华东师范大学数据学院上机实践报告

课程名称：当代人工智能　　年级：2018　　上机实践成绩:
指导教师：李翔　　姓名：孙秋实
上机实践名称：基于预训练的节点分类　　学号：10185501402　　上机实践日期：2021/12/9

Part 1
实验目的

(1) 使用预训练模型对节点进行分类

(2) 评估与比较不同模型在此任务上的效果

(3) 评估与比较不同输入信息在此任务上的效果

Part 2
实验任务

(1) 设置 Baseline

(2) BERT 类预训练模型介绍与理解

(3) 使用 BERT 类预训练模型完成机器翻译任务

(4) 比较不同输入信息对该任务性能的影响

(5) 模型评估

Part 3
使用环境

(1) Google Colab

(2) Python3.7

(3) Pytorch 1.8

(4) GPU 型号：Tesla K80/V100

Part 4
实验过程

Section 1
Baseline

从文本分类问题出发，朴素贝叶斯法是最为简单直接的，而且其计算开销非常之小。在使用预训练模型前，我先采用朴素贝叶斯算法进行文本分类，将其结果设置为本实验的 baseline，以便和开销巨大的预训练模型进行比较。

	precision	**recall**	**f1-score**	**support**
Label : 0	0.89	0.92	0.91	715
Label : 1	0.83	0.87	0.85	261
Label : 2	0.66	0.51	0.58	169
Label : 3	1.00	1.00	1.00	1255
accuracy			0.93	2400
macro avg	0.85	0.83	0.83	2400
weighted avg	0.92	0.93	0.93	2400

朴素贝叶斯算法的结果仅当作本次实验的 baseline，与使用预训练模型解决问题的关联不大，所以我将其简要过程放在了附录中，其实验调参过程如图 10 和图 11 所示。

BEET 类预训练模型含有大量的参数可以从文本数据中捕捉到更详细的词法、句法结构、现实知识等信息，理论上其在微调后的性能应该能显著高于朴素贝叶斯算法，因此我将该实验的 baseline 设置为调参后的朴素贝叶斯算法最优结果 0.93

Section 2
BERT 类模型概览

Subsection 1
（自）注意力机制

自注意力，即将词元序列输入注意力池化中，在注意力机制的基础上让同一组词元同时充当 Q(Query), K(Key), V(Value)

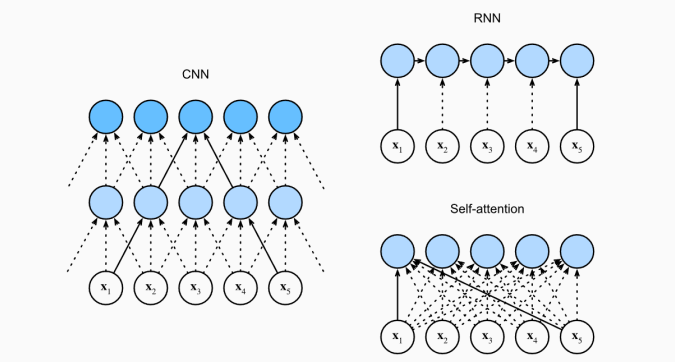

图 1: CNN vs RNN vs Self-Attention

以往我们使用卷积神经网络（CNN）或循环神经网络（RNN）对序列进行编码，目标都是将由词元组成的序列映射到另一个长度相等的序列，其中的每个输入词元或输出词元都由 d 维向量表示。具体来说，我们将比较的是卷积神经网络、循环神经网络和自注意力这几个架构的计算复杂性、顺序操作和最大路径长度。

	CNN	RNN	Self-Attention
复杂度	$O(knd^2)$	$O(nd^2)$	$O(n^2\ d)$
并行度	O(n)	O(1)	O(n)
最长路径	O(n/k)	O(n)	O(1)

表 1: CNN vs RNN vs Self-Attention

需要注意

- RNN 为顺序操作，会严重影响并行度
- 路径越短，就能更轻松地学习序列中的依赖关系

因此，在算力允许的情况下，我们显然选择自注意力机制用于对文本序列的编码。

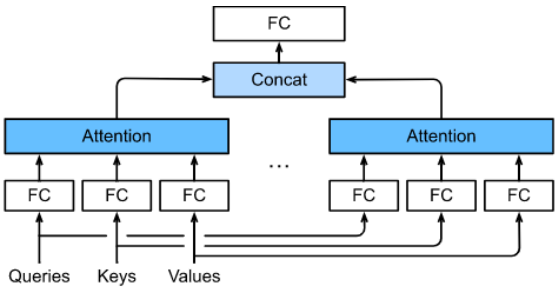

图 2: Multi-Head Attention

在 Transformer 模型中使用了多头注意力机制，它允许注意力机制学习多组 Q,K,V 的线性投影，最后将其组合作为输出这会增加模型计算的开销，但是对模型的性能显然是有益的。在 Transformer 中，每一个自注意力计算的输出都被称作一个 “头”。

Subsection 2
Transformer

Transformer 是 Attention is all you need 提出的模型结构，如图 7 所示，Transformer 作为 Encoder-Decoder 结构的一个实例，Transformer 的编码器和解码器是基于自注意力的模块叠加而成的，源（输入）序列和目标（输出）序列的嵌入（embedding）表示将加上位置编码（positional encoding），再分别输入到编码器和解码器中。Transformer 模型完全基于注意力机制，没有任何卷积层或循环神经网络层。

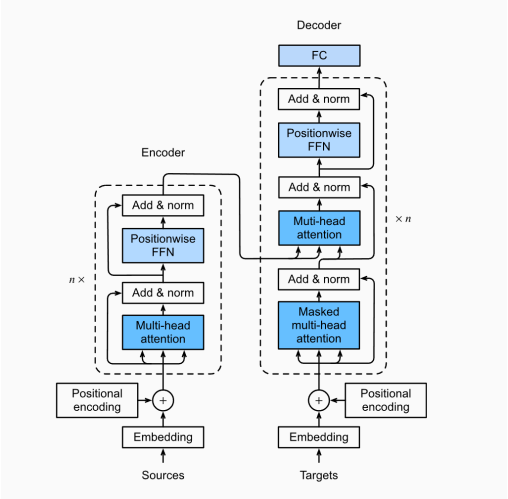

图 3: The Transformer Architecture

在 Transformer 结构中：

- 每个层都有两个子层（子层表示为 Sublayer ）
  - Layer1: 多头自注意力
  - Layer2: Positionwise feed-forward network（其实就是前馈神经网络）
- 子层之间采用残差链接，然后使用 Layer Norm

同样的，解码器也是由多个相同的层叠加而成的，并且层中使用了类似的残差连接和层归一化。除了编码器中描述的两个子层之外，解码器还在这两个子层之间插入了第三个子层，称之编码器 - 解码器注意力（encoder-decoder attention）层。在这个编码器 - 解码器注意力中，Q 来自前一个解码器层的输出，而 K 和 V 来自整个编码器的输出。

更多关于 Transformer 的内容可以参考：The Annotated Transformer

Subsection 3
Pre-training BERT

BERT 是基于 Transformer 的改进工作，原文中的实现完全照搬了 Transformer 的工程实现，摘要中对其描述为“conceptually simple and empirically powerful”。在 BERT 出现之前，面向下游任务的预训练模型可以被分为两种：

- Feature-based model
- Fine-tuning model

其中，Feature-based 的 ELMo 仍然使用了 RNN 架构，而 Fine-tuning model，如 GPT，使用了 unidirectional language models 学习一个具有泛化能力的语言表示。其缺点是，在做 question answering 一类的 token-level tasks 时候，无法考虑到文本的双向信息。

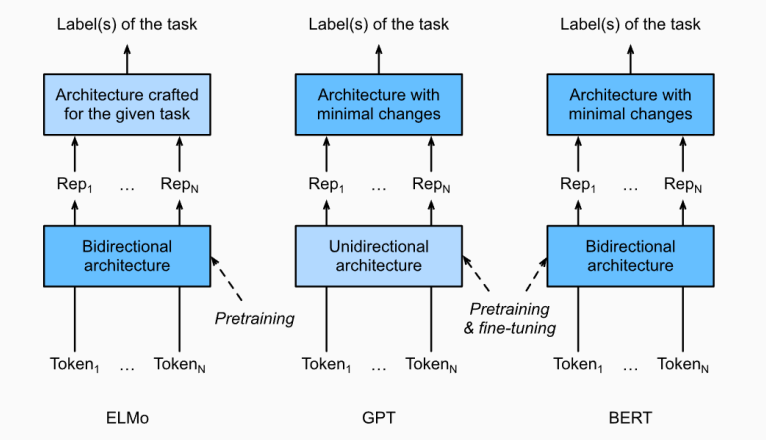

图 4: A Comparison of ELMo, GPT, and BERT

而 BERT，解除了上述工作的单向限制，并受到 Cloze 任务的启发，在预训练阶段使用了遮蔽语言模型。此外，还使用了 NSP(‘Next Sentence Prediction’) 任务来训练文本对。三者的对比如图 4 所示。

对 NSP 任务的解释

以原始论文附录中的一个 case 进行简单解释

正例

$Input = [CLS]\ the\ man\ went\ to\ [MASK]\ store\ [SEP]$

$he\ bought\ a\ gallon\ [MASK]\ milk\ [SEP]$

$Label = IsNext$

负例：

$Input = [CLS]\ the\ man\ [MASK]\ to\ thestore[SEP]$

$penguin\ [MASK]\ are\ flight\ ##less\ birds\ [SEP]$

$Label = NotNext$

可以看到在第二组数据中出现了 $flight\ ##less$，这是因为原始论文使用 WordPiece 处理后将这个不常见词切分为了 subword 导致的。

在预训练阶段，BERT 使用了两个相当大的数据源进行训练

- 800M words 的 BooksCorpus
- 2,500M words 的 English Wikipedia

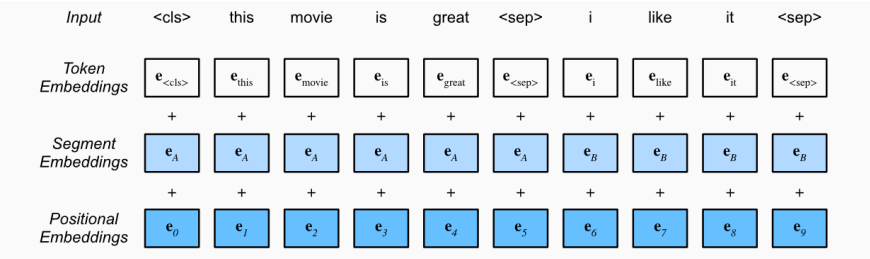

图 5: The Embeddings of the BERT Input Sequence

BERT 的输入表示可以被分为三个部分，如图 5 所示：

- Token Embeddings，可以将其理解为词源
- Segment Embeddings，可以将其理解为“文本中的第几句话”
- Position Embeddings，可以将其理解为当前词在文本中的位置

而 BERT 的输入表示，就是这三个 Embedding 的相加。

Subsection 4
Fine-tuning BERT

在已有预训练好的 BERT 时，就可以将其部署在下游任务上，值得一提的是，对于每一个下游任务，都需要启用一个独立的 BERT 模型，加载好权重后分别进行训练。

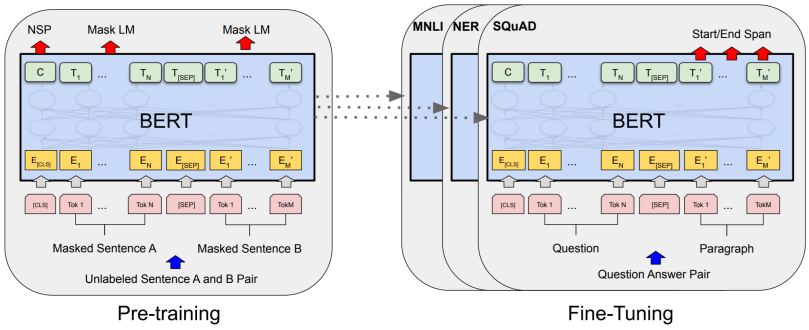

图 6: Overall Pre-Training and Fine-Tuning Procedures for BERT

在该微调阶段，对数据集的预处理为

- Normalization
- SentencePiece

Remark：SentencePiece 是一种分词的方法，在 BERT 原文中使用了 WordPiece，其目的为通过将不常见词切分为 subword 来压缩词典的大小，而 SentencePiece 是一个和它类似的 Google 开源工具，把英语单词切分更小的语义单元（grams），缩小词典大小，具体可参考 Google-SentencePiece

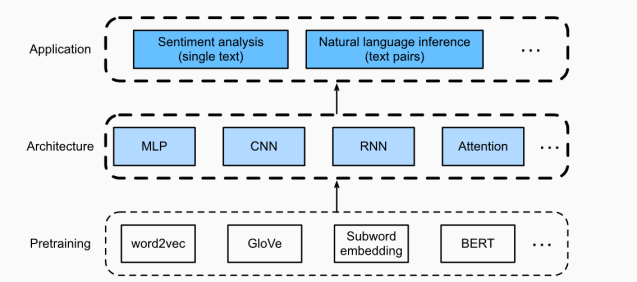

图 7: Different Downstream NLP Tasks

Section 3
模型的输入

对于这个节点分类任务，我设置了三种输入形式：

- 仅使用 Text 作为输入
- 使用 Name+Text 作为输入
- 抽取 Text 的关键词，使用 Name+Keyword+Text 作为输入

其形式如下所示：

图 8: Concate Name & Text as Input

使用 jieba 库对每条文本进行关键词抽取，将抽取的关键词插入输入文本，形式如图 9 所示

图 9: Concate Name, Keyword, Text as Input

Section 4
模型选择

本次实验使用了三个 BERT 类预训练模型，分别为

- BERT-base[1]

即原始的 BERT 模型

- RobBERTa-base[2]

[1]https://huggingface.co/bert-base-uncased
[2]https://huggingface.co/roberta-base

RoBERTa: A Robustly Optimized BERT Pretraining Approach，意思就是一个通过修改预训练阶段的方法和参数，调到最优的 BERT 模型。

Roberta 在如下几个方面对 Bert 进行了调优：

(1) Masking 策略（动态的方式进行 masking）

(2) 模型输入格式与 Next Sentence Prediction

(3) 更大的 Batch

(4) 更大语料与更长的 training steps

具体的训练细节可以参考这篇知乎文章：Roberta: BERT 调优

- Distilbert-base[3]

Geoffrey Hinton 在 NIPS2014 上提出了知识蒸馏（Knowledge Distillation）的概念，旨在把一个大模型或者多个模型 ensemble 学到的知识迁移到另一个轻量级单模型上，方便部署。简单的说就是用小模型去学习大模型的预测结果，而不是直接学习训练集中的 label。在蒸馏的过程中，我们将原始大模型称为教师模型（teacher），新的小模型称为学生模型（student），训练集中的标签称为 hard label，教师模型预测的概率输出为 soft label，模型蒸馏的核心思想是：好模型的目标不是拟合训练数据，而是学习如何泛化到新的数据。所以蒸馏的目标是让学生模型学习到教师模型的泛化能力，理论上得到的结果会比单纯拟合训练数据的学生模型要好。

在 DistilBERT 的原文中提及了 *we leverage knowledge distillation during the pre-training phase and show that it is possible to reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster.* 即 HuggingFace 提出的 DistillBERT，在预训练阶段进行蒸馏。将尺寸减小了 40%，速度提升 60%.

Section 5
性能评估

考虑到 Colab 和 Kaggle Kernel 的算力有限，本次实验选取了 Name, Keyword, Text 组合，以及 Name, Text 组合这两种不同的输入方式，经过训练，这两种输入组合方式对验证集的准确度没有显著的影响，个人认为是 BERT 模型本身对文本的词法、句法结构、现实知识已经能完全掌握，不再需要额外的信息来提升其对文本的理解能力。

以下为分别在三个模型上微调 1 个 epoch 的实验结果：

	Evaluation Loss	Accuracy	Time cost
Naive bayes $_{\text{BASELINE}}$	−−	0.930	−−
BERT $_{\text{BASE}}$	0.0353	0.989	44min
RoBERTa $_{\text{BASE}}$	0.0349	0.992	44min
Distilbert $_{\text{BASE}}$	0.0356	0.987	19min

最后以 Name, Keyword, Text 三者的结合为输入，并选取微调后的 RoBERTa 模型（epoch=3），获得如下验证集结果

- Accuracy = 0.9921
- Evaluation loss = 0.0351
- F1-Score = 0.9920

* 训练所耗费的时间为 130min.

[3]https://huggingface.co/distilbert-base-uncased

Part 6
Reference

参考文献（链接）:

- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
- RoBERTa: A Robustly Optimized BERT Pretraining Approach
- DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter
- The Annotated Transformer

Part 7
附录

以下为设置朴素贝叶斯算法为 Baseline 时进行交叉验证时使用的可视化。

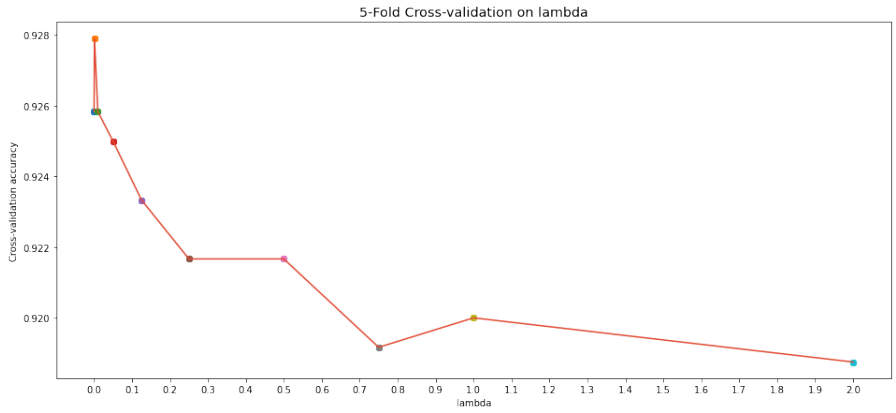

图 10: Cross Validation Curve

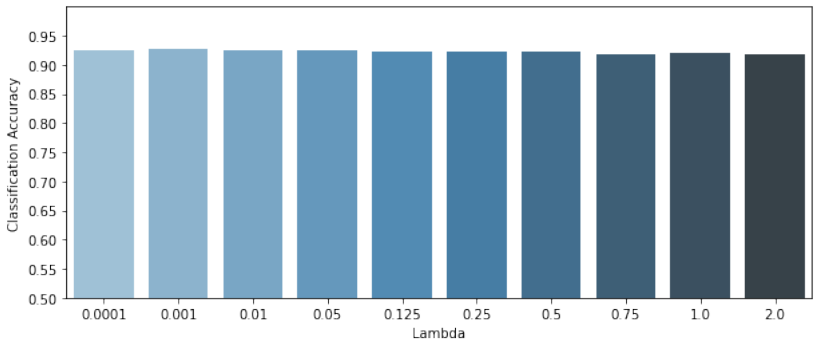

图 11: Comparison: Cross Validation