复旦大学计算机科学技术学院

2019-2020 学年第一学期期末论文课程评分表

课程名称： 自然语言处理 **课程代码：** COMP130141.01

开课院系： 计算机科学技术学院

学生姓名： 马逸君 **学号：** 17300180070 **专业：** 计算机科学与技术

论文名称： 学术论文自动生成摘要的探究

（以上由学生填写）

成绩：

学术论文自动生成摘要的探究

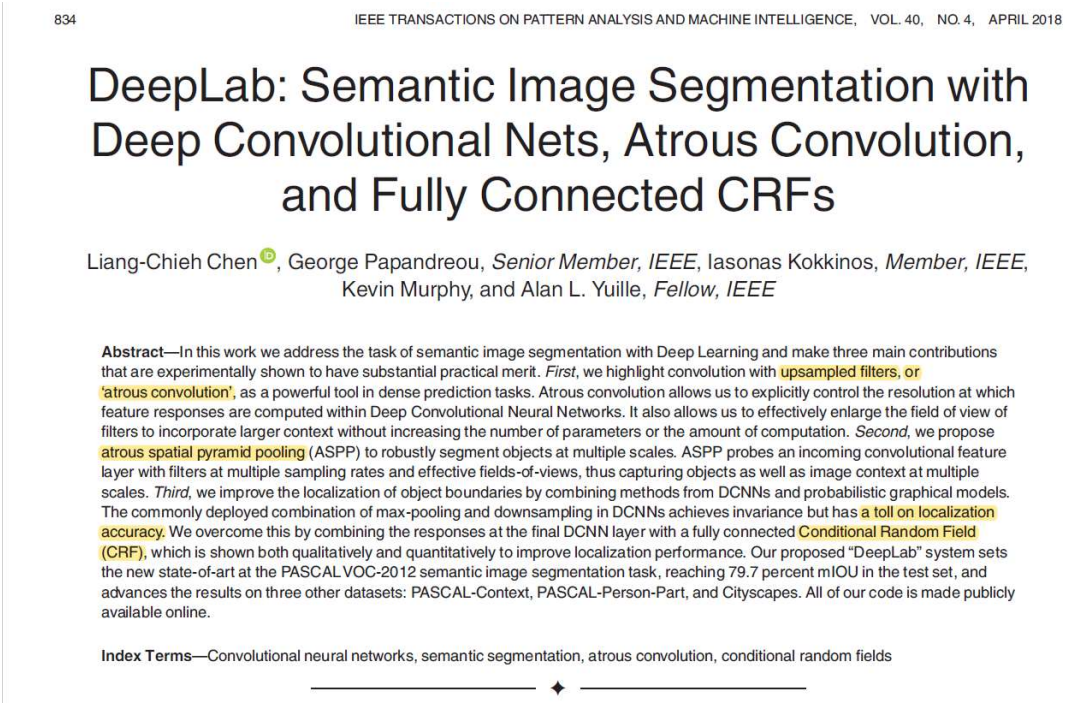
马逸君 17300180070

1 问题引入

本次选题的想法来源于本学期刚刚开始接触的学术研究。

本学期的两门课程都有要求读 paper，因为我之前从未接触过，读起来非常缓慢，一篇 15 页左右的顶会，我需要读一整天。对此，一位学姐教导我如果非必要的话可以不读正文，从摘要部分就可以了解到重要的信息，这也体现出了摘要在学术论文中的重要性。

期末时，另一门课程要求我们以标准学术论文格式撰写课设报告。作为从来没有写作论文经验的人，我就先写了正文部分，最后由于时间所迫，我写摘要的方法就是：从正文的每一章节摘取一句话拼合到一起，就构成了摘要。这样一来我产生了想法，是否有可能将这样的过程自动化，让计算机自动生成学术论文的摘要呢？这就是本次课设的灵感来源。



DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs

Liang-Chieh Chen, George Papandreou, *Senior Member, IEEE*, Iasonas Kokkinos, *Member, IEEE*, Kevin Murphy, and Alan L. Yuille, *Fellow, IEEE*

Abstract—In this work we address the task of semantic image segmentation with Deep Learning and make three main contributions that are experimentally shown to have substantial practical merit. *First*, we highlight convolution with upsampled filters, or 'atrous convolution', as a powerful tool in dense prediction tasks. Atrous convolution allows us to explicitly control the resolution at which feature responses are computed within Deep Convolutional Neural Networks. It also allows us to effectively enlarge the field of view of filters to incorporate larger context without increasing the number of parameters or the amount of computation. *Second*, we propose atrous spatial pyramid pooling (ASPP) to robustly segment objects at multiple scales. ASPP probes an incoming convolutional feature layer with filters at multiple sampling rates and effective fields-of-views, thus capturing objects as well as image context at multiple scales. *Third*, we improve the localization of object boundaries by combining methods from DCNNs and probabilistic graphical models. The commonly deployed combination of max-pooling and downsampling in DCNNs achieves invariance but has a toll on localization accuracy. We overcome this by combining the responses at the final DCNN layer with a fully connected Conditional Random Field (CRF), which is shown both qualitatively and quantitatively to improve localization performance. Our proposed "DeepLab" system sets the new state-of-art at the PASCAL VOC-2012 semantic image segmentation task, reaching 79.7 percent mIOU in the test set, and advances the results on three other datasets: PASCAL-Context, PASCAL-Person-Part, and Cityscapes. All of our code is made publicly available online.

Index Terms—Convolutional neural networks, semantic segmentation, atrous convolution, conditional random fields

学术论文的摘要部分包含很多重要信息，
我在读 paper 时往往也在摘要部分有较多标记

目前在摘要生成问题上的可选算法有[12]：

直接抽取文本的头三句(lead-3)、排名(TextRank)、聚类等简单算法；

建模成序列标注问题，然后使用 GRU 等模型进行抽取式摘要，或使用 Seq2Seq 模型，或者是多种方法共同使用，如序列标注+Seq2Seq+强化学习；

建模成句子排序问题，然后结合 ROUGE 评测方式使用 GRU 和 MLP 完成；

利用 Seq2Seq 模型完成生成式摘要，包括多任务学习、生成对抗等多种改进和变体；

也可以结合生成式和抽取式，多种算法综合运用。

本文着重探究基于 Seq2Seq 模型的 Pointer-Generator 深度学习模型（生成式）和基于朴素贝叶斯分类器的非深度学习方法（抽取式）。

2 数据集

显而易见，为了完成这一任务，我们需要找到大量的学术论文并（至少）分离出正文和摘要，来作为我们的数据集。

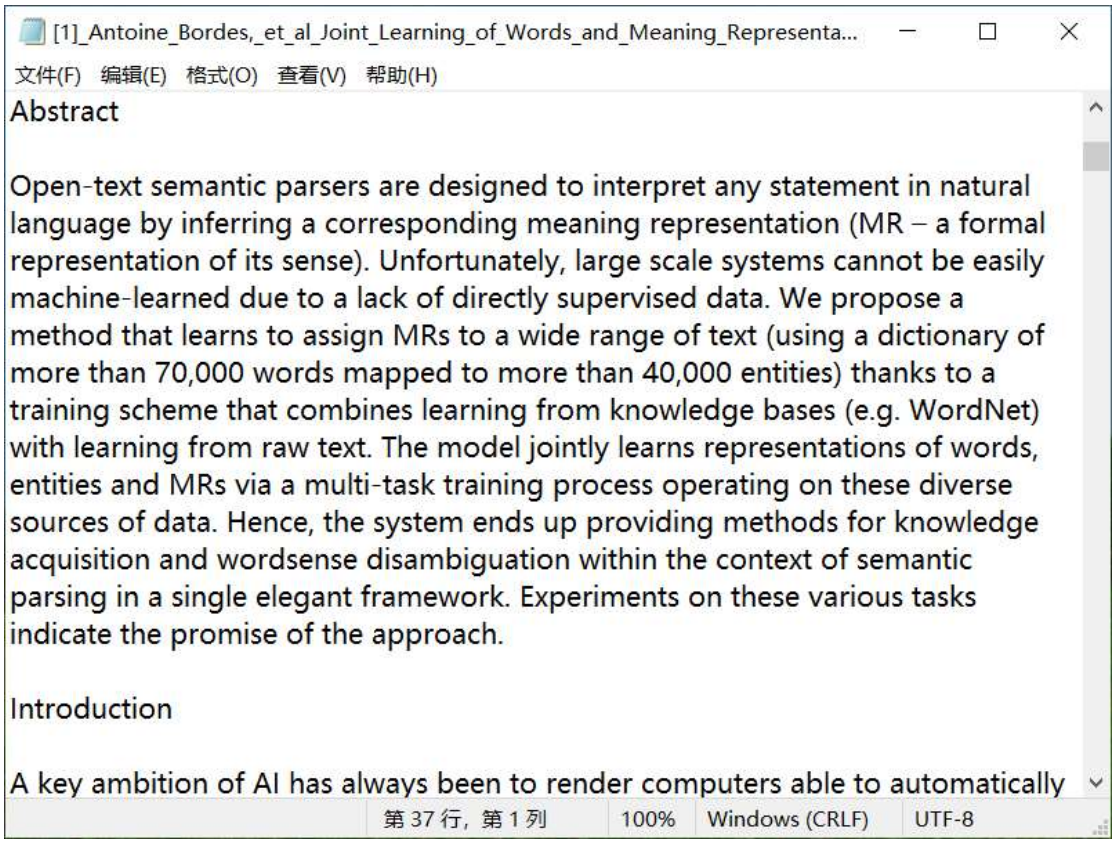
最开始我试图自己制作数据集，而第一步就是**查找并获得大量学术论文**。我在 github 上查找是否有人整理了学术论文的语料库（用"paper corpus"等作为搜索词），但没有找到。随后我改为查找顶会学术论文文集，在 github 上找到了 awesome-deep-learning-papers[1]、Deep-Learning-Papers-Reading-Roadmap[2]、papers-we-love[3]等论文集合。当然我没有特意收集深度学习领域的论文，但搜索"papers"这样的通用搜索词得到的排名靠前的搜索结果却都是关于深度学习的，足见这一领域最近确实非常火。按上述方法我一共搜集到 338 篇学术论文。

名称	修改日期	类型	大小
[1]_Antoine_Bordes,_et_al_Joint_Lear...	2020/1/10 3:59	Adobe Acrobat ...	374 KB
[1]_LeCun,_Yann,_Yoshua_Bengio,_an...	2020/1/10 3:51	Adobe Acrobat ...	2,035 KB
[2]_Mikolov,_et_al_Distributed_repre...	2020/1/10 3:59	Adobe Acrobat ...	110 KB
[3]_Sutskever,_et_al_"Sequence_to_s...	2020/1/10 4:00	Adobe Acrobat ...	140 KB
[4]_Girshick,_Ross_Fast_r-cnn_Procee...	2020/1/10 4:01	Adobe Acrobat ...	236 KB
[5]_Yoon_Kim,_et_al_Character-Awar...	2020/1/10 4:00	Adobe Acrobat ...	470 KB
[7]_Karl_Moritz_Hermann,_et_al_Teac...	2020/1/10 4:00	Adobe Acrobat ...	6,187 KB
[8]_Alexis_Conneau,_et_al_Very_Dee...	2020/1/10 4:00	Adobe Acrobat ...	92 KB
[9]_Armand_Joulin,_et_al_Bag_of_Tric...	2020/1/10 4:00	Adobe Acrobat ...	70 KB
[14]_Hinton,_Geoffrey_E,_et_al_Impro...	2020/1/10 3:52	Adobe Acrobat ...	1,627 KB
[17]_Ba,_Jimmy_Lei,_Jamie_Ryan_Kiro...	2020/1/10 3:53	Adobe Acrobat ...	599 KB
[18]_Courbariaux,_Matthieu,_et_al_Bi...	2020/1/10 3:53	Adobe Acrobat ...	262 KB
[19]_Jaderberg,_Max,_et_al_Decoupl...	2020/1/10 3:53	Adobe Acrobat ...	5,732 KB
[20]_Chen,_Tianqi,_Ian_Goodfellow,_a...	2020/1/10 3:53	Adobe Acrobat ...	502 KB
[21]_Wei,_Tao,_et_al_Network_Morp...	2020/1/10 3:54	Adobe Acrobat ...	1,089 KB
[22]_Sutskever,_Ilya,_et_al_On_the_im...	2020/1/10 3:54	Adobe Acrobat ...	526 KB
[23]_Kingma,_Diederik,_and_Jimmy_B...	2020/1/10 3:54	Adobe Acrobat ...	571 KB

搜索到的几百篇学术论文（如果全部读一遍可能需要若干个月的时间）

值得讨论的是，其实本次任务并不一定非要使用顶会 paper，可以从学术期刊等其他途径找到更大规模的语料，但考虑到这些来源获得的论文质量低于顶会，其摘要的质量可能也差一些，故不予采用。

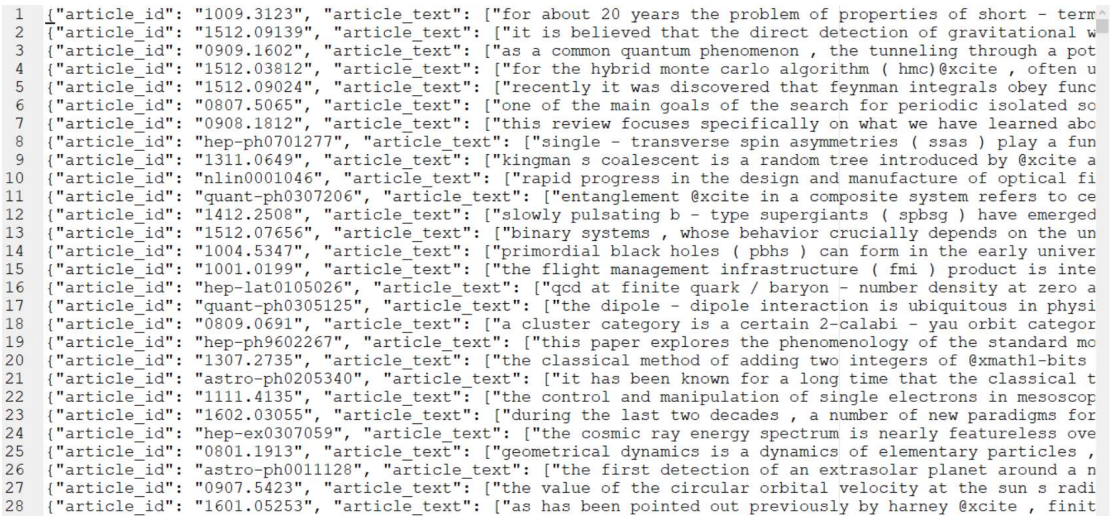
第二步，**处理上一步获得的学术论文。**在第一步的过程中，笔者顺便搜索到一个自动将英文论文 pdf 文件的文本提取出来、并调用谷歌翻译 API 译成中文输出的程序[4]，笔者也参照着写了一个将 pdf 文件的文本提取并写入 txt 文件的程序。



上一页图片中的第一篇论文提取文本的结果

在此基础上，可以利用学术论文本身的格式要求，在文本里匹配"Abstract""Reference|Bibliography"这些关键字来提取摘要、去除参考文献及题头作者信息等无关内容、分离文章的各章节，以进一步获得我们想要的信息。

此外，受制于如此之小的样本，笔者（因为之前从来没有接触过爬虫）也考虑过自学爬虫来批量自动下载学术论文。但非常幸运的是，在进行这一步之前，我成功找到了**现成的学术论文数据集** arxiv-dataset[4]和 pubmed-dataset[5]，前者的训练集包含的 paper 数量以万计，这样一来，既获得了成熟的标注数据，也不必花时间在自行采集数据源上了。故以下都采用现成的数据集 arxiv-dataset。



arxiv-dataset 的测试集。共 6440 篇，

论文编号、正文文本、摘要等部分都是提取分离好的

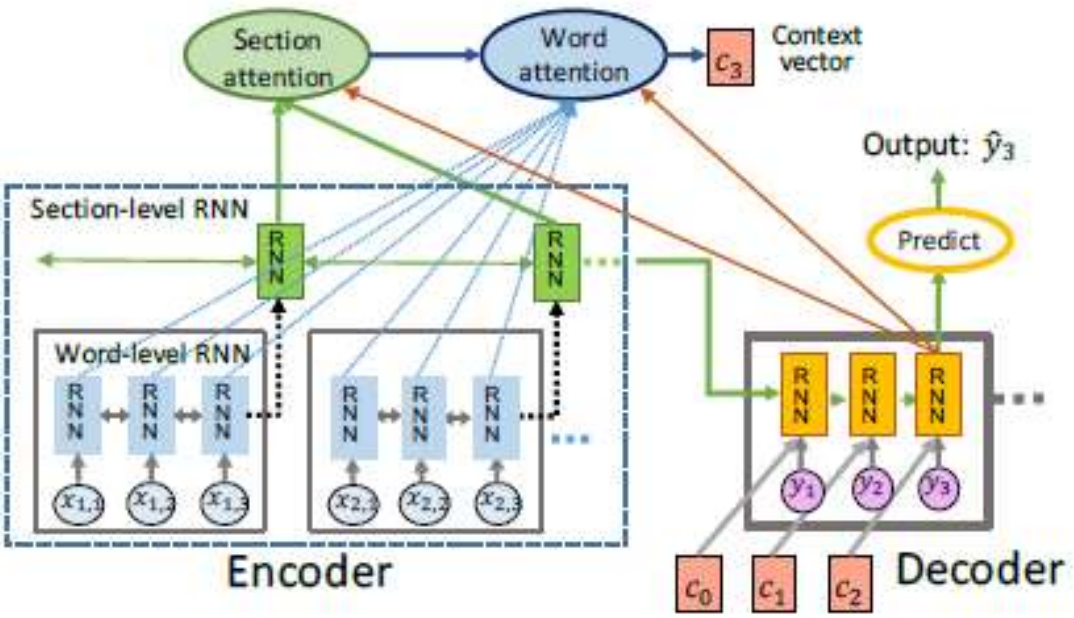
3 深度学习方法：语篇感知注意力模型

3.1 原理与实现

笔者首先尝试了近期最火的深度学习方法。

显而易见的是，学术论文生成摘要与普通文本生成摘要的一大不同之处，就是学术论文具有严格而统一的正文格式，例如其必分为介绍、相关工作、核心算法、总结等章节。在搭建深度学习模型时，我们需要充分利用学术论文的这一特性，才能得到特别适用于学术论文的摘要生成模型。

笔者使用 ACL 2018 的论文《长文档生成式摘要的语篇感知注意力模型》（"A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents"）[6][7]提出的模型结构，该模型是 github 上学术论文摘要生成方面 star 数最高的模型之一；该模型由 Pointer-Generator 模型[8]改进而来，模型[8]基于 ACL 2017 的论文《切入点：“指针生成器”神经网络进行文本摘要》（"Get To The Point: Summarization with Pointer-Generator Networks"）[9]，在 github 上获得了 1.6K 的高 star 量。



以下介绍本语篇感知注意力模型的结构。示意图如上所示。

编码器：与一般的编码器解码器模型不同的是，本模型使用分层 RNN 以捕获文档语篇结构，先对每一章节进行编码，再对整篇文档进行编码。从技术上说，文档的编码向量 d 的表达式是：

$$\mathbf{d} = \mathrm{RNN}_{doc}\left(\{\mathbf{h}_1^{(s)}, ..., \mathbf{h}_N^{(s)}\}\right)$$

$$\mathbf{h}_j^{(s)} = \mathrm{RNN}_{sec}\left(\mathbf{x}_{(j,1)}, ...\mathbf{x}_{(j,M)}\}\right)$$

其中 $h^{(s)}_j$ 是文章第 j 章节部分的编码向量，$x_{(j,\ i)}$是该章节中第 i 个符号(token)w(i, j)对应的稠密向量(dense embedding)。RNN_{sec} 的参数在文档所有章节之间是共享的。RNN_{doc} 和 RNN_{sec} 都带有单层双向 LSTM，并使用一个简单前馈神经网络将前向和后向 LSTM 状态合成一个：（式中的“[,]”表示连接）

$$\mathbf{h} = \mathrm{relu}(\mathbf{W}([\overrightarrow{\mathbf{h}}, \overleftarrow{\mathbf{h}}] + \mathbf{b})$$

解码器：Frederick Suppe 在 1998 年曾经说过，科学论文的摘要一般包括问题描述、方法讨论、结果与结论（刊载于《Philosophy of Science》）；人们一般撰写长文章摘要时，往往也会写到来自文章不同章节的各个重点。因而该模型也采用语篇感知的注意力模型

（discourse-aware attention model），在解码器的每一步都用语篇相关的参数来调整字词级别的注意力函数值。从技术上说：

$$\mathbf{c}_t = \sum_{j=1}^{N} \sum_{i=1}^{M} \alpha_{(j,i)}^{(t)} \mathbf{h}_{(j,i)}^{(e)}$$

$$\alpha_{(j,i)}^{(t)} = \underset{(i,j)}{\mathrm{softmax}} \left(\beta_j^{(t)} \, \mathrm{score}(\mathbf{h}_{(j,i)}^{(e)}, \mathbf{h}_{t-1}^{(d)}) \right)$$

$$\beta_j^{(t)} = \underset{j}{\mathrm{softmax}}(\mathrm{score}(\mathbf{h}_j^{(s)}, \mathbf{h}_{t-1}^{(d)}))$$

c_t 是源文档的上下文向量(context vector)，$h^{(e)}_{(i,j)}$是语篇第 j 章节中第 i 个字词的编码状态，$\alpha^{(t)}_{(j,i)}$是与该编码状态配套的注意力权重，t 为时间戳，score()是常见的累加计分函数，$\beta^{(t)}_{(j)}$也是随时间更新的一个参数值。

除此之外，为了解决未知单词（词典外单词 out of vocabulary words）的问题，本模型采用基于概率的算法，以一定的概率从原文中拷贝单词到最终输出结果中；且为了避免长文本处理任务中生成式模型不断生成重复文段序列的问题，还对注意力范围进行了跟踪。

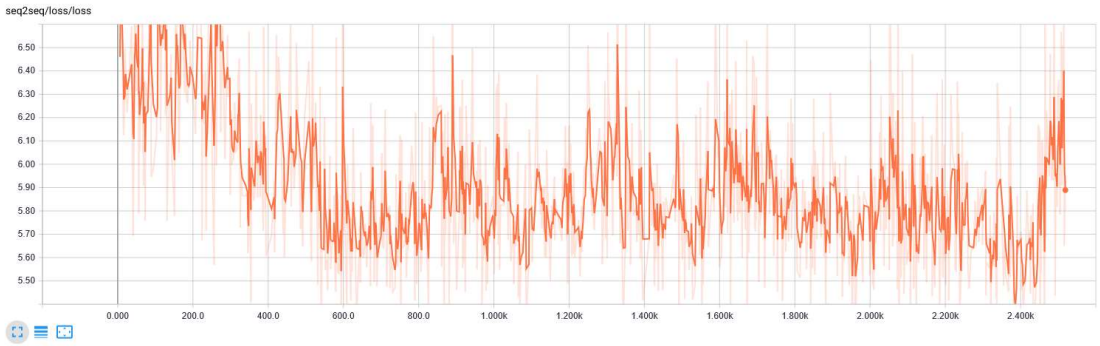
3.2 过程与实验结果

按照作者的开源代码提供的脚本，笔者使用如下参数训练该模型：

```
python run_summarization.py \
--mode=train \
--data_path=./data/arxiv-release/train.bin \
--vocab_path=./data/arxiv-release/vocab \
--log_root=logroot \
--exp_name=test-experiment-arxiv \
--max_dec_steps=210 \
--max_enc_steps=2500 \
--num_sections=5 \
--max_section_len=500 \
--batch_size=2 \
--vocab_size=50000 \
--use_do=True \
--optimizer=adagrad \
--do_prob=0.25 \
```

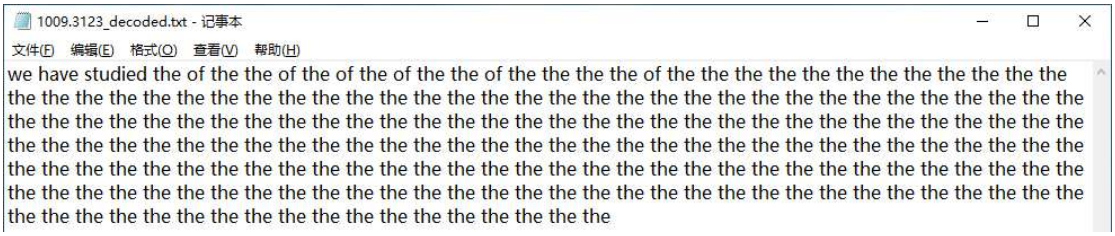
其中值得解释的几个参数：max_dec(enc)_steps 是解(编)码器时间步骤上限值；do_prob 是 LSTM 单元的 dropout 概率；batch_size 作者给的默认值为 4，受限于有限的 RAM 大小（8GB 内存 + 6GB 显存），我们无法在 batch_size=4 的设置下运行，故减小为 2。

因为作者并未在该脚本中显式地提供训练结束相关的设定，且笔者是第一次接触深度学习，经验不足，所以在训练刚两千步时就中断了训练来跑测试，然而在调试集上的输出结果全部是<unk><unk><unk>。笔者尝试查看了损失函数图像，如下图所示：

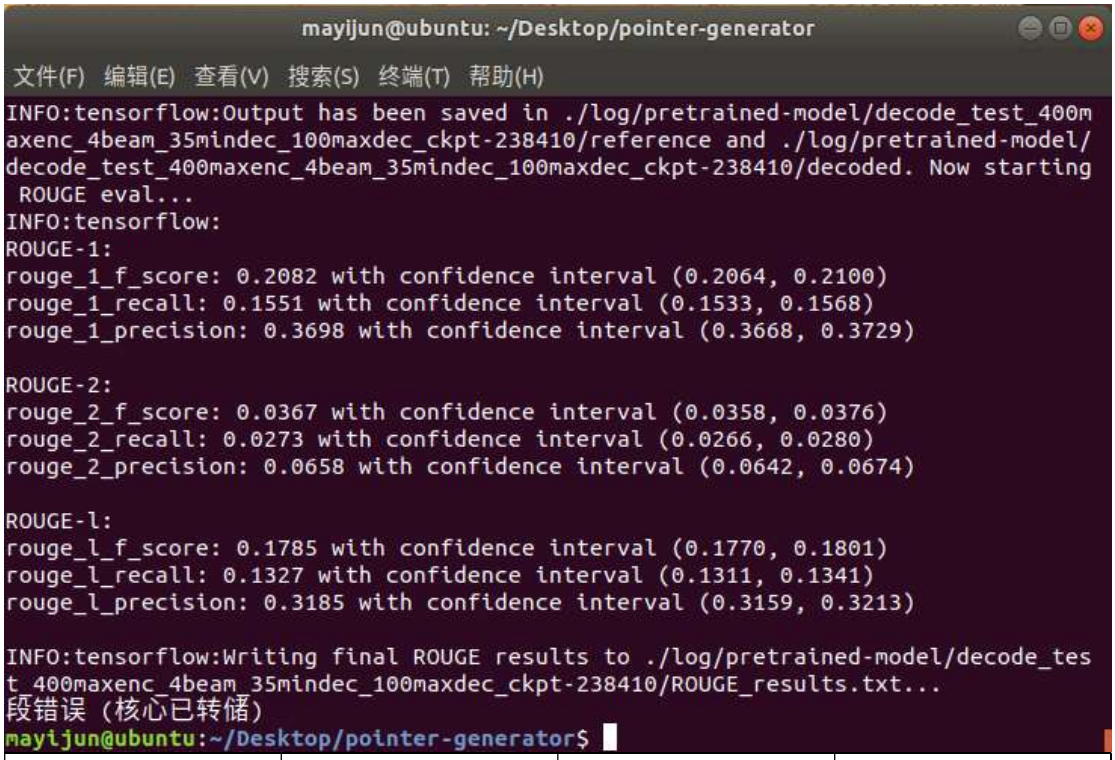


可见损失函数的下降速度很慢，且出现了相当大的波动。鉴于这样的实验结果并不符合我对深度学习模型的认知（一开始应快速下降，后期趋于平稳），我在课程群发出求助，得到的回应是“抖很正常”“应该只是因为数据点采得太密了”“batch_size 调大应该就不这么抖了”。我还尝试询问“把学习率调高有用吗？现在是 0.15”，有人说“因模型而异，我觉得 0.15 的学习率一般来说已经是一个大得可怕的模型了”。这样，我们决定不更改设置，继续训练这个模型。

训练至一万步时，输出结果不再是<unk><unk><unk>了，稍稍有了一些起色，在预测结果的开头已经可以产生一些通顺的语句片段了，如下图所示。



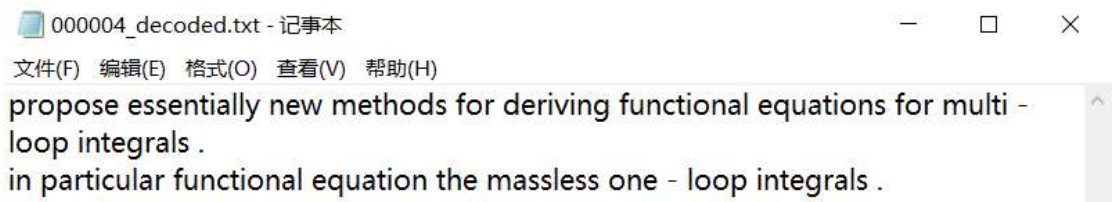
因为论文[6]中声称需要训练 25 万步才可以达到文中的实验结果，折算下来在笔者的个人电脑上需要训练 25 天的时间，遂放弃了继续训练。为了复现这一模型的实验结果，笔者下载了预训练模型；然而作者在 github 上并未提供预训练结果，因此只能使用其原型 Pointer-Generator[8]的预训练结果。

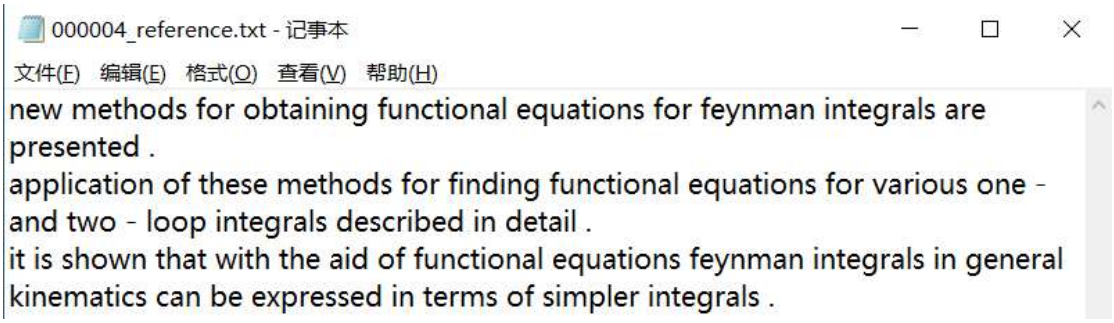


数据集	ROUGE-1 得分	ROUGE-2 得分	ROUGE-l 得分
Arxiv-Dataset	20.64	3.58	17.70

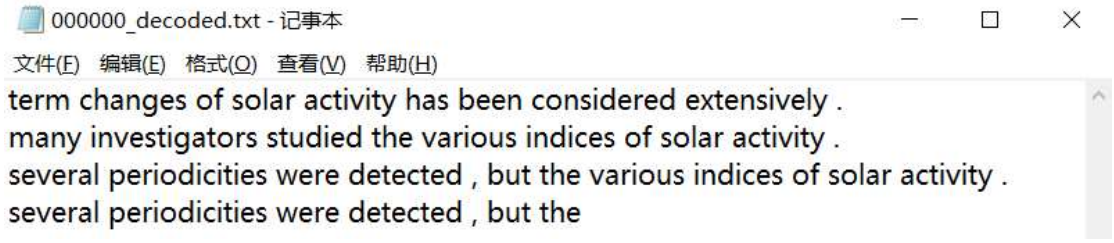
这一测试结果与[6]中声称的测试结果有较大差距，但也在意料之中，因为我们并没能找到[6]的预训练模型，是用[8]的预训练模型代替。

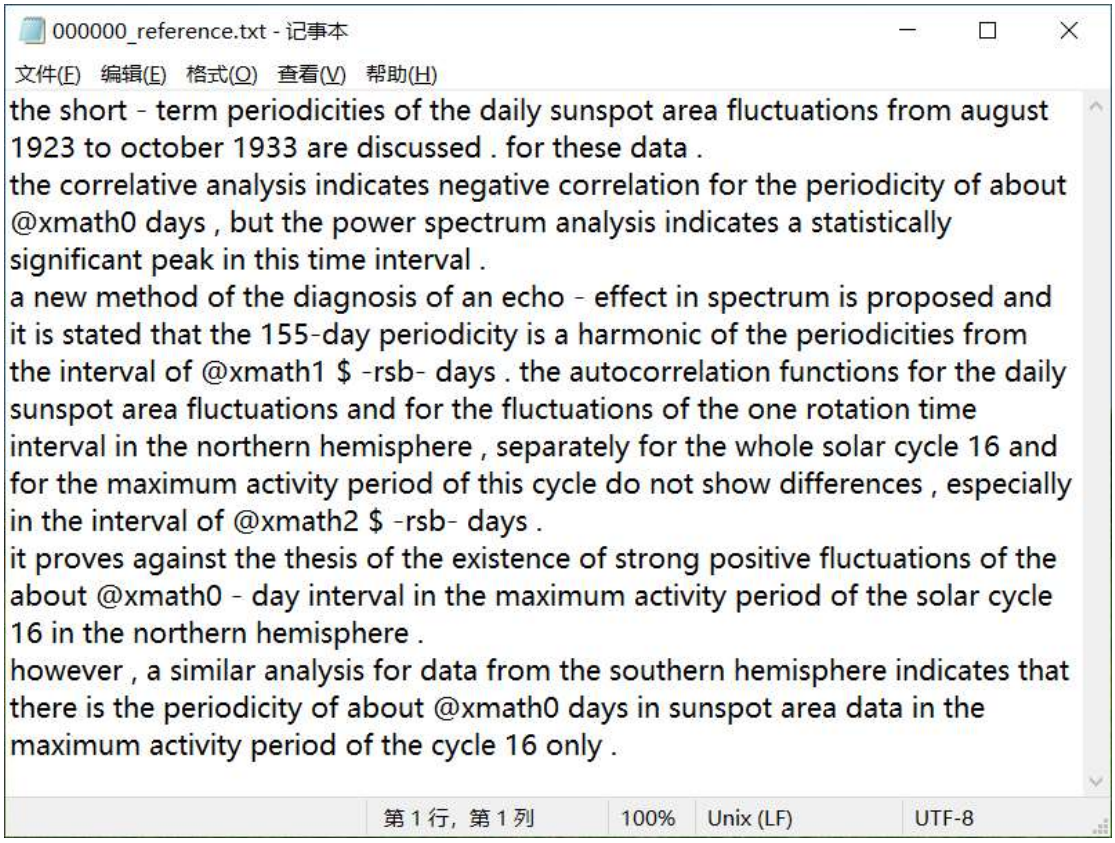
接下来我们来看测试结果中的一些典型案例：





在部分测试点中（如上所示），模型输出的摘要基本通顺、表意基本完整，能涵盖文章主旨大意，但相比原作者的摘要(gold standard)缺少一些要点。





而在更多的测试点中（如上所示），模型产生的输出并不能涵盖文章的主旨大意，也出现了前面提到过的重复现象。(但是因为我们使用的是[8]的预训练模型，[6]声称其针对该问题进行了一些改进，可以期望我们的模型[6]在这个问题上有所改善。)

感性地说，深度学习方法生成摘要的效果只能用“差强人意”来形容。我们的模型输出效果并不见佳，但 ROUGE 得分也不低，说明摘要生成还是自然语言处理中一项相当有难度的工作。

4 非深度学习方法：基于朴素贝叶斯分类器的修辞方法

4.1 原理与实现

作为对比，接下来我们讨论一项非深度学习的解法：基于朴素贝叶斯分类器的修辞方法。[10][11]这是一种抽取式生成摘要的方法。

该方法的原理是，首先人工标注测试数据（学术论文），将句子区分成以下几类。

目的(aim)：本论文的具体研究目标

结构文本(textual)：作为当前论文结构的参考

自有(own)：描述自己的工作：方法，结果，讨论

背景(background)：科学背景知识

对比(contrast)：与其他作品对比的陈述、其他工作的弱点

依据(basis)：与其他工作的统一性或在其他工作基础上的延续

其他(other)：研究人员工作的其他说明（中性）

在以上分类的基础上，就可以将该论文中的句子进行论据划分(argumentative zoning)，句子的功能可以分为位置（文本中句子出现的位置）、章节结构（句子出现在章节中的位置）、段落结构（句子出现在段落的开头、中间还是结尾）、标题、长度、TF-IDF 分数（该分数说明句子是否包含重点字词）、语音、时态、情态等方面。

在得到句子的这些数据以后，就可以根据这些数据来将句子分成摘要和非摘要两类，将属于摘要的句子输出就得到了答案。

4.2 实验结果

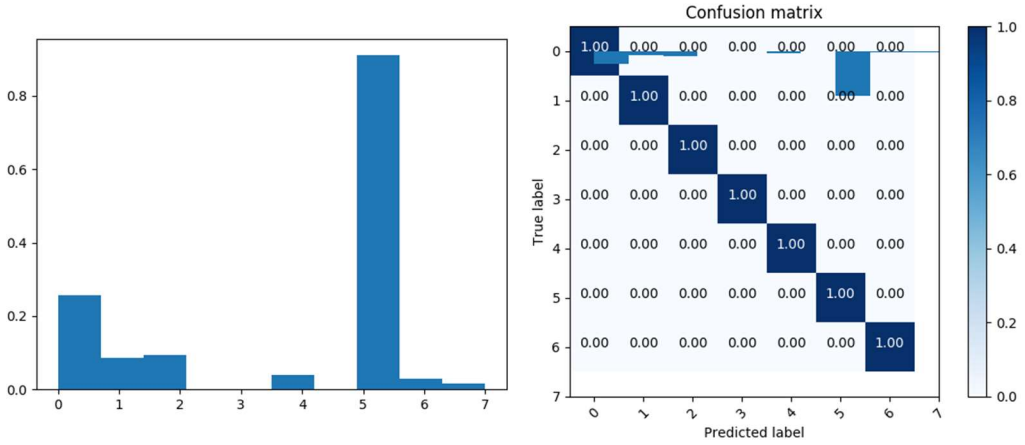
本贝叶斯分类器内置的预训练模型的测试结果如右图所示。表格形式如下：

```
mayijun@ubuntu:~/Desktop/ire$ python src/
=======Bernouli Distribution=======
Data split between train and test: 0.8
Train dataset:  64
Mislabelled sentences: 1708 out of 10286
Train Accuracy: 83.39490569706398

Test dataset length:  15
Mislabelled sentences: 281 out of 2315
Test Accuracy: 87.86177105831533

=======Multinomial Distribution=======
Data split between train and test: 0.8
Train dataset:  64
Mislabelled sentences: 2166 out of 10119
Train Accuracy: 78.59472279869553

Test dataset length:  15
Mislabelled sentences: 451 out of 2596
Test Accuracy: 82.62711864406779

=======Complement Distribution=======
Data split between train and test: 0.8
Train dataset:  64
Mislabelled sentences: 1826 out of 9911
Train Accuracy: 81.57602663706992

Test dataset length:  15
Mislabelled sentences: 445 out of 2605
Test Accuracy: 82.91746641074856

=======Gaussian Distribution=======
Data split between train and test: 0.8
Train dataset:  64
Mislabelled sentences: 0 out of 10373
Train Accuracy: 100.0

Test dataset length:  15
Mislabelled sentences: 0 out of 2296
Test Accuracy: 100.0
Test dataset length:  15
Mislabelled sentences: 0 out of 2296
Test Accuracy: 100.0
```

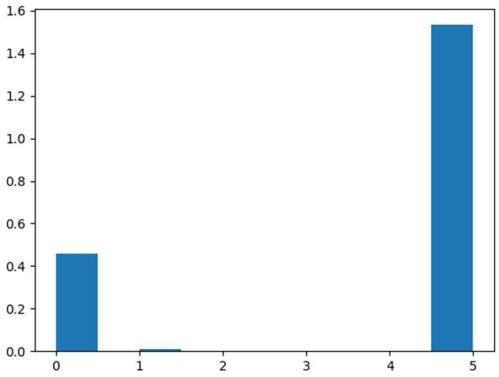
分布	准确率
伯努利分布	87.86%
高斯分布	100.0%
多项分布	82.63%
补体分布	82.92%

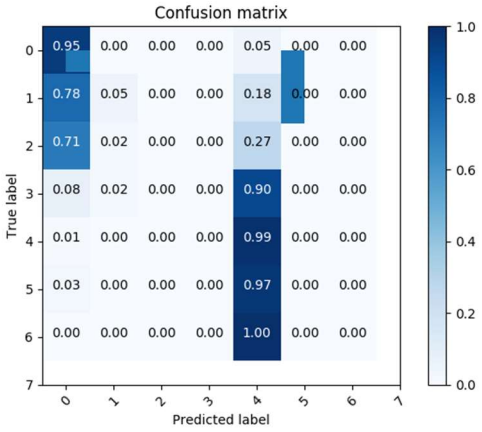
调用 matplotlib.pyplot 和 sklearn.metrics.confusion_matrix 进行可视化，效果如下：

高斯分布：分布图和困惑度矩阵(confusion matrix)

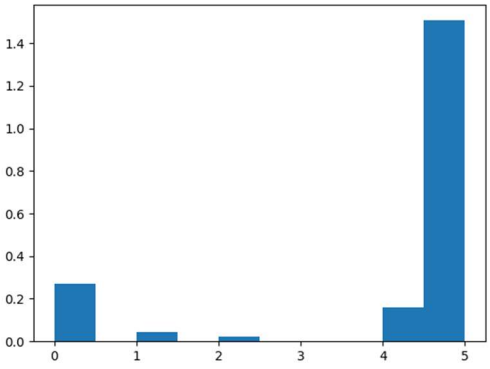


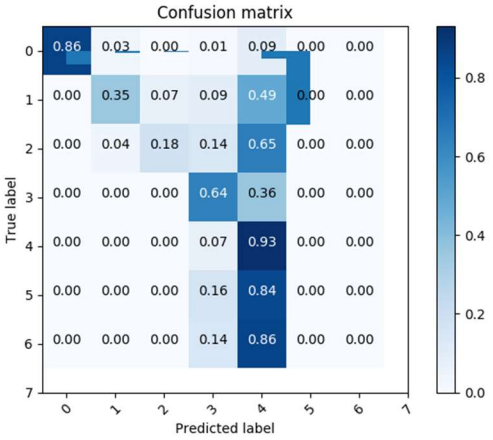
补体分布



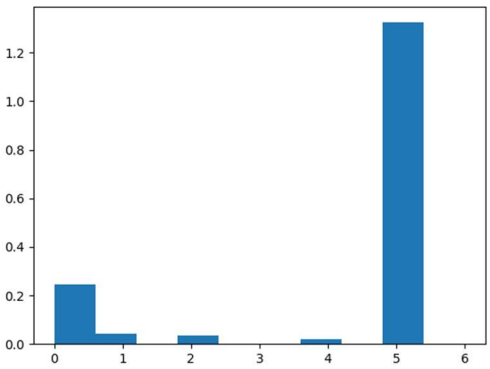


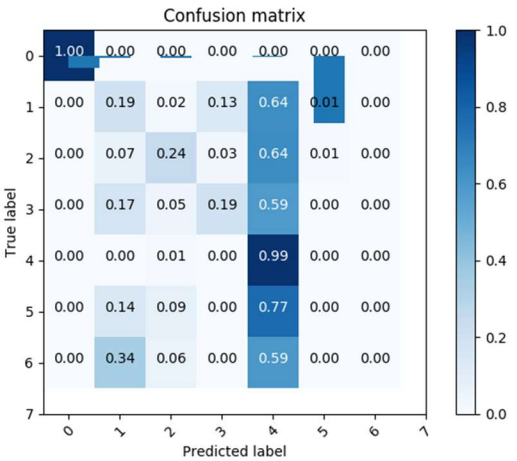
多项分布

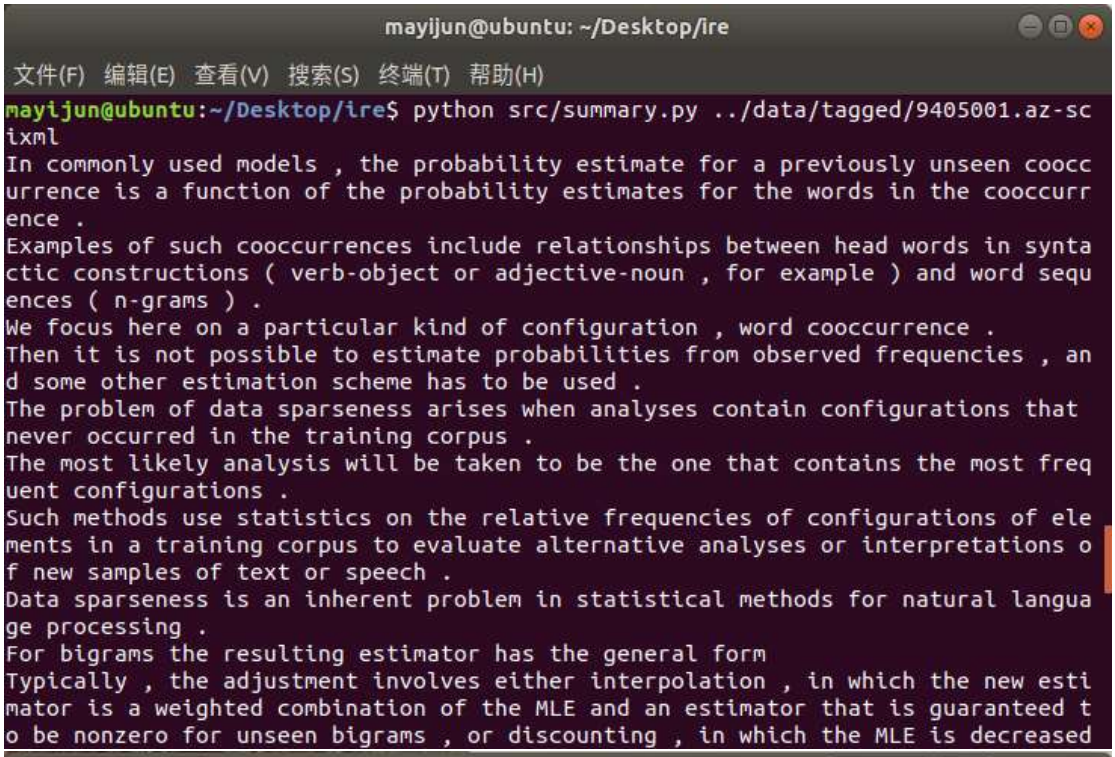


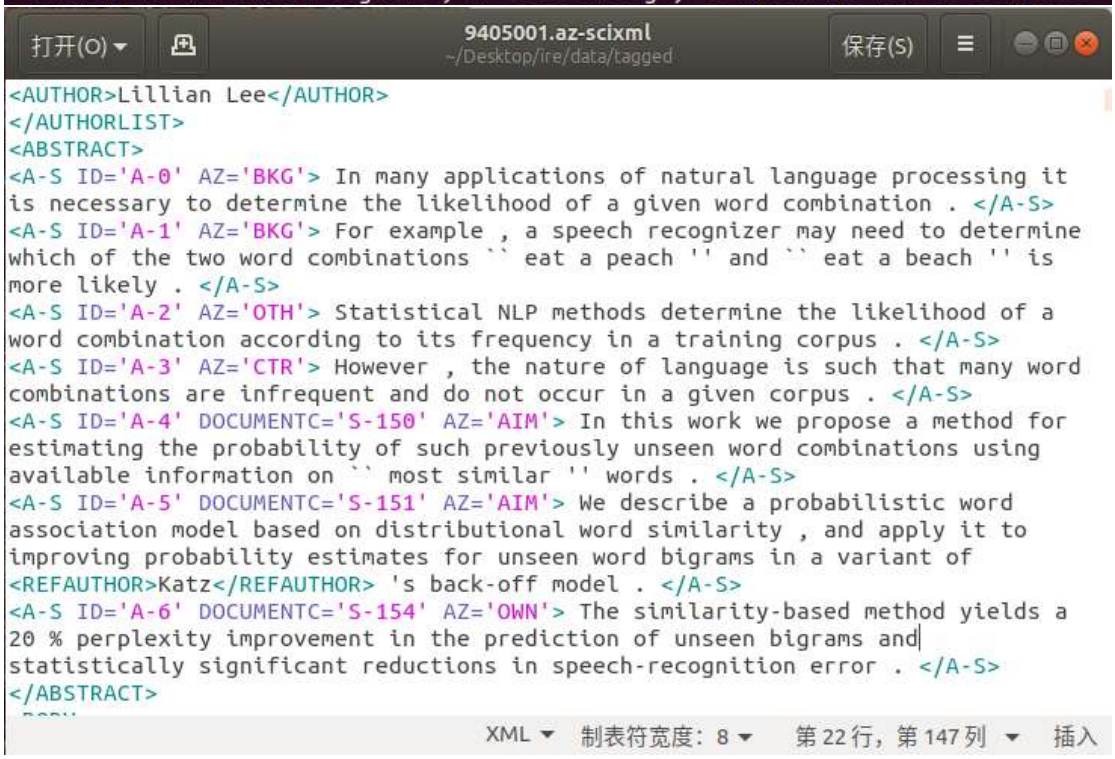


伯努利分布









以上展示了一笔测试数据，上图为计算结果，下图为金标准。

我们可以看到，计算结果总体上是通顺的，没有出现前面深度学习模型的语法不通的输出，更不会出现<unk><unk><unk>这样的输出，由此可以验证抽取式摘要的一个优点：天然的在语法、句法上错误率低[12]。然而，我们的分类器选出的句子不佳，与金标准对比

可见，输出结果并不能很好地作为原文的摘要。

5 讨论与展望

本次课程项目是我第一次真正意义上接触深度学习，因而也出现了各种各样的问题。最大的问题就是，因为对深度学习的流程和时间消耗没有认识，导致**没有预留充足的时间来训练深度模型。**一周左右的时间，扣除因选题、配环境等各种问题消耗掉的时间，最后只有一两天的时间实际用来训练，这对于训练深度模型是远远不足的，最后也没有取得很好的训练效果。这是一个教训，下次做相关项目的时候，我一定会预留充足的时间来训练我的模型，或是引入外部算力资源。

另外，我在做项目期间遇到并处理了真的是各种各样、应有尽有的错误，其中一大部分是**运行环境的问题，花费了大量的时间**（保守估计 2 整天），这也是导致训练时间不足的一个重要原因。包括但不限于 tensorflow 报错 no module named ***、tensorflow 报错 DLL load failed (这两项应该是 tensorflow 版本不匹配造成的)、HDF5 的 version doesn't match、Stanford CoreNLP 报错 java.lang. ClassNotFoundException。此外还有一些粗心大意的错误，我在改别人的 python 代码用来转自己的数据的时候，删除了别人的一个 if 语句，却忘了减小后面代码块的缩进，结果这些代码块就被对应到前一个 if 语句下面，我又花了大约一个小时才调出来这个问题。这些问题要通过多做深度学习、多写代码来解决，做多了就会熟练一些。

另外**本次的选题过程也出现了一些波折。**其实本次 PJ 的选题我在一个月前就开始构思，当时老师提供给我们一些优秀参考选题，包括《基于复旦大学表白墙的文本分析和情书生成》、《为网络聊天信息添加表情》、《歌词分析与语言模型比较——以黄伟文歌词为例》、《让 AI 做英语单选题》、《基于情感分析挖掘影响我国外交关系的重要事件》，之后我就思考了数

个晚上，希望也能找到一个像这样的创意和情怀兼具的选题，但是一直想不出来。我也尝试过集思广益，向周围人广泛地征集选题，也从一位学长那里得到了一个得到老师肯定的高价值选题：现代文和古文的机器翻译；然而，鉴于这一方向的研究成果实在太少，不仅语料库稀少、只能自己制作，而且在 github 上只能找到一些文档和 README 信息残缺的似是而非的模型，我用好不容易制作的语料库试着跑了一个模型，输出乱码，遂放弃了这一选题，改为了最终的这个学术论文生成摘要的选题。这样一来，算上自己制作语料库、为模型配环境，就又花去了一天多的时间。至于感想，只能说一句，好的选题实属难得，灵感这种东西大概是可遇不可求吧。虽然我们是理工科的学生，但平时也要多读书多看报，保持一些对语言文字的接触才好，这带来的增益也是不仅仅限于在我们这门课程上的（或许老师也可以考虑一下培养大家的这个习惯？）。

以上就是对本次课设经历的一些讨论；总而言之，下次还是应该预留更多的时间，虽然这次前后大概有一个星期的时间，但是被选题、配环境、训练深度模型这样一些因素一消耗，时间就非常紧张了，甚至为了能够利用起我睡觉的时间来做训练、跑模型，好几天我都熬夜到 4 点（例如 P2 图片中那些 pdf 文件的修改时间），让模型跑起来了才去睡觉。以后我会**更加合理地安排时间。**

6 总结

本次我们探究了用于学术论文自动生成摘要的两个算法：一种算法属于深度学习方法，使用语篇感知注意力模型，进行生成式摘要；另一种属于非深度学习方法，使用朴素贝叶斯分类器，分析句子的修辞结构。

两种算法的测试成绩都较高，但生成摘要的效果并不见佳：深度学习模型的输出在很多测试点上并不能涵盖文章的主旨大意，甚至输出不通顺的内容，还可能出现重复现象；后者

抽取式摘要面临内容选择错误、灵活性较差的问题。

由此可见，摘要生成尚属于自然语言处理中相对较难的问题，值得人们继续探究。

参考资料

[1] https://github.com/terryum/awesome-deep-learning-papers

awesome-deep-learning-papers: The most cited deep learning papers

[2] https://github.com/floodsung/Deep-Learning-Papers-Reading-Roadmap

Deep-Learning-Papers-Reading-Roadmap: Deep Learning papers reading roadmap for anyone who are eager to learn this amazing tech!

[3] https://github.com/papers-we-love/papers-we-love

papers-we-love: Papers from the computer science community to read and discuss

[4] https://drive.google.com/file/d/1K2kDBTNXS2ikx9xKmi2Fy0Wsc5u_Lls0/view?usp=sharing

arxiv-release.zip – Google 云端硬盘

[5] https://drive.google.com/file/d/1Sa3kip8IE0J1SkMivlgOwq1jBgOnzeny/view?usp=sharing

pubmed-release.zip – Google 云端硬盘

[6] https://arxiv.org/abs/1804.05685

A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui,

Seokhwan Kim, Walter Chang, Nazli Goharian

[7] https://github.com/armancohan/long-summarization

long-summarization: Resources for the NAACL 2018 paper "A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents"

[8] https://github.com/abisee/pointer-generator

pointer-generator: Code for the ACL 2017 paper "Get To The Point: Summarization with Pointer-Generator Networks"

[9] https://arxiv.org/abs/1704.04368

Get To The Point: Summarization with Pointer-Generator Networks Abigail See, Peter J. Liu, Christopher D. Manning

[10] https://www.mitpressjournals.org/doi/pdf/10.1162/089120102762671936

Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status Simone Teufel，Marc Moens

[11] https://github.com/pbteja1998/ire_project_18

ire_project_18: Scientific Paper Summarization

[12] https://www.jiqizhixin.com/articles/2019-03-25-7

机器之心：文本摘要简述

附录

开源数据集 arxiv-dataset：

https://drive.google.com/file/d/1K2kDBTNXS2ikx9xKmi2Fy0Wsc5u_Lls0/view?usp=sharing

开源数据集 pubmed-dataset：

https://drive.google.com/file/d/1Sa3kip8IE0J1SkMivlgOwq1jBgOnzeny/view?usp=sharing

提交稿中的 pdf2txt.py 将 pdf 格式的学术论文中的文本提取成 txt 文件，format.py 用于将 txt 文件转换成 pointer-generator 网络需要的格式。

论文评语（教师填写）：

任课教师签名：

日　　期：