NLP Project 2

Story Generation

xxx,xxxxxxxx 朱仁博,xxxxxxx

xxxxx@pku.edu.cn zhurenbo@pku.edu.cn

2021 年 4 月 1 日

目录

1 问题描述

1.1 任务描述

给定一个故事标题，生成包含 5 个句子的短故事，具体例子如表 1 所示。任务的目标为使得生成的故事更具有“人性”，即符合人的审美意识。

表 1: 任务示例

标题	故事
The Test	Jennifer has a big exam tomorrow. She got so stressed, she pulled an all-nighter. She went into class the next day, weary as can be. Her teacher stated that the test is postponed for next week. Jennifer felt bittersweet about it.
The Hurricane	Morgan and her family lived in Florida. They heard a hurricane was coming. They decided to evacuate to a relative's house. They arrived and learned from the news that it was a terrible storm. They felt lucky they had evacuated when they did.
Spaghetti Sauce	Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting. She was touched that he pretended it was good to spare her feelings.

1.2 数据集

我们将原本包含 98161 个样本的 ROCStories[1] 数据集，经过筛选处理成如表 2 所示的训练集、验证集和测试集。

表 2: 数据集划分

数据	数量
训练集	74,777
验证集	9,460
测试集	7,713

[1]https://www.cs.rochester.edu/nlp/rocstories/

1.3 评测标准

我们采用了人工评判和 BLUE 分数两种评测方式，并以人工评判为主。人工评判主要关注以下两个方面：

- 保真度，指故事内容与所给标题应保持一致性，即不能离题。
- 逻辑性，指故事内容的叙述内容需要符合逻辑。

1.4 任务分工

我们将采用论文 [1] 中使用的框架，如图 1.4 所示，先由标题生成故事线 (Storyline)，再又标题和故事线生成最终的故事。

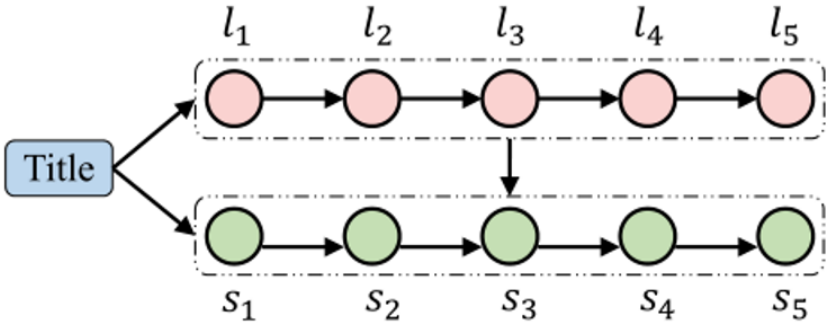

图 1: 整体框架

小组的任务分工如表 3 所示。

表 3: 任务分工

组员	分工
朱仁博	Title → Storyline
潘宜城	Title + Storyline → Story

2 模型：Title → Storyline

2.1 语言模型

2.1.1 模型介绍

如图 2.1.1 所示，我们使用了一个最基本的语言模型。其中，RNN 模型采用了 GRU，Title 输入时，会经过一个 embedding 层，将每个 word 转化成向量，embedding 层的参数采用了 GloVe[2] 预训练，并在模型训练时不固定参数进行微调。GRU 的输出直接连接到一个全连接层，输出下一个单词的概率，最终组成 Storyline。

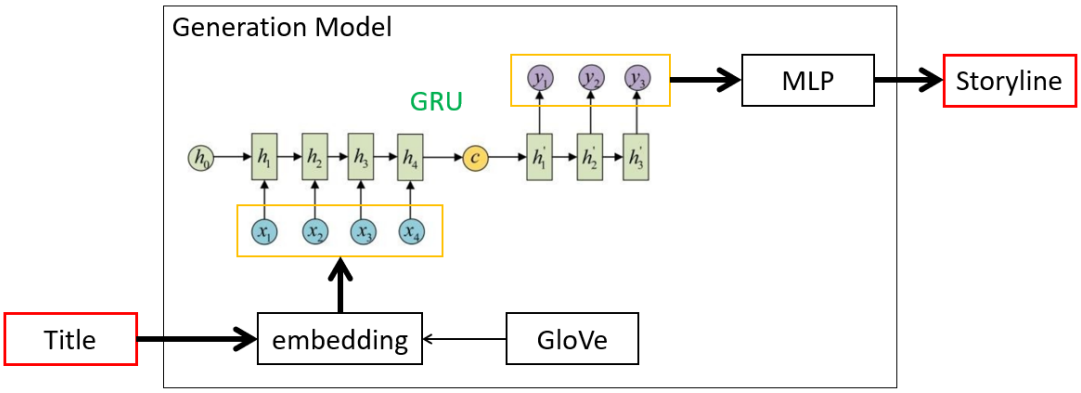

图 2: 语言模型

2.1.2 模型训练

在模型训练时，将训练集所有故事的 Title+Storyline 首尾连接在一起，并按照 batch_size 将其划分，根据 seq_len 从中选取输入，选取输入数据的下一个位置作为输出。这种数据的划分方式解决了标题长度不一致的问题，并使得模型可以有效收敛。训练采用的损失函数为交叉熵。

2.1.3 模型使用

在 Storyline 生成时，依次将 Title 的每一个词 embedding 之后串行输入至 GRU 模型，将最后一个词的输出结果经过全连接层取得 Storyline 第一个词的分布概率，此时可取概率最大或按照概率随机采样，确定第一个词，并将这个词输入至 GRU 模型，以此类推，逐个生成 Storyline 中的每个词，至 Storyline 的结尾符号为止。

此外，上述生成 Storyline 的过程可采用 BeamSearch[3] 算法，这也正式此次任务

[2]https://nlp.stanford.edu/projects/glove/
[3]https://en.wikipedia.org/wiki/Beam_search

使用的算法。BeamSearch 算法可以认为是维特比算法的贪心形式，在维特比所有中由于利用动态规划导致当字典较大时效率低，而 BeamSearch 算法使用 beam_size 参数来限制在每一步保留下来的可能性词的数量。BeamSearch 算法最终的输出为 k 个 Storyline，并带有相关的概率。

2.2 Storyline 存在的问题

在论文 [1] 中使用 RAKE[4] 算法在每句话中提取一个词作为这句话的关键词，最终每个故事的 Storyline 由 5 个词组成。我们发现了一些问题，一方面，在该数据集中，训练数据中某一些词的提取存在一些问题；另一方面，如图 2.2 所示，在故事生成过程中，Storyline 的生成质量对最终 Story 的生成质量极为关键，若 Storyline 生成得不合理，Title → Storyline → Story 的效果将比 Title → Story 还要差。

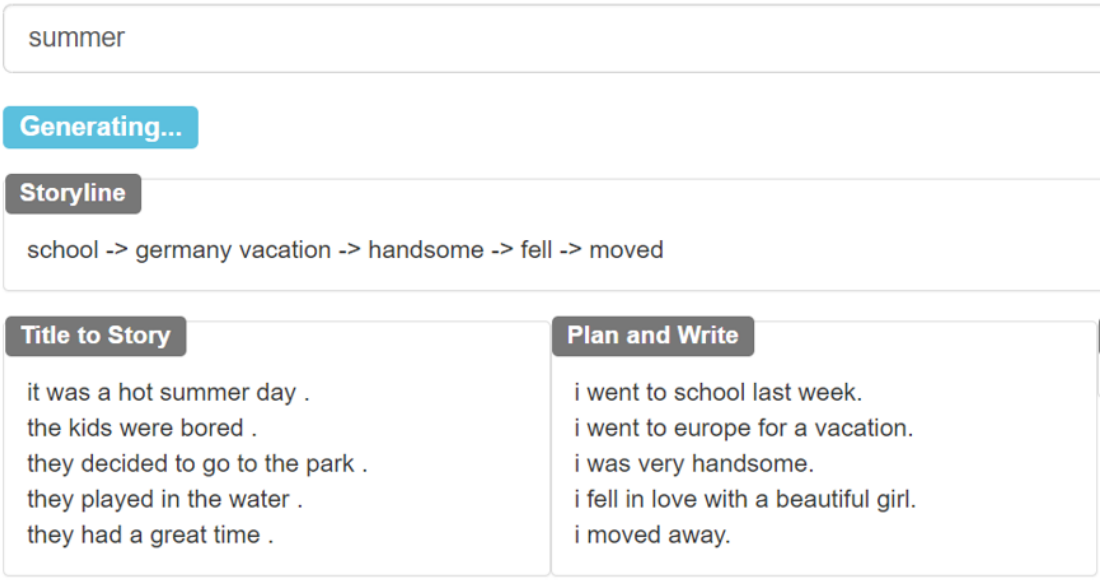

图 3: Storyline 的重要性

针对以上两点，我们分别提了两个方面的改进，第一点改进了 Storyline 的选词方案，修改了关键词提取的规则，使得 Storyline 与标题更吻合且 Storyline 之间联系更紧密；第二点改进了 Storyline 的生成模型，使用了一种生成模型 + 纠正模型的结构，使得 Storyline 的结果更符合 Title。

2.3 Storyline 的选词改进

如图 2.3 所示，我们将每句话的 1 个关键词修改为每句话 2 个关键词，即每个 Storyline 包括 10 个关键词。每句话中的 2 个关键词根据句子的词性提取。接下来将介

[4]https://github.com/aneesha/RAKE

绍训练数据中关键词选取规则，其包括单词过滤、词性过滤、选词组合、选词方向、重复词过滤五个步骤。

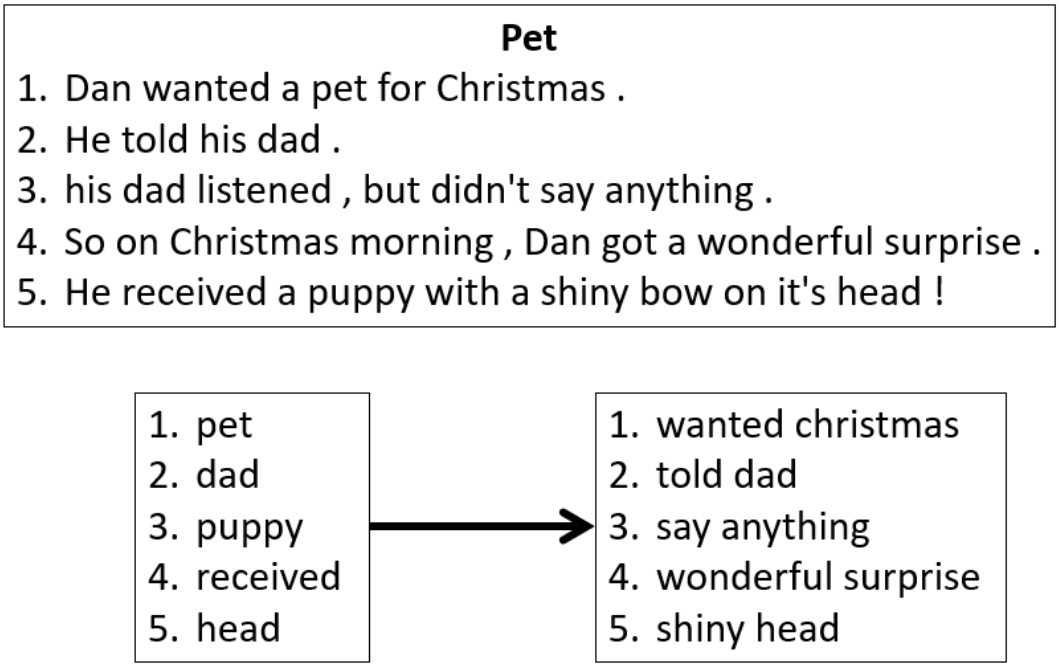

图 4: Storyline 的选词改进

2.3.1 单词过滤

我们过滤了一些在句子中频繁出现但对句子内容无实际意义的单词，如',’、’.’、’i’、’he’、’she’、’it’、’we’、’they’、’was’、’were’。

2.3.2 词性过滤

我们观察了数据集的故事内容，发现每句话的关键内容几乎都集中在名词、形容词或动词上，因此，我们仅在每个句子中提取出以上三种词性的单词，仅当句子中不包含。

2.3.3 选词组合

经过多次实验，我们将选词规则定为以下 7 个优先级。

1. 名词 + 形容词
2. 名词 + 动词
3. 动词 + 形容词

4. 动词 + 动词

5. 形容词 + 形容词

6. 名词 + 名词

7. 任意两词

2.3.4 选词方向

经过实验，我们发现句子的关键信息出现在其后半部分的概率比较大，即后半部分包括了更多的关键词，故我们选词策略为每句话从后往前遍历，直至找到符合当前规则的两个词为止。

2.3.5 重复词过滤

在一条 Storyline 中，五句话可能会包括多个重复的关键词，我们需要在选取 10 个关键词的过程中，考虑去重操作，即每次选取的关键词不与已经确定的关键词重复。

2.4 Storyline 的生成改进

2.4.1 模型介绍

在 Storyline 的选词过程中，我们选取的词仅使用了一种固定的规则，虽然较原有 Storyline 效果有所提高，但仍然存在不少问题。其可能造成的影响为，由于关键词提取的偏差，造成生成模型的效果偏差。为了消除这种偏差，我们尝试使用一个故事的多条 Storyline，以此训练生成模型。但是这种单输入多输出的模型不符合逻辑，会造成生成模型的不收敛，故我们引入另外一个如图 2.4.1 所示的纠正模型，该模型的语言模型与生成模型完全相同，仅仅是输入和输入元素的对调。

2.4.2 模型训练

不同于生成模型，在模型训练时，我们不再将所有 Storyline+Title 的数据首尾连接在一起，而每条数据视为单独的一个样本。在数据处理时，我们已经将 Title 的单词数量限制在 5 个及以内，并将 Title 以结尾符填充至 5 长度。输入模型的 Storyline+Title 的前 4 个词，以此取得模型输出的最后 5 个词，即对应生成的 Title，与真实的 Title 取交叉熵作为损失函数。

2.4.3 模型使用

如图 2.4.3 所示，将生成模型与纠正模型相结合。对于每一个输入的 Title，经过一个生成模型，由 BeamSearch 算法产生多条不同的 Storyline。我们将多个不同

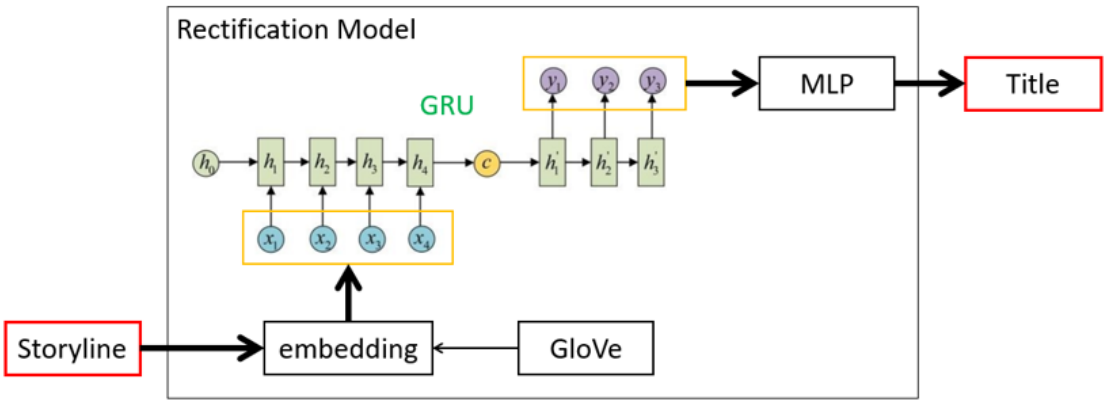

图 5: 纠正模型

模型输出的 Storyline 视为 Title 的候选结果。将每一个 Storyline 输入至纠正模型中，由 BeamSearch 算法产生多条不同的 GenTitle，并将这些与原 Ttitle 计算相似度 Sim(GenTitles,Title), 将此分数作为该 Storyline 的候选分数 Score(Storyline)。

$$Score(Storyline) = \sum_{i} Sim(GenTitles_i, Title)$$

其中两个标题间的相似度被定义为两两单词之间余弦相似度的最大值。

$$Sim(T1, T2) = \underset{i=0,\ldots,n;j=0,\ldots,m}{MAX} CosSim(T1_i, T2_j)$$

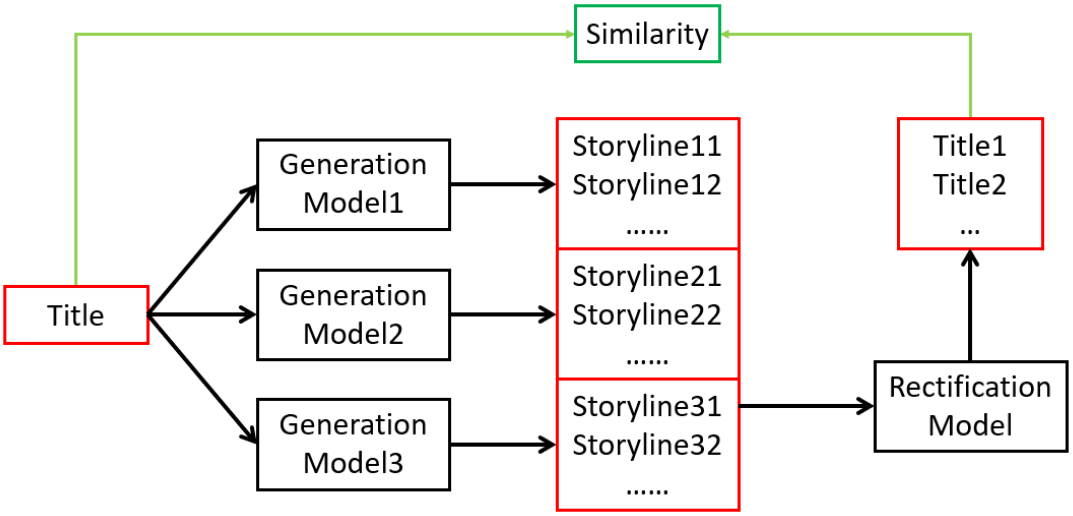

图 6: 生成模型 + 纠正模型

最终，取分数最高的 Storyline 作为 Title 的输出。此外，在经过一批次 Title 的使用后，使用此批 Title+Storyline 再次训练生成模型，以微调其参数，起纠正作用。

3 实验：Title → Storyline

3.1 生成模型

我们训练了个 seq_len 分别取 20、50、80 的 3 个生成模型，其 batch_size 均为 100，最终的训练结果如表 4 所示。

表 4: Generation Model Results

Model	**seq_len**	**Train PPL**	**Valid PPL**	**Test PPL**
Generation Model1	20	47.84	78.47	77.98
Generation Model2	50	35.22	76.60	76.72
Generation Model3	80	31.56	78.51	78.34

3.2 纠正模型

我们由原来的数据集，重新生成纠正模型的训练集、验证集、测试集的数据，分别通过限制每个故事最多生成的样本数量，重新制作了 4 个不同规模的数据集。图 3.2 表示 4 个数据集的训练集样本数量, 其中 Dataset1 表示与原 Title → Storyline 数据集相同，仅调换了 Title 和 Storyline 的位置；Dataset10 表示每个故事最多新增了随机生成的 10 个 Storyline+Title 样本，并且添加了原数据集；剩余 2 个同理。

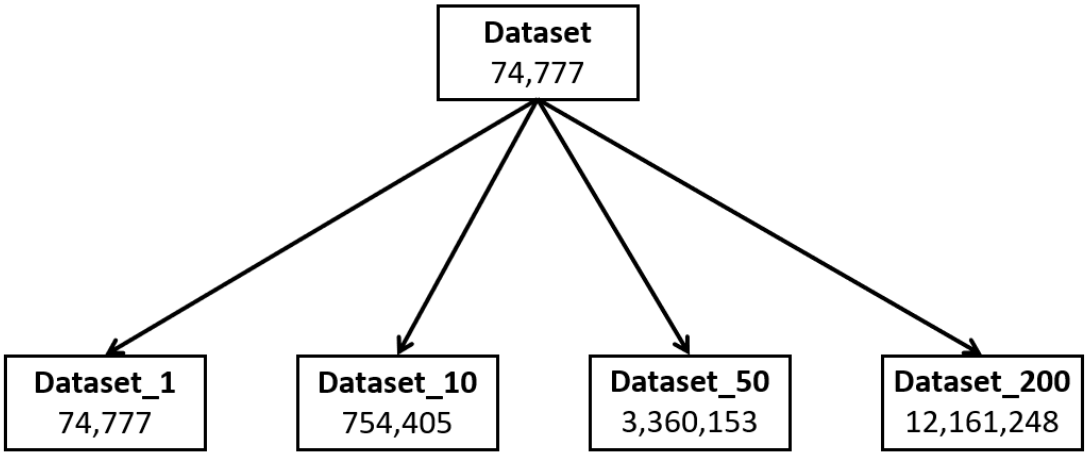

图 7: 纠正模型样本

我们分别使用以上 4 个数据集训练了 4 个纠正模型，其最终训练结果如表 5 所示。

表 5: Rectification Model Results

Model	Dataset	Train PPL	Valid PPL	Test PPL
Rectification Model1	Dataset1	7.90	12.97	12.91
Rectification Model2	Dataset10	9.83	13.14	13.26
Rectification Model3	Dataset50	7.81	15.09	14.95
Rectification Model4	Dataset200	6.14	14.69	14.87

3.3 实验内容

我们将比较 Generation Model1、Generation Model2、Generation Model3、Generation Models+Rectification Model1、Generation Models+Rectification Model2、Generation Models+Rectification Model3、Generation Models+Rectification Model4 这 7 个模型的由 Title 生成 Storyline 的效果。分别使用每个模型产生 100 个测试集数据的 Storyline，将 700 个 Title+Storyline 数据打乱，并根据表 6 所示的标准给其人为评分。

表 6: Criterion

State	Score
Title 与 Storyline 无联系，Storyline 不存在逻辑性	0
Title 与 Storyline 相联系，Storyline 不存在逻辑性	0.5
Title 与 Storyline 无联系，Storyline 存在逻辑性	0.5
Title 与 Storyline 相联系，Storyline 存在逻辑性	1

3.4 实验结果

经过人为评判，并最终统计汇总，我们得到如表 7 所示的评测结果。可以看出，在原来的三个生成模型中，seq_len 选取的越大，生成 Storyline 将更好。增加判别模型后，仅仅是不添加其他数据集的 Rectification Model1 也能提升生成 Storyline 的质量，这说明了图 2.4.3 所示的模型确实有一定效果。使用少量数据的纠正模型，可以使得生成 Storyline 的质量大幅度提高，这个结果是令人意想不到的。另外，由于数据集的增加，导致的过拟合现象严重，故使得后两个纠正模型的效果有所下降，这也符合预期。

表 7: Model Results

Model	**Score**
Generation Model1	0.49
Generation Model2	0.57
Generation Model3	0.59
Generation Models Rectification Model1	0.67
Generation Models Rectification Model2	**0.75**
Generation Models Rectification Model3	0.72
Generation Models Rectification Model4	0.60

4 模型：Title + Storyline → Story

4.1 L2W Story Model

Learning to Write 是论文 [2] 中用于 Text Continuation 的方法。论文中利用辅助的语言学模型缓解单纯的语言模型的缺陷，实现了更加合理的文本续写。如表 8 中的例子。这里借鉴他们的改进思路，也对故事生成这个任务采取语言学模型缓解缺陷。

表 8: Text continuation example

Context: The two guards thudded into the door on the other side and began pounding on it, shouting furiously. Scious raised the large bunch of keys then placed them in a large pocket in his coat. "Come, we have to go," he whispered and moved up the corridor, the guards still hammering on the door. Jared looked around. The table next to the room they had been in was bare.
Continuation: Only a flagon of wine and a couple of mugs plus a crude dice game. Light flickered in the stone corridor from lanterns on the walls. The place was dank and gloomy, moss in scattered patches on the floor and walls. The corridor ended just beyond the door to their former prison. No one else was about.

在故事生成中，标题和故事线是上下文，要生成的故事作为续写的文本。

如表 9 所示，在 L2W 中，为了弥补单纯使用语言模型的的缺陷，在训练完语言模

型之后，还训练了 4 个额外的语言学评分模型，如表每个区分判别模型通过判断生成的序列的好坏程度，在众多可能的候选序列中选择最好的生成序列。

表 9: L2W Communication Model

Model	Description
Repetition Model	重复程度评分模型，给予复读情况少的文本更高的分数
Entailment Model	蕴涵评分模型，给予包含合理蕴含推理关系地文本更高的分数
Relevance Model	相关程度评分模型，给予和上下文信息相关程度高的文本更高的分数
Lexical Model	词法风格评分模型，给予用词多样性高的文本更高的分数

由于蕴含（Entailment）评分模型更加注重于文本内逻辑的合理性，一般在具有较长的信息较多的上下文中才能充分体现其重要性，而故事生成任务中的上下文定义为短的标题和故事线，并不具备逻辑上的连续性，这里不使用该模型。

下面分别介绍采用的 3 个评分模型，并说明他们所使用的训练数据集。

4.2 Repetition Model

重复模型用于评价生成的语句的重复程度，通过给重复程度高的语句给予低的分数，给真值语句提供高的分数，期望模型能够更加偏好生成重复度低的语句，以此减少语言模型的复读现象。

假设续写的故事为 y，其分量 y_i 表示一个词。然后计算每个词 y_i 与之前的 k 个词的向量表示的相似度，通过对其求最大值即可反映当前词 y_i 的重复程度。因此，当一个词重复出现过，则该值为 1.0。如下式所示：

$$d_i = \max_{j=i-k...i-1} \{CosSim\left(e(y_j), e(y_i)\right)\}$$

然后通过一个 RNN 表示故事 y 的总体重复评分：

$$s_{rep}(y) = w_r^T \mathrm{RNN}_{rep}(d)$$

至于训练数据，因为语言模型生成的故事存在重复的可能性比较大，可以将其作为寻览的负样本，而真值故事作为正样本。通过比较它们的重复评分，使用排名对数似然损失函数训练。假设 x 表示上下文标题和故事线，y_g, y_{lm} 分别为真值故事和语言模型

生成的故事，训练目标如下所示：

$$L_{rep} = \sum_{(x,y_g,y_{lm})} \log \sigma \left(s_{rep}(y_g) - s_{rep}(y_{lm}) \right)$$

4.3 Revelance Model

相关程度评分模型用于衡量生成的故事和上下文标题及故事线的相关程度，这样的目的是为了生成的故事能够尽可能符合标题和故事线。

对一个训练样本 x, y，首先使用卷积操作生成向量表示

$$a = \text{maxpool} \left(\text{conv}_a(e(x)) \right)$$
$$b = \text{maxpool} \left(\text{conv}_b(e(y)) \right)$$

然后生成相关程度评分 (使用按元素的乘法 $\circ$，这样当向量表示相近时评价便更高):

$$s_{rel}(x, y) = w_l^T \cdot (a \circ b)$$

至于训练数据，原本的故事随机打乱之后作为负样本 (使其和上下文标题及故事线不相关)，真值故事作为正样本，使用排名对数似然损失函数训练：

$$L_{rel} = \sum_{(x,y_g,y_{rnd})} \log \sigma \left(s_{rel}(x, y_g) - s_{rel}(x, y_{rnd}) \right)$$

4.4 Lexical Style Model

词法风格模型目的是让生成的故事中的用词具有多样性，而不是总是重复使用一个单词。因此对于词法风格的刻画只需要用生成的故事 y 本身，如下：

$$s_{lex}(y) = w_s^T \cdot \text{maxpool}(e(y))$$

考虑到语言模型的词法风格一般来说比较单一，训练数据和重复模型一样使用语言模型生成的故事作为负样本，真值故事作为正样本，最大化下面的对数似然函数：

$$L_{lex} = \sum_{(x,y_g,y_{lm})} \log \sigma \left(s_{lex}(y_g) - s_{lex}(y_{lm}) \right)$$

4.5 Mixture Model

混合模型生成故事序列的时候使用上面 3 个模型和语言模型 P_{lm} 的对数概率的线性组合 (在对数域内的线性组合等价于原概率的相乘)，给定上下文标题和故事线 x，则生成故事 y 的对数概率为：

$$f_\lambda(x, y) = \log(P_{lm}(y|x)) + \sum_k \lambda_k s_k(x, y)$$

为了学习上面的线性组合参数 λ_k，固定基础模型，在验证数据集上训练，假设模型的生成函数为 $\mathcal{A}(x)$，则训练的目标使最小化模型预测的故事和真值故事的对数概率的平方差：

$$L_{mix} = \sum_{(x,y)\in D_{valid}} (f_\lambda(x,y) - f_\lambda(x,\mathcal{A}(x)))$$

4.6 Beam Search

上面的混合评分模型仅用于评价一个生成是否好坏，而从一个概率模型中采样多个可能的候选故事则需要使用 Beam Search。Beam Search 的思路非常简单，在概率模型生成词的时候，例如故事生成的语言模型，它们的输出并不是一个固定的词，而是所有词的一个概率分布。在通常的文本生成中，只在这个概率上采样一个词作为输出，然后继续生成下一个词。而 Beam Search 则会在这个概率上采样多个词，并且都保留到下一轮的采样，最后会生成大量的候选故事，依据额外的规则将它们排序并选择最好的。

算法的伪代码如下：

Input: Sample Model P, Context x,
Output: Best story y
Set beam $B = [x]$;
Set *best*=None;
while B *has better candidate than best* ***or*** *not reached max step* **do**
  Use P to expand beam B to B_{new};
  Calculate the scores of each story in B_{new};
  Set B to the best k story in B_{new};
  Set *best* to the best in B;
end

Algorithm 1: Beam Search

利用上面的 Beam Search 算法和混合评分模型，在测试集上生成最后的输出。

5 实验：Title + Storyline → Story

5.1 L2W Results

实验部分介绍区分评价模型的作用，通过对 BeamSearch 的样例的分析展示它的作用。采用 L2W 模型包含多个模型，这里逐一介绍。

5.1.1 语言模型

语言模型在这个框架中是可以通用的，可以使用任何合理的模型设计，这里论文 [2] 中提供了一个 AdaptiveSoftmax 的语言模型，这里也使用了上一次的作业的语言模型，它们的结果如表10 所示。考虑到 Project1 LM 的测试集 PPL 较好，后面均采用该模型。

表 10: LM Model Results

Model	Train PPL	Valid Loss	Test PPL
AdaptiveSoftmax LM	60.86	64.97	62.18
Project1 LM	79.84	36.60	34.12

5.1.2 生成评分模型结果

重复评分模型和词法评分模型均是在语言模型生成的故事和真值故事上训练的，相关程度评分模型实在随机打乱的故事和真值故事上训练。因为最终评测的时候要使用它们，所以没有评测测试集的准确率。最终结果如表 11 所示。

表 11: Scorer Results

Scorer	Train Acc	Valid Acc
Repetition	0.98	0.95
Relevance	0.65	0.98
Lexical	0.99	0.98

5.2 Case Study

为了直观理解上述工作的效果，这里从测试集中选取了一些样例对比语言模型和使用混合评分模型的输出。从表 12 中可以看到语言模型生成的故事重复了一句话，而混合模型则没有明显的重复。从词法上看，前者的用词也比后者更加多样化一点。

为了更加详细地说明混合模型中各个部分对故事生成的影响，这里将 Beam Search 中的中间过程以及对应的各个模型给出的评分进行比较，结果如表13 所示。从表格中可以看到，对于重复程度模型，由于输入的上下文中包含 books，所以第二个生成的故事的重复模型分数最低，其他 3 个基本一致。对于相关性模型，第 4 个故事达到了最高的分数，这是因为它提到了 novels 这个关键词，这和上下文中的 write 有很好的对应关系。至于词法模型，这个例子的大部分词都相同，模型给了第 2 个例子较高的分数，但是词 books 明显出现过了，这里应该是应为第 2 个故事的词的数量少，所以在计算词法分数的时候有较高的分数 (分数是考虑所有词获得的)。这种情况也发生在语言模型上，故事短的第 2 个例子语言模型得分较高。

表 12: Comparison of LM and Mixture Model

	Text
Context(Title, Storyline)	literature vs math ⟨EOT⟩ loved literature path indecisive more books love strong write problems ⟨EOL⟩
LM	I wanted to learn how to read. *I read a book of books.* I decided to write a novel. *I read a lot of books.* I was able to write a novel.
Mixture Model	Charles loved reading books. One day , he decided to learn how to draw. His friend suggested that he should write an essay. He wrote a lot of research and read it. He was able to write a novel.

表 13: Mixture scores during Beam Search. LM,Lex,Rel,Rep,Mix mean language model, lexical model,relevance model,repetition model, mixture model.

Context(title,storyline)	literature vs math ⟨EOT⟩ loved literature path indecisive more books love strong write problems ⟨EOL⟩				
Generated Story	LM	Lex	Rel	Rep	Mix
charles loved reading. but	-13.64	-0.19	-0.34	-0.33	-20.67
charles loved reading books.	**-10.01**	**-0.17**	-0.13	-0.49	**-15.29**
charles loved reading. he	-11.11	-0.30	-0.34	**-0.30**	-18.74
charles loved reading his novels.	-16.03	-0.24	**-0.12**	-0.33	-20.85

最后的模型也使用 Bleu 分数进行了评测，结果如表14所示。

表 14: BLEU Score

Storyine	B1	B2	B3	B4	Bavg
Generated	33.20	13.18	2.92	0.66	2.26

参考文献

[1] L. Yao, N. Peng, R. M. Weischedel, K. Knight, D. Zhao, and R. Yan, "Plan-and-write: Towards better automatic storytelling," *arXiv preprint arXiv:1811.05701*, 2018.

[2] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi, "Learning to write with cooperative discriminators," *arXiv preprint arXiv:1805.06087*, 2018.