaderboard. Contemporary method is marked by *.

Part 8 在MhRC方面的数据增强工作

8 MhRC数据增强工作简单介绍

在低资源场景下，例如hotpotQA数据集，只给30%的训练集，如何还能够保证模型的预测效果？

一种非常自然的想法就是数据增强：

- 利用已有的样本数据基于一些特定的规则直接产生新的数据用于扩充训练集。
- QG：利用已有的样本数据，生成多跳问题以及对应的答案。以此来扩充训练集。

8 MhRC数据增强工作简单介绍

Labeled	Answer span		Support pred.		Joint	
Data#	EM	F1	EM	F1	EM	F1
QG + QA(i.e., the DFGN model)						
10%	.501	.633	.469	.764	.277	.509
20%	.551	.672	.500	.801	.317	.574
30%	.567	.697	.520	.815	.339	.600
40%	.569	.704	.521	.829	.340	.615
50%	.571	.713	.531	.833	.344	.619
60%	.586	.717	.533	.834	.346	.624
70%	.593	.729	.535	.839	.353	.626
80%	.606	.731	.540	.845	.356	.630
90%	.610	.741	.550	.853	.359	.632
100%	.614	.746	.558	.858	.360	.635
QA(i.e., the DFGN model)						
100%	.563	.697	.515	.816	.336	.598

只使用了**30%**的训练集，通过数据增强就能够达到原先使用**100%**训练集的效果。

Yu, J., Liu, W., Qiu, S., Su, Q., Wang, K., Quan, X., & Yin, J. (2020). Low-Resource Generation of Multi-hop Reasoning Questions. ACL.

8 MhRC数据增强工作

9.数据增强

序号	论文	发表会议	备注
1	Avoiding Reasoning Shortcuts: Adversarial Evaluation, Training, and Model Development for Multi-Hop QA	ACL 2019	non-Open 这篇文章也揭示了 HotpotQA 数据集有些问题不用推理也能回答，他们设置了攻击实验，发现在对抗数据集（通过在答案区间以及支撑文档的标题上进行短语级别的干扰得到，这样模型如果还是是用推理捷径的话将会得到多个可能的答案，从而影响模型的表现）上现有的SOTA模型表现都会下降很多，除此之外他们设计了一个控制单元来指导模型进行多跳推理。
2*	Low-Resource Generation of Multi-hop Reasoning Questions	ACL 2020	QG工作
3*	Logic-Guided Data Augmentation and Regularization for Consistent Question Answering	ACL 2020	讨论核心点是对比问题的数据增强，基于对称一致性和传递一致性来增强训练样本
4*	Generating Multi-hop Reasoning Questions to Improve Machine Reading Comprehension	WWW 2020	QG工作
5*	Asking Complex Questions with Multi-hop Answer-focused Reasoning	arXiv 2020	QG工作

(*代表仅属于本分类下的工作)

https://github.com/krystalan/Multi-hopRC#9数据增强

Part 9 模型真的学会了多跳推理么？

https://github.com/krystalan/Multi-hopRC#10在本质方面的探索

Understanding Dataset Design Choices for Multi-hop Reasoning

Jifan Chen and **Greg Durrett**
The University of Texas at Austin
{jfchen,gdurrett}@cs.utexas.edu

基于HotpotQA（抽取式）以及Wikihop（多选式）多跳阅读理解数据集设计了很多有意思的实验，并得出了很多有趣的结论。

9.1 单跳QA模型也能答对多跳问题

Task：判断一个句子是否包含多跳问题的答案（note：这个任务比直接预测答案要简单）

分析：如果我们每次只给模型输入一个句子（模型不具备推理能力），就让模型去判断这句话是否包含多跳问题的答案。按理来说模型的预测结果应该非常低下。多跳问题的答案是要在多个段落的支撑句上推理才可以得到的。

于是作者使用一个传统MRC模型（BiDAF）每次只告诉模型问题和一句话，让模型来预测该句话包含答案的概率

Method	Random	Factored	Factored BiDAF
WikiHop	6.5	60.9	66.1
HotpotQA	5.4	45.4	57.2
SQuAD	22.1	70.0	88.0

9.2 wikihop不看文章也能答对

刚才的实验里面，我们好歹还给了模型一句话，去让模型预测答案是否存在于该句话。结果发现，毫无推理能力的模型也能取得很好的效果（超过50%）。

于是作者设计了一个更绝的实验，由于Wikihop是一个多选类型的多跳阅读理解数据集，所以直接告诉模型问题和候选答案集合，看看模型还能不能保持住较好的效果。

使用GRU分别编码问题q和每一个候选答案ci，之后将候选答案的表示与编码问题的表示进行双线性点积运算得到该候选答案为正确答案的概率。

NoContext	Coref-GRU	MHQA-GRN	Entity-GCN
59.70	56.00	62.80	64.80
	NAACL 2018	arXiv 2018	EMNLP 2018

9.3 抽取式比多选式数据集更能考验模型的真实多跳能力：Span-based Vs. Multiple-choice

已知HotpotQA是一个抽取式数据集；Wikihop是一个多选式数据集。

将HotpotQA转换成多选式数据集：从HotpotQA的distractor documents中随机选了9个实体，与gold truth answer 共同组成一个大小为10的集合，作为多选式HotpotQA的答案集合。

将Wikihop转换成抽取式数据集：将每一个问题对应的文档拼接起来，第一次出现答案提及(mention)的地方标记成抽取式Wikihop的答案。

Dataset	HotpotQA-MC	WikiHop-MC
Metric	Accuracy	Accuracy
NoContext	68.01	59.70
MC-BiDAF++	70.01	61.32
MC-MemNet	68.75	61.80
Span2MC-BiDAF++	76.01	59.85

Dataset	HotpotQA-Span		WikiHop-Span	
Metric	EM	F1	EM	F1
BiDAF++ (Yang+ 18)	42.79	56.19	–	–
Span-BiDAF++	42.45	56.46	24.23	46.13
Span-MemNet	18.75	26.11	13.54	19.23

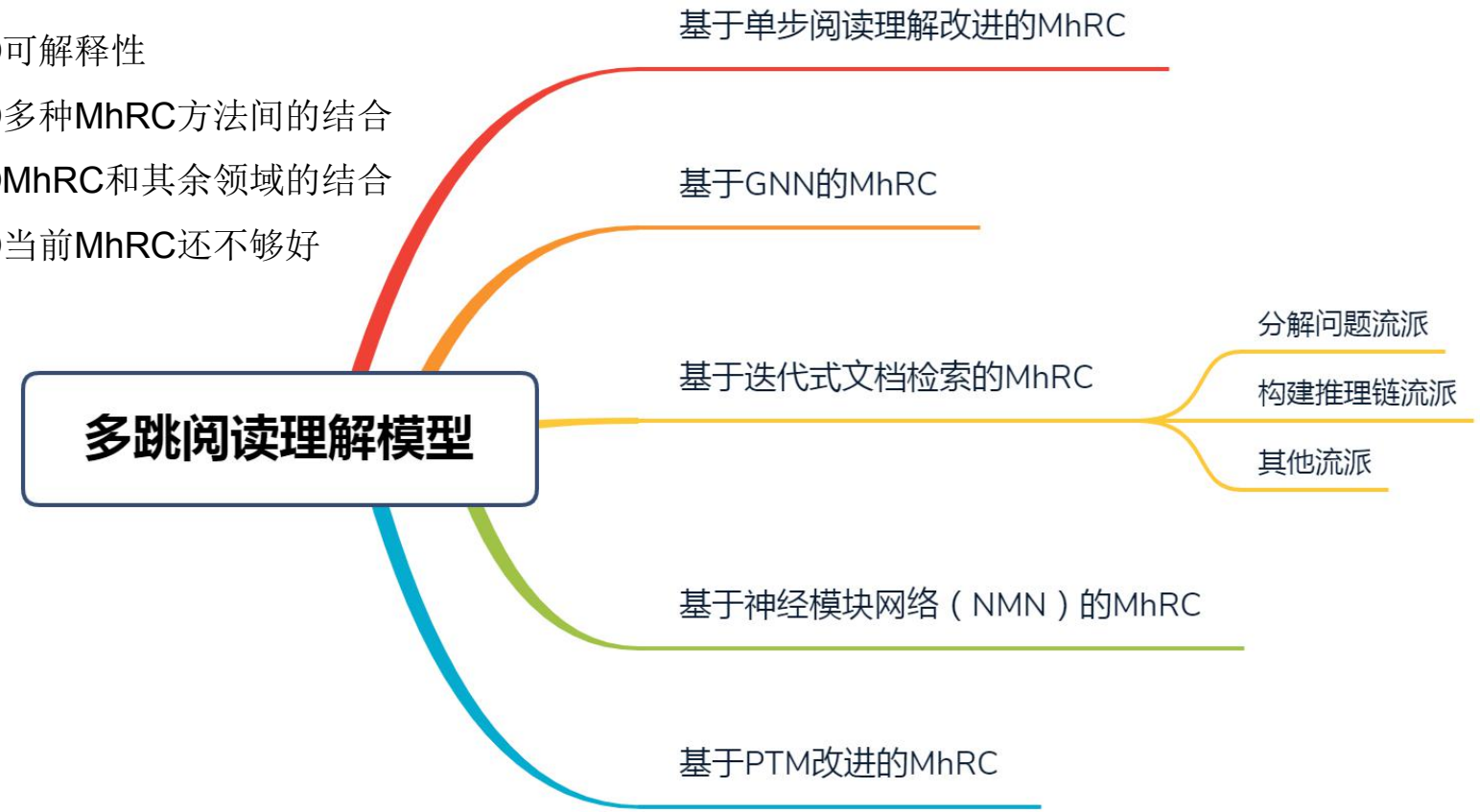
Part 10 总结

①可解释性

②多种MhRC方法间的结合

③MhRC和其余领域的结合

④当前MhRC还不够好



感谢聆听