EMNLP19 Summarization

Xiachong Feng

目录

1 BottleSum: Unsupervised and Self-supervised Sentence Summarization using the Information Bottleneck Principle

★★★★★ 可能由于 Unsupervised，和 Self-Supervised 兴起，信息论相关的摘要论文开始出现，让人眼前一亮，结合了深度学习和传统信息论的优势，具有很强的解释性，而且别的领域有很强的借鉴意义，首推 ACL19[Peyrard, 2019] 的和 EMNLP19[West et al., 2019]。一个详细的全文解读：BottleSum—文本摘要论文系列解读。利用 Information Bottleneck Principle 来完成句子摘要任务。Information Bottleneck Principle(IB) 的大致意思是说：根据 X 得到一个压缩后的 $\tilde{X}$，使得根据 $\tilde{X}$ 可以更好地预测和 X 相关的 Y。在摘要里，可以理解为：根据原文，得到一个摘要，根据这个摘要可以更好的预测和原文相关的某一个东西（如果有 golden，直接是 golen 就行，如果没有，下面将讲办法）。这样可以形式化定义为：$p(\tilde{S}|S) = I(\tilde{S};S) - \beta I(\tilde{S};Y)$，其中 I 表示了互信息，例如 $I(\tilde{S};S)$ 表示了 $\tilde{S}$ 包含了多少 S 中的信息。这个式子表达了一种对抗的想法，一方我们

希望根据原文生成的摘要 $\tilde{S}$ 尽可能的丢弃原文的信息，另一方面，我们希望 $\tilde{S}$ 尽可能的包含多的 Y 中的信息。论文中第一项被称为 pruning term，第二项被称为 relevance term。那么 Y 是什么东西？论文从无监督的角度出发，也就是并没有 golden，而是利用当前句子的下一句话来表示 Y，像比较流行的预训练的 MLM 和 Next sentence prediction。希望对当前句子做摘要，得到的结果可以预测出下一句。论文中将这种方法和 Auto-encoder 进行了对比，说明了其好处：auto-encoder 由于有一个重构损失，为了让这个损失降低，所以 model 就倾向于记录全部内容，所以包含了一些无关紧要的信息。而使用本文方法，目的并不是重构原文，所以没有这个问题。论文基于上述理论，对摘要问题重新建模，把上面的公式用 PMI（点互信息）进行了转换得到了一个目标方程：$-\log p(\tilde{s}) - \beta_1 p(s_{next}|\tilde{s}) p(\tilde{s}) \log p(s_{next}|\tilde{s})$，越小越好。其中 $p(\tilde{s})$ 和 $p(s_{next}|\tilde{s})$ 可以用一个预训练的语言模型得到，论文中提到了这个语言模型越强，效果越好。那么这种方法和之前的 DL-based 的模型不同，不需要在整个大的数据集上学习一个映射，也就是拟合数据分布。而是对于每一条数据，都可以按照上述公式直接找到一个对应的压缩。具体算法如1。

Algorithm 1 *BottleSum*^{Ex} method

Require: sentence s and context s_{next}

$C \leftarrow \{s\}$ ▷ set of summary candidates
for l in $length(s)...1$ **do**
  $C_l \leftarrow \{s' \in C | len(s') = l\}$
  sort C_l descending by $p(s_{next}|s')$
  for s' in $C_l[1:k]$ **do**
    $l' \leftarrow length(s')$
    for j in $1...m$ **do**
      for i in $1...(l'-j)$ **do**
        $s'' \leftarrow s'[1:i-1] \circ s'[i+j:l']$
        if $p(s'') > p(s')$ **then**
          $C \leftarrow C + \{s''\}$
return $\arg\max_{s' \in C} p(s_{next}|s')$

图 1: BottleSum 算法

首先候选摘要集合中只有 s，然后从最长长度 $length(s)$ 到 1（从最长长度到最短长度），开始迭代，每次选取该长度的候选摘要，然后根据概率排序一下，选择 top-k 个，k 是一个超参。然后对于每一个候选，开始删除其中的词语，最长可以删 m 个，m 是另一个超参。如果删除以后的 s'' 比 s'（根据预训练语言模型得到）增加，那就列入候选集合。最后从候选集合中选取一个可以生成概率 $p(s_{next}|s')$ 最高的。

在抽取完以后已经可以直接作为输出，除以以外，本篇论文还利用 GPT-2 和抽取出来的摘要构建了 pair 对训练一个生成式 model。

2 NCLS: Neural Cross-Lingual Summarization

★★★★★ 来自于中科院自动化所模式识别国家重点实验室。针对跨语言摘要（Cross-Lingual Summarization）提出了新的数据集（准确的说是第一份数据集，ACL19[Duan et al., 2019] 利用

Gigaword 只标注了开发集和测试集）。基于该数据集完成端到端的训练，并利用单语言摘要（MS，Monolingual Summarization）和机器翻译（MT，Machine Translation）作为辅助任务进行多任务学习。针对的问题还是由于之前没有直接的数据，所以需要基于”摘要“和”翻译“的 pipeline 完成，这会导致错误传播。有了新的数据集以后，就可以来完成端到端的训练了。某种意义上类似于机器翻译，但是机器翻译是一一对应，相当于压缩比是 1:1，但是跨语言摘要还需要完成重要内容提取，然后在目标端重新组织语言。

跨语言摘要简单来说就是输入源语言文档，输出目标语言摘要。源语言与目标语言不同。之前的方法是基于 pipeline 的方法。先摘要后翻译：这种方法首先需要在源语言端有摘要的数据来训练一个源语言的摘要模型，对于 Low-resource 来说，就比较麻烦。或者可以使用一些类似 TextRank 之类的无监督方法实现。如果摘要这一步就比较差，再加上翻译效果不好，那最后的结果可想而知。另一种是先翻译再摘要：翻译主要有 domain 的问题，训练翻译模型的 domain 和摘要的 domain 不同，那第一步翻译结果就不行，然后再摘要最后效果也会一般。所以整体来说，pipeline 方法会有严重的错误级联问题，一直被诟病。[Ouyang et al., 2019] 觉得先摘要的方法对于 Low-resource 来说行不通，所以更加 prefer 先翻译的方法。那么解决 pipeline 方法的根本方式就是有一个端到端的数据集，这篇论文就干了这样一件事情。

数据集由英语单语数据集 CNNDM 和多模态摘要数据集 MSMO[Zhu et al., 2018] 和中文单语数据集 LCSTS[Hu et al., 2015] 得到。其中 CNNDM 和 MSMO 统称 ENSUM，属于新闻领域，LCSTS 数据来源于新浪微博。基于 ENSUM 来构建 En2Zh，基于 LCSTS 来构建 Zh2EN。采用的方法叫做 Round-trip translation strategy，round-trip 翻译就是首先把一个句子翻译到另外一种语言（forward translation），然后再翻译回来（back translation）。基本过程如图2。

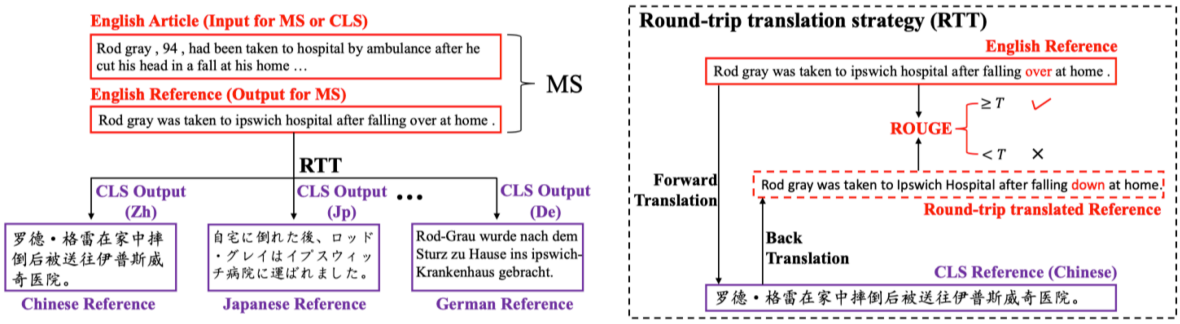

图 2: Round-trip translation

拿英语来说，原有英语摘要对为 (D_{en}, S_{en})，将 S_{en} 中的每一句话进行 round-trip 翻译，如果翻译之后的结果与原来的句子 ROUGE F1 值高于设定的值，那么就选取，否则就剔除。对于整个 S_{en}) 来说，如果三分之二的句子都被选取了，那么这个跨语言摘要对就会被采用。(对于中文来说也一样，计算 ROUGE 用 Char，实际写代码我用 BERT 词表把字符替换成了数字进行计算)。最开始看到这篇论文，我以为找到了某一种资源直接可以得到跨语言的摘要对，但实际还是通过一种自动化的方式，加了一些限制。不过也挺好，有了标准数据集，可以有 baseline 了，之前 Cross-Lingual 的论文都是各搞各的，现在即使做无监督，也可以有个比的。（不过目前见到的 cross-lingual 都是中英。）最后得到 370759 En2Zh 对，1699713 Zh2En 对。

Baseline 方法包括了 Early Translation (ETran) 和 Late Translation (LTran)。还有基于端到端的 Transformer-based NCLS models (TNCLS)。之后利用 MS 和 MT 来进行多任务学习 CLS+MS，CLS+MT。对于 CLS+MS 来说，输入是一个文档，输出是不同语言的摘要。对于 CLS+MT 来说，输入也不同，一个是摘要数据，一个是翻译数据。如图3。

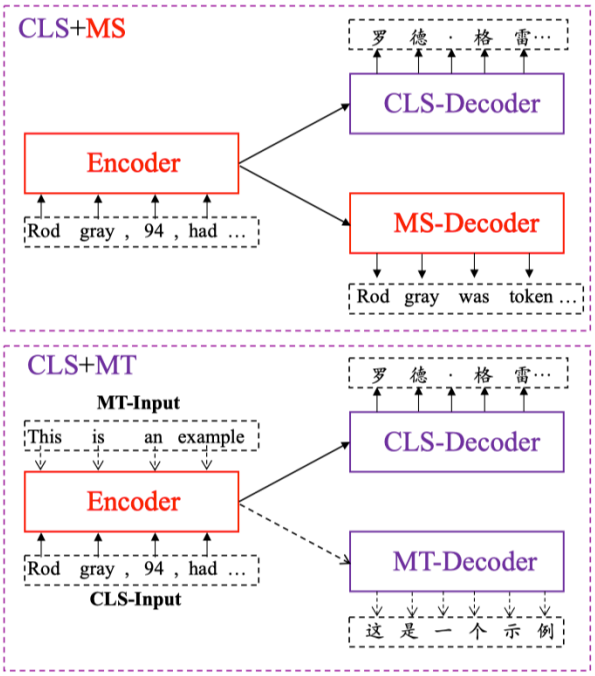

图 3: Multi-task NCLS

1. **TETran** 首先利用 LDC 训练基于 Transformer 的机器翻译模型，先翻译，再利用 LexRank 摘要。利用无监督摘要方法的原因是：在翻译以后，缺少这种单语语料的训练数据。
2. **TLTran** 利用基于 Transformer 的模型，分别训练翻译模型和单语摘要模型。串联进行。
3. **GETran** 利用 Google Translator。
4. **GLTran** 利用 Google Translator。
5. **TNCLS** 基于 Transformer 的 end2end 模型。
6. **CLS+MS** 结合 MS 多任务。
7. **CLS+MT** 结合 MT 多任务。

下面看一下实验结果4，还是有很多有意义的结果。

1. GETran、GLTran 要比 TETran、TLTran 效果好，说明在 Pipeline 方法中，翻译效果比较重要。
2. 只看 TNCLS 的 En2Zh 数据集，发现 w-c 的分割方式要比 w-w 好得多。也就是中文端用 char 靠谱。论文中的解释是：中文端基于 char 分割，可以显著降低词表大小，decode 时候生成更少 unk。
3. 在 Zh2En 上，sw-sw 效果好，论文解释是可以降低词表，减少 UNK，c-w 效果不如 sw-sw，但是论文没有实验 c-sw，不知道效果如何？
4. En2Zh 和 En2Zh* 结果差不多，原因是 En2Zh 是新闻领域的，现有 MT 在新闻领域效果不错。Zh2En 和 Zh2En* 效果相差很多，因为 Zh2En 是微博数据，MT 系统翻译效果差。人工校正就比较多。

Model	Unit	En2ZhSum	En2ZhSum*	Zh2EnSum	Zh2EnSum*
		RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)
TETran	–	**26.12-10.59-23.21**	**26.15-10.60-23.24**	**22.81- 7.17-18.55**	**23.09- 7.33-18.74**
GETran	–	**28.17-11.38-25.75**	**28.19-11.40-25.77**	**24.03- 8.91-19.92**	**24.34- 9.14-20.13**
TLTran	c-c	–	–	32.85-15.34-29.21	33.01-15.43-29.32
	w-w	**30.20-12.20-27.02**	**30.22-12.20-27.04**	31.11-13.23-27.55	31.38-13.42-27.69
	sw-sw	–	–	**33.64-15.58-29.74**	**33.92-15.81-29.86**
GLTran	c-c	–	–	34.44-15.71-30.13	34.58-16.01-30.25
	w-w	**32.15-13.84-29.42**	**32.17-13.85-29.43**	32.42-15.19-28.75	32.52-15.39-28.88
	sw-sw	–	–	**35.28-16.59-31.08**	**35.45-16.86-31.28**
TNCLS	c-w	–	–	36.36-19.74-32.66	35.82-19.04-32.06
	w-c	**36.83-18.76-33.22**	**36.82-18.72-33.20**	–	–
	w-w	33.09-14.85-29.82	33.10-14.83-29.82	38.54-22.34-35.05	37.70-21.15-34.05
	sw-sw	–	–	**39.80-23.15-36.11**	**38.85-21.93-35.05**

图 4: Baseline 实验结果，* 代表了人工校正以后的结果

5. 基于 Transformer 的 End2End 方法要比 pipeline 方法好。

下面是 Multitask 的结果5。

Model	En2ZhSum	En2ZhSum*	Zh2EnSum	Zh2EnSum*
	RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)	RG1-RG2-RGL(↑)
TNCLS	**36.83-18.76-33.22**	**36.82-18.72-33.20**	**39.80-23.15-36.11**	**38.85-21.93-35.05**
CLS+MS	38.23-20.21-34.76	38.25-20.20-34.76	41.08-23.67-37.19	**40.34-22.65-36.39**
CLS+MT	**40.24-22.36-36.61**	**40.23-22.32-36.59**	**41.09-23.70-37.17**	40.25-22.58-36.21

图 5: 多任务学习，En2Zh(w-c)、Zh2En(sw-sw)

1. 多任务学习对于 baseline 都有帮助。
2. 在 En2Zh 上，MT 帮助更显著，原因可能是：（1）MT 数据集大；（2）MT 数据集是新闻领域的。但是在 Zh2En 上，效果基本一致。

另外关于一些实验细节的说明，在 En2Zh 数据集上，En 没有采用 subword 分割，因为这样会导致输入太长，那当然 char 就更不用说了，没有意义。所以在 En2Zh 上输入端英文只用 word 分割。在 En2Zh 数据集上可以有两种方式，w-c、w-w，分别代表了英语输入端用 word，输出端中文用 char；英语输入端用 word，输出端中文用 word。对于 En2Zh 数据集输入截断为 200 个词语，英语摘要截断为 120 个词语，中文截断为 150 个字。

相比于单语言摘要，跨语言摘要的难点在哪里呢？反正都有了端到端的数据集了，直接训练不就好了？所以是不是还是应该从输入输出语言不同下手？能不能同这两个反向的互相帮助？再仔细想想 MT 和 MS 的作用？基于 cross-lingual pretrian 来做呢？

3 Reading Like HER: Human Reading Inspired Extractive Summarization

★★★☆☆ 在 CNNDM 上做抽取式摘要，以人类的阅读过程作为动机。人类在阅读一篇文章的时候分为三个步骤：pre-reading，reading 和 post-reading。pre-reding 为粗读过程，获得粗粒度的信息，留有一个大致的印象，reading 会根据 pre-reading 获得信息来寻找一些细致的信息，post-reading 属于查缺补漏阶段。论文将三个过程合并为两个过程 rouge-reading 和 careful-reading，粗读和精读。整个抽取式摘要过程建模为一个 Contextual-bandit 过程（有一个上下文，根据这个上下文，采取一个 action，然后得到一个奖励）。对应到抽取式摘要里面就是根据得到的文档表示的全局特征和局部特征，选取句子，然后根据得到的句子集合和标准 golden 比较得到奖励。我第一遍读的时候觉得没毛病，但是第二遍读的时候觉得好像去除这个故事，就是一个正常的建模为 *Contextual-bandit* 的过程，好像和精读粗读没啥关系……。整个过程如图。

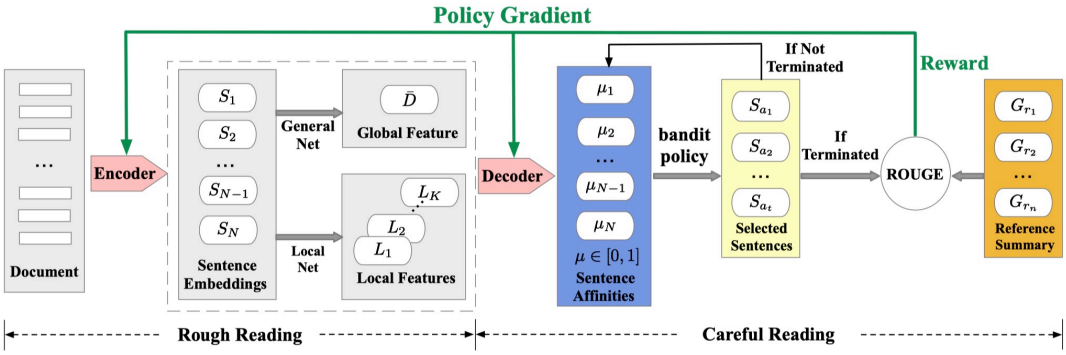

图 6: HER model

Rouge-reading 部分用 Bi-LSTM 来层次化编码得到句子表示 $S_1, S_2, ..., S_N$，然后平均一下得到全局表示 $\overline{D}$，局部表示 L 用 k 不同窗口的个 max-over-time pooling 得到。然后对于每一个句子，与全局表示和局部表示相结合可以有一个得分。根据这个得分，采用强化学习中的探索与利用思想，利用 ϵ-greedy 来进行选择，引入一定的随机性。同时提出了一个 Termination Mechanism，当触发这个机制的时候，就会停止选择。 Done $\sim$ Bernoulli $\left(\min\left(\frac{\mu_{\min}}{\mu_{\max}}, 1-\mu_{\max}\right)\right)$ 当剩下的得分非常相近，或者得分很低的时候会停止。训练的时候用 REINFORCE 训练。

4 Neural Extractive Text Summarization with Syntactic Compression

★★★★☆ 本篇工作来自于得克萨斯大学奥斯汀分校，基于句子抽取（Extraction）和句子压缩（Compression）来完成单文档摘要。首先从文档中选取一些句子，然后对于选中的每一个句子，从一个压缩规则集合中选择一个规则对该句进行压缩。这些规则是从句法成分分析（syntactic constituency parses）取得的。因为是压缩是显示的规则，所以对于最后的生成结果，可解释性也比较强。

压缩规则：同位名词短语; 关系从句和状语从句; 名词短语中的形容词短语和状语短语 (见图7); 作为名词短语一部分的动名词短语 (参见7); 某些配置中的介词短语，比如周一; 括号和其他括号内的内容; 当然这些规则覆盖面肯定不是非常全，对于特定的领域和任务，可以制定特定的规则。一个具体的压缩示例7。

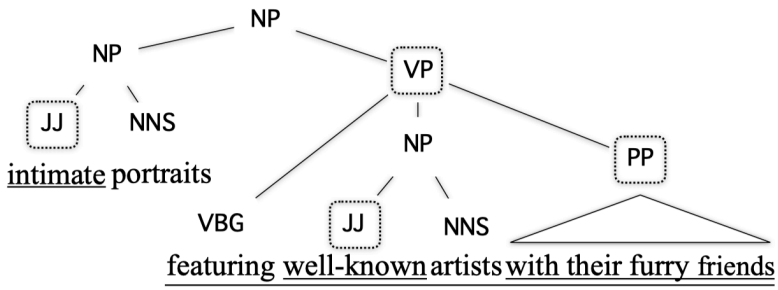

图 7: 文本压缩示例

句子抽取：文档中的句子用 Bi-LSTM 表示，CNN 得到句子整体表示，过 DOC LSTM 得到表示，DOC CNN 得到文档整体，表示然后一个 DOC LSTM decoder 来选择句子。在每一个 deocde step，拿到 doc 表示、上一时刻选取句子表示、上一时刻隐层表示，得到新的隐层表示，利用新的隐层和每一个句子表示得到 logit，选择最大的一个。被选取的句子之后不能再被选取，可以通过强制该句 logit 为 0 来实现。如图8。

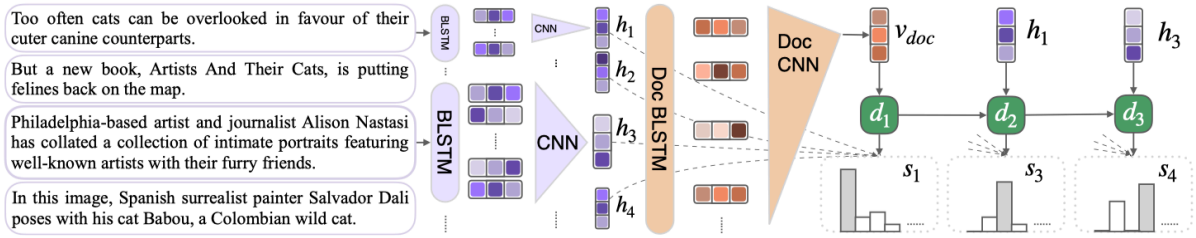

图 8: 句子抽取模型

句子压缩：根据规则可以得到可能压缩的 span，对于每一个可能压缩的词语或者短语，模型判断是否删除。标准的 Label 由规则给出。如图9。

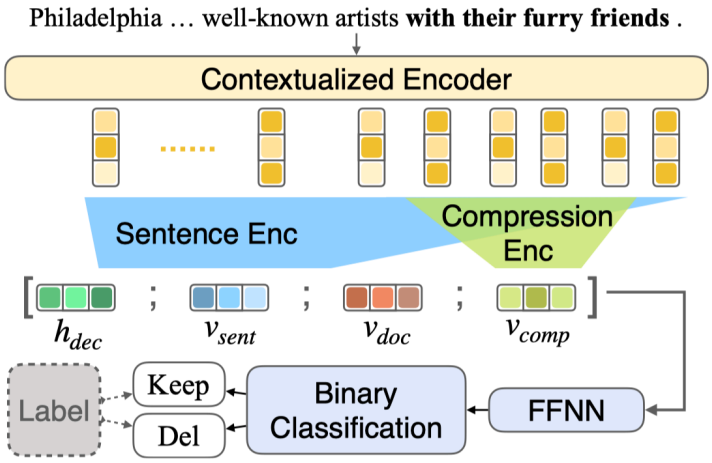

图 9: 句子压缩模型

5 Concept Pointer Network for Abstractive Summarization

★★★★☆ 生成式摘要模型 pointer-generator 往往从原文中 copy，很少能像人一样可以完成高度的概括。本篇论文引入了外部知识库，叫做 Microsoft Concept Graph，是一个 is-a 的个概念库，对于一个词语，可以有一系列的候选概念词语。通过这方式来引入高层概念，一个明显优势就是可以缓解 oov。整个 model 如图10。

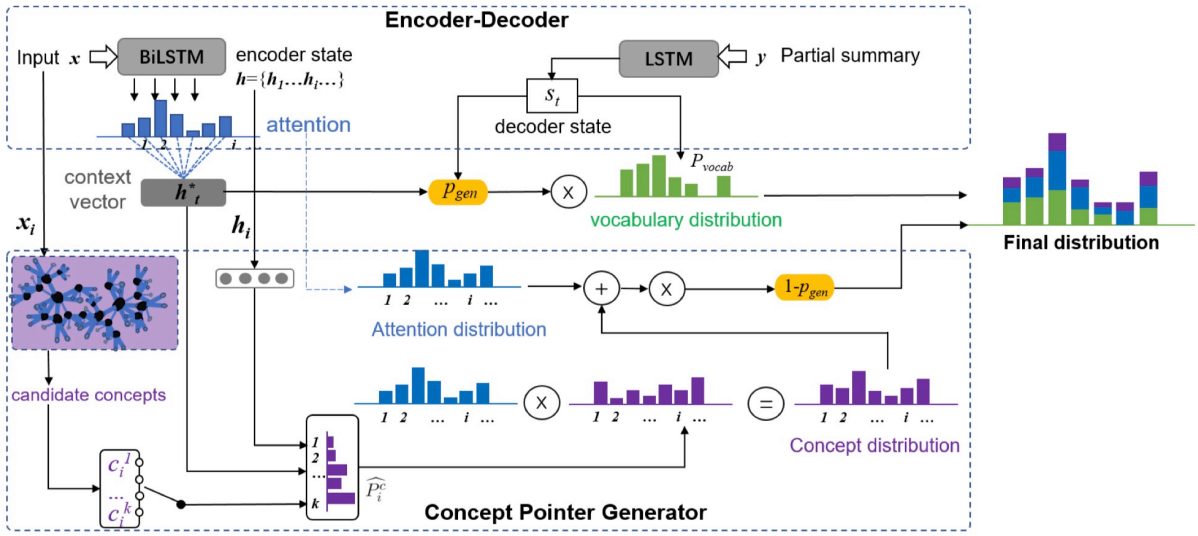

图 10: Concept Pointer Network

模型一共分为两个两个部分，一个是 encoder-decoder 部分，一个是 concept pointer generator 部分。encoder-decoder 就是 pointer-generator，没有变化。concept 部分，对于输入的每一个词语 x_i，可以从外部知识中找到一系列对应的 concept：$C_i = \left\{c_i^1, c_i^2, \cdots, c_i^k\right\}$。同时这个外部知识可以提供一个概率：$P\left(C|x_i\right) = \left\{p\left(c_i^1\right), p\left(c_i^2\right), \cdots, p\left(c_i^k\right)\right\}$ 代表了是某一个 concept 的概率。然后根据 context vector h_t^* 和 h_i 可以得到一个更新之后的分布：$\beta_i^j = \text{softmax}\left(\boldsymbol{W}_{h'}\left[\boldsymbol{h}_i, \boldsymbol{h}_t^*, \boldsymbol{c}_i^j\right]\right)$，最后得到 $P_{i,j}^c = p\left(c_i^j\right) + \gamma\beta_i^j$。相当于对于每一个输入词语，有一个候选列表，这个候选列表上有一个概率分布。最后选的时候一种方法按照 max 来选，一种按照 random 来选。通过把这个分布整合到 concept pointer generator 中引入 concept。

6 Countering the Effects of Lead Bias in News Summarization via Multi-Stage Training and Auxiliary Losses

★★★☆☆ 来自于加拿大麦吉尔大学的工作。虽然 lead bias 是一种非常显著的特征，尤其在新闻领域，但是并非所有情况都是这样的，过度的使用这种特征会导致一些重要内容并非集中在开始的样例效果差。如果能平衡 lead bias 和 semantic content selection 能力，就可以很好地应对两种情况。这篇论文提出了两种办法来增强内容选择的能力，一种是两步训练（Multi-Stage Training），首先在打乱句子顺序的数据集上预训练，然后在原始数据集上的训练。先在乱序的数据集上训练，可以使得模型最开始有一定的内容选择能力，而不会过分的依赖于选择 lead 位置。另一种是用一种句子级别的辅助 loss（Auxiliary Losses），在选择句子的时候，考虑预测的得分和真实的得分，会使得模型能够知道，即使在后面，也会有高分句子出现，这个方法有点类似于 [Zhou et al., 2018]，利用文本句子和 golden 来计算一个 rouge 得分，归一化为一个分布，模型会预测一个分布和这个

分布算 loss。最后效果发现 pretrain 效果一般，没有太大提升，甚至会查。辅助 loss 效果好。应该认真探索一下 pretrain 的方法，怎么好使。

7 An Entity-Driven Framework for Abstractive Summarization

★★★☆☆ 乍一看以为是 NER 那种人名，地名，机构名的实体，但实际上是共指的实体。给一篇文章，首先利用斯坦福的 coreference resolution 系统对文章进行一下共指消解，然后把相同的 mention 作为一个 cluster。那这样搞可能有很多的 cluster，最后根据两个规则选取（1）实体在前三句出现过，（2）top-k 个最多词语的 cluster。然后根据文档和 cluster 来选取重要句子。

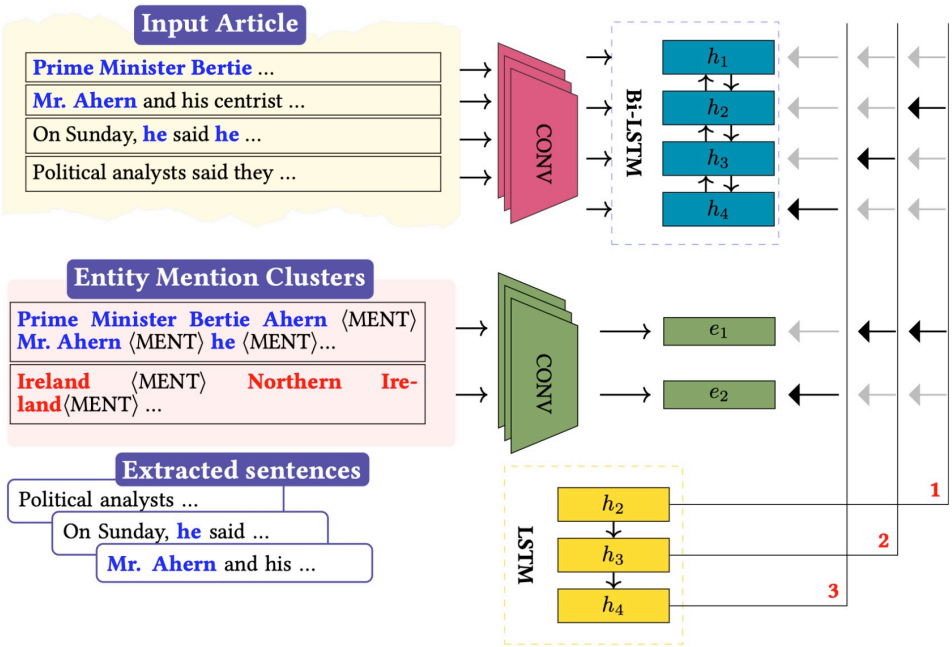

图 11: Entity-aware content selector

核心就是一个 pointer-network，每一个部分有 nn 来得到表示，特殊的是 cluster 部分，把所有的 mention 拼成字符串得到。根据句子的 attention 和 entity 的 attention 来选取句子，抽取 label 通过和 golden 比较 rouge 得到。在选取得到以后，训练一个 seq2seq 来生成最终的摘要。可以 mle 也可以强化学习。这篇论文还额外提出了其他的奖励，连贯性和一些语言学相关的指标。

8 Attention Optimization for Abstractive Document Summarization

★★★☆☆ 核心在于提出了一个 Attention Refinement Unit (ARU) 组件和两个和 Attention 有关的 loss 来优化 attention，一方面，在 decoder 的每一步，让 attention 更倾向于 attned 到重要的少数部位，另一方面，不要重复 attened 同一个位置（和 coverage 思想相似）。首先 ARU 是一个门，根据 decoder 当前状态 s_t 和当前 attention 分布 a_t 可以得到：

$$r_t = \sigma\left(W_s^r s_t + W_a^r a_t + b_r\right) \tag{1}$$

根据该门可以得到更新以后的 attention 分布，然后引入 local variance loss 来使得 attention 分布更加的 sharp，也就是类似于 hard attention，尽可能使其 attned 到局部，不要所有都注意，会分散注意力。那么这个分布的方差越大，分布越不稳定，也就越 sharp：

$$\begin{aligned} \operatorname{var}\left(a_t^r\right) &= \frac{1}{|D|}\sum_{i=1}^{|D|}\left(a_{ti}^r - \hat{a}_t^r\right)^2 \\ \mathcal{L}_L &= \frac{1}{T}\sum_{t}^{T}\frac{1}{\operatorname{var}\left(a_t^r\right)+\epsilon} \end{aligned} \tag{2}$$

其中 $\hat{a}$ 实际操作是用的中位数，前人工作证明了更有效。如图12。除此以外还提出了一个 Global

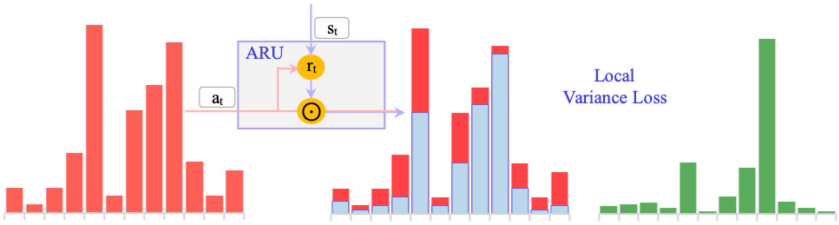

图 12: Attention Refinement Unit+Local Variance Loss

Variance Loss 直接计算，作用和 coverage 一致，coverage 将之前 $t-1$ 步骤的 attention 累计和当前 t 的 attention 进行比较。Global Variance Loss 直接计算 decoder 所有 t 步的一个比较。直觉是，如果 attention 不重复 attned 一个地方，那么一个位置 atten 值的求和和其 max 值应该基本差不多：

$$\begin{aligned} g_i &= \sum_t\left(a_{ti}^r\right) - \max_t\left(a_{ti}^r\right) \\ \mathcal{L}_G &= \frac{1}{|D|}\sum_{i=1}^{|D|}\left(g_i - \hat{g}\right)^2 \end{aligned} \tag{3}$$

g_i 代表了一个位置的 attn 值求和和最大值的差值。这个分布是越平均越好。

9 Text Summarization with Pretrained Encoders

★★★☆☆ 来自于爱丁堡大学 Yang Liu 和 Mirella Lapata，Yang Liu 已经出过一个基于 Pretrain 的 Summarization 论文 [Liu, 2019]，当时在抽取式摘要上达到了 SOTA。这篇论文的抽取式摘要部分看起来就是 BERTSUM。在生成式摘要部分，由于 Encoder 部分预训练过，Decoder 部分没有预训练，所以有两个优化器来分别优化 Encoder 和 Decoder。

10 Abstract Text Summarization: A Low Resource Challenge

★★☆☆☆ 利用了机器翻译中的 back-translation 思想来解决。完成 German 的摘要任务。用已有的摘要数据训练一个从摘要生成原文的 model，然后给一些新的 German 数据，利用训练好的 model 可以生成对应原文，把这些数据和原来真实的数据合起来训练摘要 model。如图6。

参考文献

Xiangyu Duan, Mingming Yin, Min Zhang, Boxing Chen, and Weihua Luo. Zero-shot cross-lingual abstractive sentence summarization through teaching generation and attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3162–3172, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1305.

Baotian Hu, Qingcai Chen, and Fangze Zhu. Lcsts: A large scale chinese short text summarization dataset. *arXiv preprint arXiv:1506.05865*, 2015.

Yang Liu. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*, 2019.

Jessica Ouyang, Boya Song, and Kathy McKeown. A robust abstractive system for cross-lingual summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2025–2031, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1204. URL https://www.aclweb.org/anthology/N19-1204.

Maxime Peyrard. A simple theoretical model of importance for summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1101. URL https://www.aclweb.org/anthology/P19-1101.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. BottleSum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3750–3759, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1389. URL https://www.aclweb.org/anthology/D19-1389.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. Neural document summarization by jointly learning to score and select sentences. *arXiv preprint arXiv:1807.02305*, 2018.

Junnan Zhu, Haoran Li, Tianshang Liu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. Msmo: Multimodal summarization with multimodal output. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4154–4164, 2018.