基于预训练模型的情感对话分类

1 背景分析

1.1 任务目的

通过历史的对话信息，预测最后一句话的情绪。根据 macro-f1 值进行测评。

1.2 数据分析

数据集包含 8414 条对话数据，每句话都有对应的情绪标签。标签包含六个类别：1：Happiness，2：Love，3：Sorrow，4：Fear，5：Disgust，6：None。

每段对话中包含 n 个句子（大部分是四句）以及对应情绪标签。

2 设计思路

2.1 方案选择

通过对数据以及任务的分析，我初步的想法是可以把每句话以及其标签都单独摘出来，这样这个任务就变成了一个文本分类的形式。

通过预训练模型的课程，这里采用了预训练模型进行训练。预训练模型在本质上，这是一种迁移学习的方法，在自己的目标任务上使用别人训练好的模型。

多方面调研综合考量后，本次情感分析任务上游模型选用的是：bert-base-chinese 以及 Erlangshen-Roberta-110M-Sentiment。

2.2 数据预处理

（一）数据拆分并组合成需要的格式：

1. 训练集：提取每段话并按__eou___符号分开，并将每句话和其对应的标签配对

2. 测试集：提取每段话并按__eou___符号分开，取最后一句话

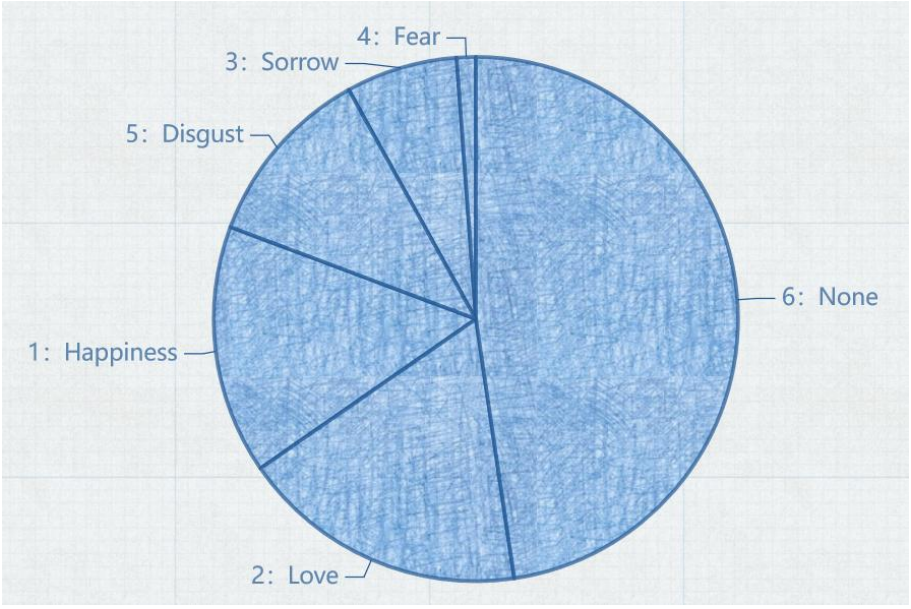
（二）数据过滤

一开始我将数据剔除英文、数字，以及空格，英文标点符号和特殊符号以及中文标点符号，但是发现准确率要比未过滤前低，因为类似：！？....等这些符号在数据中都带有了强烈的情感，所以后面只是选择过滤空格以及特殊的符号，准确率发现有提升。

（三）数据增强

通过对数据集的每个标签数据的展示，发现了标签 4：Fear 极度的缺少，

以及其他标签数据分布不均匀。但是参考网上的增强方法，一般都是等比的扩充，由于时间问题，后续并没有更加深入研究这部分内容。



2.3 模型构建

使用预训练模型+微调的方式进行模型的构建。

分词通过 AutoTokenizer 采用自带的 tokenizer

模型使用：AutoModelForSequenceClassification

在使用模型中会有的输出：

Some weights of the model checkpoint at bert-base-chinese were not used when initializing BertForSequenceClassification: ['cls.predictions.decoder.weight', 'cls.predictions.transform.LayerNorm.weight', 'cls.seq_relationship.bias', 'cls.seq_relationship.weight', 'cls.predictions.transform.dense.weight', 'cls.predictions.bias', 'cls.predictions.transform.dense.bias', 'cls.predictions.transform.LayerNorm.bias']

通过查询这个警告信息，不是报错信息。这只说明对应的加载的预训练模型与任务类型不完全对应。如这个模型的架构是 BertForMaskedLM，因此用 BERT 在别的任务上的 model 类来调用该预训练模型时，要么出现有些参数用不到的情况（如本例），要么出现有些参数没有、需要随机初始化的情况。

2.4 训练参数

本次训练需要调节的超参数有如下几个：

属性	备注
epoch	训练次数
lr	学习率
batch_size	训练批次数量
num_warmup_steps	预热步数

由于设计的网络不太复杂且数据量少，所以通过观察经过三轮训练模型就能达到拟合。

经过多次尝试，最终确定学习率为 2e-5 此处参考了 bert 文档。

对于学习率这块还采用了先线性增加再线性衰减的学习率调整策略，由于刚开始训练的时候，模型的权重是随机初始化的，此时如果选择一个较大的学习率，可能带来模型的不稳定（震荡），选择预热学习率的方式，可以使得开始训练的几个 epoch 或者一些 step 内学习率较小，在预热的小学习率下，模型可以慢慢的趋于稳定，等模型相对稳定后再去选择预先设定的学习率进行训练，使得模型收敛的速度变得更快，模型的效果更好。一般建议是总数量的 10%，但是在几次实际的训练中，找到相对来说较好的 num_warmup_steps 为 100。

多次尝试后得到了最终两个上游模型的超参数：epoch：3，batch_size：32，lr：2e-5，num_warmup_steps：100。

2.5 训练过程

模型训练没有使用 Trainer API 训练，选择自己构建的训练方法，训练中采用的优化器均为 AdamW，准则均为 CrossEntropyLoss。

同时为了得到每次训练的结果，训练过程中加入了每次训练的随机种子，并且我将种子的每一位都赋予都应的含义，方便我们通过随机种子就能看到对应的超参数配置例如：1:第 i 次 2:epoch 3:warmup_steps。

3 思路扩展尝试

3.1 使用更大的上游模型

一开始我用的是 Erlangshen-Roberta-110M-Sentiment 模型，后面尝试使用更大的预训练模型 Erlangshen-Roberta-330M-Sentiment 以及 Erlangshen-MegatronBert-1.3B-Sentiment 但是最终得到的结果并无显著提升，而且占用空间和运行时间都增长，所以综合考量后还是决定使用 Erlangshen-Roberta-110M-Sentiment 模型作为上游模型。

3.2 配置权重

因为上面已经知道初始的数据集中，数据的分配极度不平衡，所以对于数量较少的类别，我们赋予它更多的权重，这样如果网络在预测这些类的标签时出错，就会受到更多的惩罚，对于具有大量的类别，我们赋予它较小的权重。

一般有两种方式（建议第二种）：

第一种，用样本数的倒数当做权重。

第二种，用类别中最大样本数量除以当前类别样本的数量，作为权重系数。

weights = [3.1, 2.7, 6.9, 40.6, 4.2, 1.0]

通过几次测试，发现第一种配置权重后，recall 率有明显的提高，但是准确率也下降了不少，所以最后的 f1 值提升并不明显。但是后来通过这篇博客(https://blog.csdn.net/chumingqian/article/details/126625183)了解到，一般情况下，假如$\mathrm{num}_{\max}$表示数量最多类别的样本个数，$\mathrm{num}_{\min}$表示数量最少类别的样本个数，当的$\frac{\mathrm{num}_{\max}}{\mathrm{num}_{\min}} < 10$ 时候，是不需要考虑样本不平衡问题的。所以最终把权重设置为：weights = [1.0, 1.0, 1.0, 40, 1.0, 1.0]，得到进一步提升的模型

但是我发现第二章权重设置能够较好的保持准确率，这与博客里面的说法相反也比较疑惑，最终的模型相比较于未设置权重 accuracy 准确率下降了 0.15 但是 recall 得到了 0.5 的提高。

3.3 时序模型的使用

一开始我想的是仅仅靠对文本分类去实现这个任务，但是在这里我忽略了一个很关键的信息就是，在此处的背景是对话，既然对话他的上下文就很重要，但是我之前都没有用到上下文文本的信息，仅仅是粗暴的把每句话都单独拎出来进行分析。

由于之前上过自然语言的课程，我想到机器翻译中的时序模型，在机器翻译中是通过前 n-1 个单词预测第 n 个单词，这样我类比到这个任务中，通过前 n-1 个类别标签的语句，预测第 n 个句子的情绪标签。并且采用 GRU 来学习前向关系。

虽然使用时序模型准确率有提高，但是 recall 指标缺比较低。后续通过结合上述几种尝试的模型才得到了比较好的效果。

3.4 模型融合

（一）单模型融合

通过设置 5 个不同轮次的随机种子，得到其中五个最优模型后，在即把同一个模型的多个检查点的参数进行平均。在自己划分的测试集中得到的结果并不理想，除随机初始化参数不一样以外，我觉得各模型版本之间由于每次使用的训练集和测试集都不同所以导致每个模型间的参数权重融合后反而造成了精度的下降。但是在最后发布的验证集上面模型精度是有提升效果的。但是所需时间消耗太大。

由于上面尝试过配置权重去提高模型性能，但是发现精度会有损失，所以通过融合原始权重以及手动配置权重两个模型参数进行平均发现能够在保持精度的同时 recall 也有极大的提升相较于原始未配置权重的模型。

（二）多模型融合

多模型融合即把多个模型最后输出的概率向量进行平均。

第一种两种不同上游模型融合：上游模型有 bert-base-chinese 以及 Erlangshen-Roberta-110M-Sentiment 两个方案，通过对这两个模型最后输出的概率向量进行平均。平均后发现精度反而下降了，而且 recall 评价指标缺大大的下降了，通过两个对比发现 bert 拉低了 Erlangshen-Roberta。其原因可能是 Erlangshen-Roberta 是专门针对情感分析微调过的上游模型，所以表现效果要好于 bert-base-chinese。又去找到一个新的上游模型 roberta-base-finetuned-jd-full-chinese 但是发现召回率还是很低，遂放弃使用。

第二种使用两种不同权重的同一上游预训练模型+时序模型融合：通过上述几种模型单独拿出来发现其在各项指标都有着自己的独特优势，例如：原始预训练模型准确率高，加权重后预训练模型的 recall 较高。而且发现这两种预训练模型与时序模型搭配后精度都有大幅的的提高，并且加权与未加权参数进行平均能够有着不错精度的同时 recall 也有着明显的提高。

4 总结

通过实际项目尝试，让我对预训练模型的使用更加熟悉而且对于深度学习中的每个步骤都有着更深入的理解。并且预训练上游模型的选择也极为重要，本次选用的是 Erlangshen-Roberta-110M-Sentiment 其是针对情感分析微调过的上游模型，综合对比发现相较于其他未微调过的模型各项指标都有着极大的提高。而且在数据量较少的情况下，选择较小参数的模型性价比最高。而且如果能够更好的对原始数据集进行增强的话感觉能够进一步提高模型的性能，这也更让我感觉到除了模型的设计数据的预处理也很重要。

但是在实际的调整超参数的过程中，个人感觉都是在盲目的一个个去尝试并没有很好的头绪，这更让我感觉到自己对于底层掌握的不熟悉，目前只是会简单处理一下数据然后粗暴的调参。感觉用自动化搜索超参算法能够能方便得到更准确的超参数。