机器学习大作业——问题摘要生成

|  |  |  |
| --- | --- | --- |
| 学号 | 姓名 | 分工 |
| ZY2006160 | 王宇翔 |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[机器学习大作业——问题摘要生成 1](#_Toc59112466)

[1 题目分析与调研 2](#_Toc59112467)

[1.1 题目分析 2](#_Toc59112468)

[1.2 调研结果 2](#_Toc59112469)

[2 模型原理说明 3](#_Toc59112470)

[2.1 GPT2简介 3](#_Toc59112471)

[2.2 Transformer语言建模 3](#_Toc59112472)

[2.3 模型方法及原理 3](#_Toc59112473)

[2.4 应用场景 3](#_Toc59112474)

[3 模型结构说明 4](#_Toc59112475)

[3.1 模型架构 4](#_Toc59112476)

[3.2 预训练模型 4](#_Toc59112477)

[3.3 迁移学习 4](#_Toc59112478)

[3.4 项目代码结构 5](#_Toc59112479)

[4 调参训练过程及中间结果分析 6](#_Toc59112480)

[4.1 训练环境说明 6](#_Toc59112481)

[4.2 训练集数据处理 6](#_Toc59112482)

[4.3 参数调整及中间结果分析 6](#_Toc59112483)

[5 模型评估与问题预测 7](#_Toc59112484)

[5.1 模型评估 7](#_Toc59112485)

[5.2 问题预测 7](#_Toc59112486)

[5.3 不足与改进 7](#_Toc59112487)

[6 比赛工作站成绩截图 8](#_Toc59112488)

## 1 题目分析与调研

本节主要对大作业——问题摘要生成进行题目分析，针对分析结果进行调研学习，并选择合适的模型来解决问题。

### 1.1 题目分析

（什么是问题摘要生成，要达到什么效果，分析给定数据）

### 1.2 调研结果

（抽取式：TextRank；生成式：Seq2Seq+Attention，GPT2）

<https://blog.csdn.net/hlang8160/article/details/88310815>

<https://blog.csdn.net/qq_41853758/article/details/82859654>

## 2 模型原理说明

本节主要对选定模型——GPT2进行原理分析与说明，从GPT2模型简介、Transformer语言建模、模型方法及原理和应用场景四方面展开说明。

<https://www.cnblogs.com/zhongzhaoxie/p/13064404.html#212-transformer%E8%AF%AD%E8%A8%80%E5%BB%BA%E6%A8%A1> – 仅参考

### 2.1 GPT2简介

（Introduction，为什么提出、作比较、使用方法简介、效果简介等）

### 2.2 Transformer语言建模

（语言模型、Transformer语言模型）

### 2.3 模型方法及原理

（输入编码、神经网络、注意力机制、输出编码等）

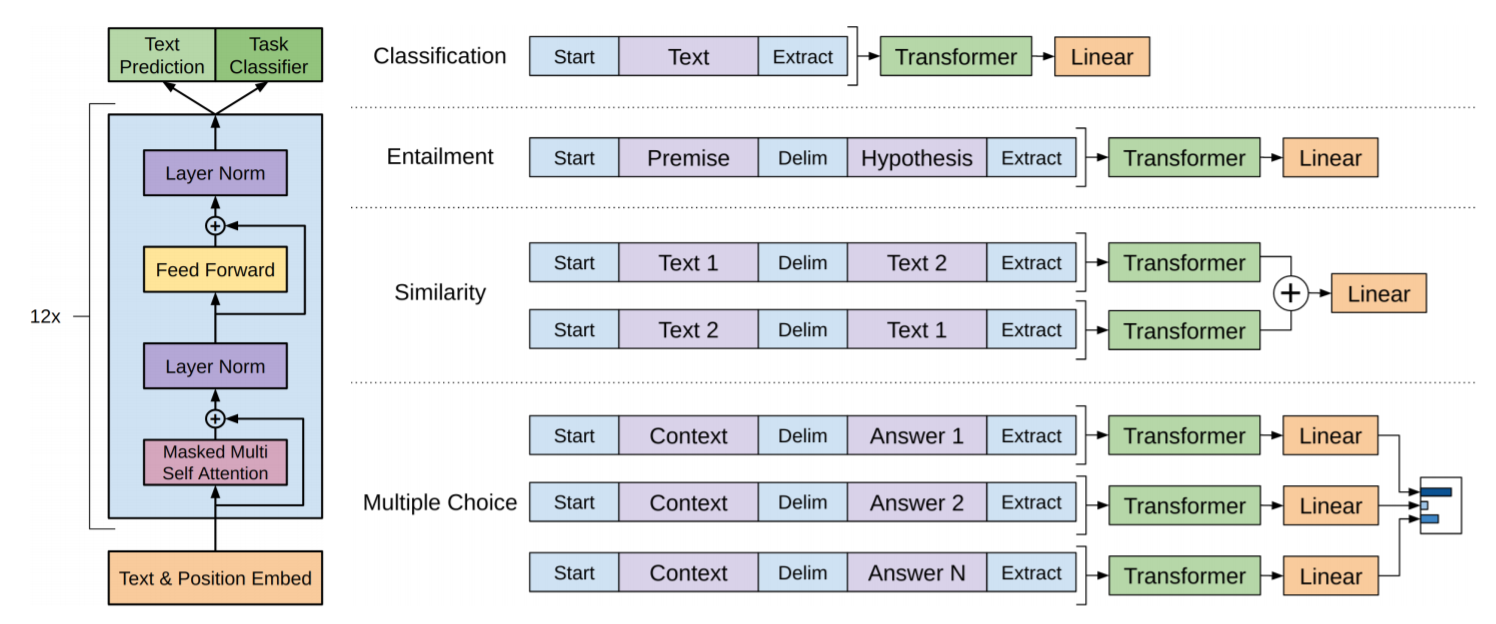
### 2.4 应用场景

（机器翻译、自动摘要生成、音乐生成等）

## 3 模型结构说明

本节主要对GPT2的模型结构进行说明，主要从模型架构、预训练模型、迁移学习和项目代码结构四部分展开说明。

### 3.1 模型架构



（对模型架构进行详细说明。）

### 3.2 预训练模型

（unsupervised pre-training，预训练原理介绍及作用）

### 3.3 迁移学习

（supervised fine-tuning，针对特定应用进行进一步训练）

### 3.4 项目代码结构



（写一个大致的代码结构说明）

## 4 调参训练过程及中间结果分析

本节主要对具体训练过程进行详细介绍，主要从训练环境说明、训练集数据处理、参数调整及中间结果分析三个方面展开介绍。

### 4.1 训练环境说明

操作系统：

Ubuntu 20.04.1 LTS

硬件环境：

CPU架构： x86\_64

CPU 运行模式 64-bit

CPU-Cores: 16

型号名称： AMD Ryzen 7 5800X 8-Core Processor

CPU MHz： 2051.057

CPU 最大 MHz： 3800.0000

内存： 16GB/2 channels

内存频率MHz： 3200

显卡型号： RTX 3070

显存： 8GB

运行环境：

conda 4.9.2

python 3.6

transformers

sklearn

tqdm

numpy

scipy

scikit-learn

tensorflow

boto3

sacremoses

torch

### 4.2 训练集数据处理

（SUMM\_TRAIN.csv，划分训练和测试集、文本数据处理、规范化数据（拼接）等）

数据预处理阶段，将原始训练数据(.csv)格式读入后，进行标签和原始文本拆分，形成如{“summarization”: “...”, “article”: “...”}格式，并在原始文本和摘要文本之间加入分隔符，形如[CLS]article[SEP]summarization[SEP]的格式。之后对文本序列化，参照vocab\_small.txt进行文本-向量转换，该文件包含所有ascii字符、简体中文汉字、中文标点符号、繁体汉字以及用于分割文本的[SEP]、[CLS]标识。数据预处理过程如下图所示。

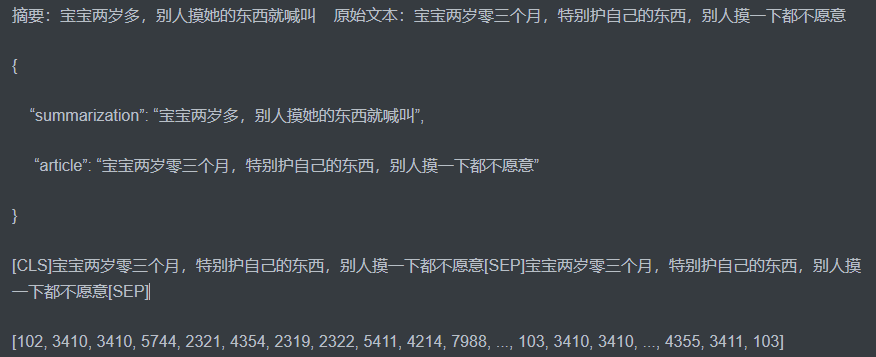


图 训练集数据预处理过程

将数据按照8:2划分训练集和测试集，每轮训练过程都将随机划分，另外，由于模型大小受限，因此对于大于输入窗口n\_ctx大小的文本向量进行截断处理，该模型窗口大小设定为300，对于其中99%的数据，能够完整输入到模型中训练。

对于训练参数的设置，其中，--epochs训练轮次设置为10，--batch\_size训练单批次大小设置为8（该参数理论上设置在6~12较为合理，且越大越有利于提升模型准确率，由于显存大小首先，这里最大只能设置为8），--lr学习率初始值为1.5\*10-4，且可以动态调整大小，--warm\_up步数设值为2000，--log\_step汇报loss步数设值20，--gradient\_accumulation梯度积累设值1（如果显存大小受限，则相应增大该参数大小）。

### 4.3 参数设置及中间结果分析

--device：设置是否使用显卡，默认值0。

--model-config：模型配置参数路径。

--vocab-path：词库路径。

--train-raw-path：原始文本存放路径。

--train\_tokenized\_path：tokenize之后数据存放路径

--log-path：训练日志存放位置

--epochs：训练轮次，本实验设值为10。

--batch\_size：批次大小，本实验设值为8，（该参数理论上设置在6~12较为合理，且越大越有利于提升模型准确率，由于显存大小首先，这里最大只能设置为8）

--lr：学习率大小，本实验设值为1.5\*10-4，且可以动态调整大小，与warmup配合

--warmup\_steps：预热步数，本实验设值为2000，达到该步数后，动态调整学习率大小。

--log\_step：多少布汇报一次loss和acc。

--gradient\_accumulation：梯度累计，本实验设值为1，如果显存大小受限，则相应增大该参数大小，将前后多次梯度合并计算

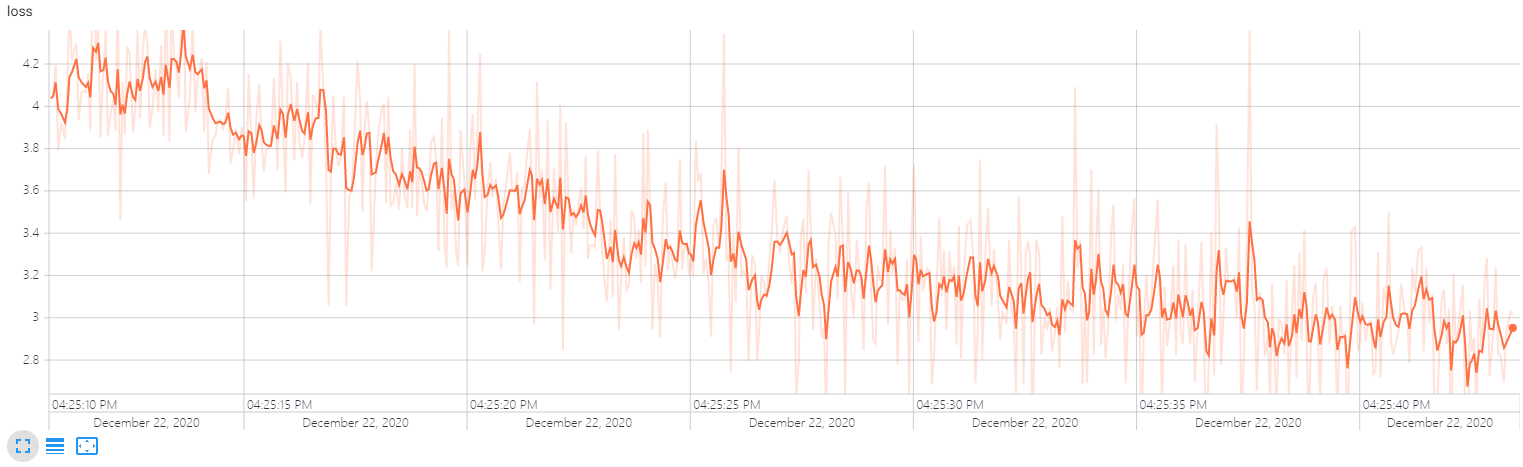
-- dialogue\_model\_output\_path：模型输出路径。

-- pretrained\_model：预训练模型GPT2路径。

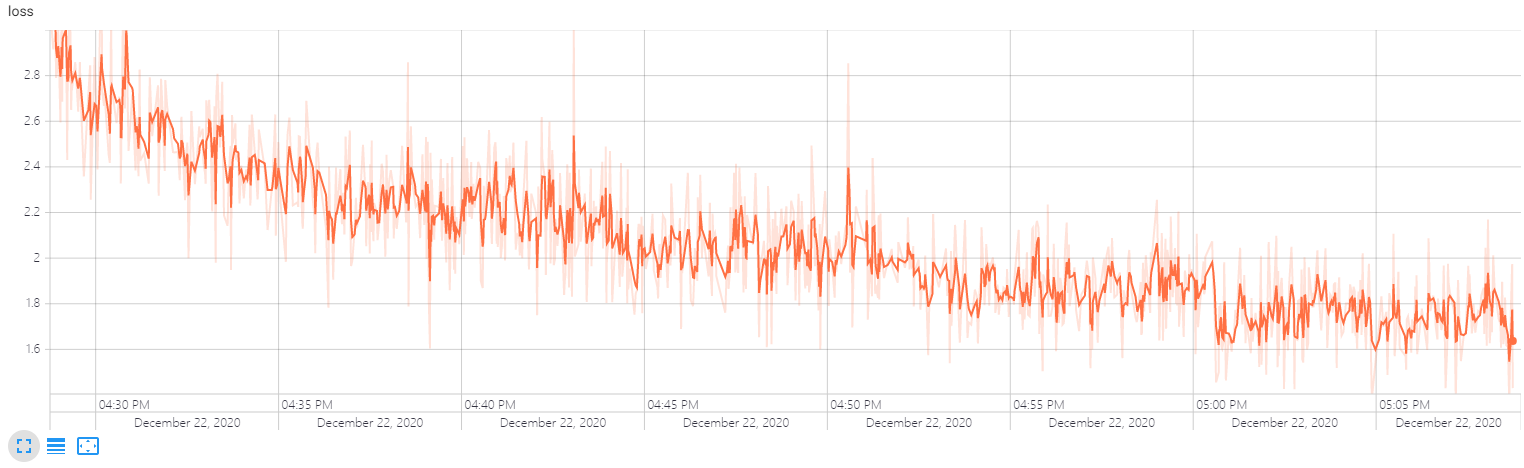
-- writer\_dir：Tensorboard存放路径。

-- seed：随机种子，用于复现模型。

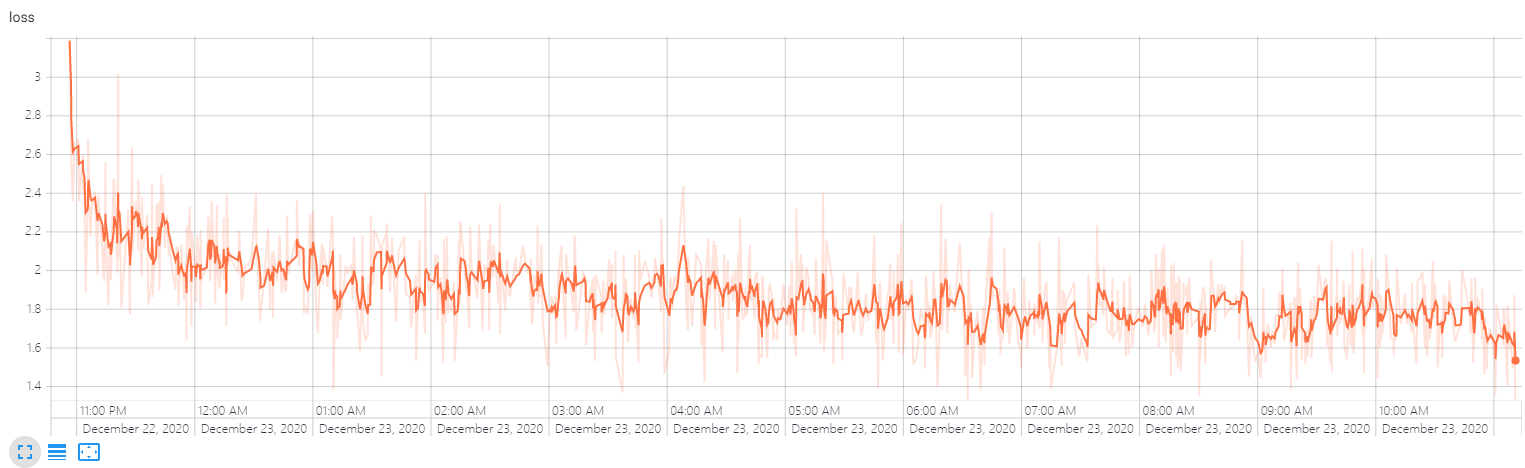
以下是对不同轮次生成的tensorboard的训练结果曲线展示。



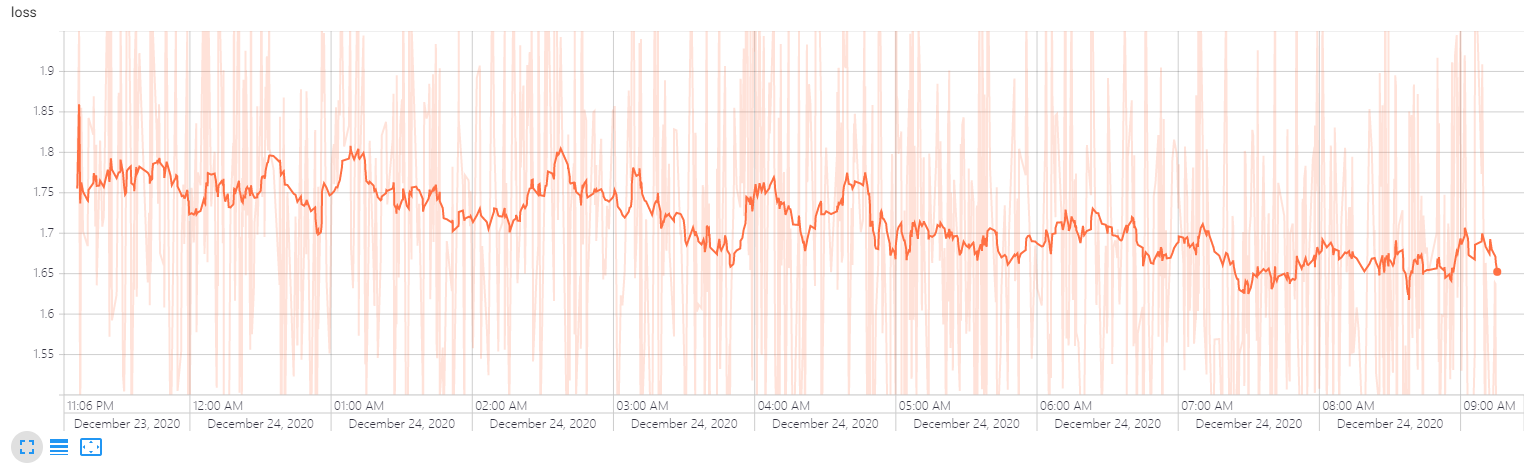
第一轮初始损失值4.295，准确率0.239



第三轮初始损失值3.674，准确率0.382



第六轮初始损失值2.672，准确率0.417



第九轮初始损失值1.853，准确率0.441

数据预测阶段参数设置：

--model-config：模型路径。

--vocab-path：词库位置。

-- repetition\_penalty：惩罚因子，对于已经生成的字符，添加惩罚项，降低其再次出现的概率，该参数可有效避免语句循环生成。

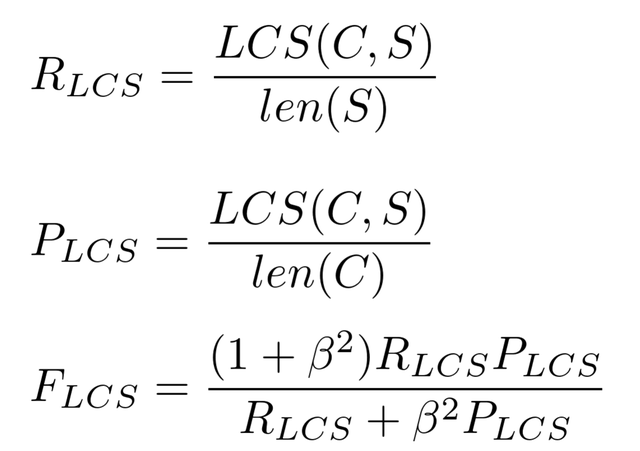
--seed：预测节点随机种子，用于结果复现。

--max\_len：指定最大生成长度，超过最大长度后，进行截断处理。

--output-nums：对于同一原始文本，设置预测次数。

针对数据预测阶段，首先将原始文本进行序列化，预处理过程与训练集预处理过程类似，不同的是[SEP]之间为空。模型将原始输入文本作为输入，并预测output-nums次，每次结果都有所不同。通过对预测出来的摘要文本结合某种算法，选取出其中某一条最为合理的摘要。

由于评价指标采用的是基于ROUGE-L的算法，其中RLCS 表示召回率，PLCS 表示精确率，FLCS 就是 ROUGE-L。通常beta值会设置的比较大，因此 FLCS 更侧重于RLCS (即召回率)。而召回率类似于公共子序列长度占比较字符串长度的比重，根据判定标准，C代表生成摘要，S代表目标摘要。那么在不改变摘要含义的情况下，通过加长摘要长度有助于提升系统得分值。



因此一个最朴素的想法是对生成出来的摘要进行长度限制，通过统计分析得出，长度小于等于5的摘要约占总体的2%，并且长度越小，摘要含义越发的偏离原始文本内涵，因此，通过限制摘要的最小长度，提升了最终得分，约为0.015。

另外，由于没有目标摘要S，因此只能通过分析生成摘要C和原始文本T的相似度，并限制该相似度需大于预先设定的阈值，进而提升系统得分值。常见的相似度判别模型有距离向量、字符串匹配KMP、最长公共子序列，本实验综合算法时间复杂度最终选择了最长公共子序列模型，阈值的设定与生成摘要C的长度相关，如果S和T的最长公共子序列长度所占S长度的比重大于该阈值，那么该摘要则是良好的。

通过结合以上两种改进方案，使得系统得分相较于单使用模型预测的结果得分提升了0.04。

## 5 模型评估与问题预测

本节主要对模型的预期效果进行评估分析，主要从模型评估、问题预测和不足与改进三方面展开说明。

### 5.1 模型评估

（首先说明评估方法）

（其次画一个表，在不同的参数下、不同训练集数据划分下、不同运行环境下的效果进行对比分析，包括准确率、训练时间和模型大小等）

### 5.2 问题预测

（对SUMM\_TEST.csv进行预测和输出，并上传系统分析结果，这里可以放不同参数下训练结果的排名截图，递进）

### 5.3 不足与改进

（可完善的地方）

## 6 比赛工作站成绩截图

（只放最好结果）