PosiGPT:基于预训练的中文积极情感评论模型

卢晨耀[1]　李敏波[1,2]

[1](复旦大学软件学院　上海 200438)

[2](上海市数据科学重点实验室　上海 200438)

摘　要　近年来,越来越多的人因为工作和生活的压力处于抑郁状态,许多人没有得到足够的鼓励和正确的指导。基于这个背景,提出中文评论模型 PosiGPT。基于生成式预训练模型,使用中文微博数据进行训练,可以检测抑郁状态的博文,并产生积极回复。除了该模型外,还发布了 PosiChat 数据集,其包含源自新浪微博的抑郁文本及其积极评论。在 PosiChat 数据集上进行评估,结果表明模型生成的文本具有较强的流畅性和合理性,且在情感倾向上属于积极情感状态,初步达到了抑郁检测及积极回复的功能。

关键词　文本生成　抑郁检测　积极情感评论　预训练

中图分类号　TP391.43　　**文献标志码**　A　　DOI:10.3969/j.issn.1000-386x.2024.09.006

POSIGPT: A CHINESE POSITIVE EMOTIONAL TEXT GENERATION MODEL BASED ON PRE-TRAINING METHOD

Lu Chenyao[1]　Li Minbo[1,2]

[1](*School of Software, Fudan University, Shanghai 200438, China*)

[2](*Shanghai Key Laboratory of Data Science, Shanghai 200438, China*)

Abstract　In recent years, more and more people have been depressed due to the pressure of work and life, and many people have not received enough encouragement and correct guidance in time. Based on this background, this paper proposes a Chinese comment model named PosiGPT. PosiGPT used a generative pre-trained model and used Chinese Weibo data for training. It could detect blogs in depression and generate positive responses. In addition to this model, this paper released a dataset named PosiChat, which contained depression texts and their comments from Sina Weibo. PosiGPT was tested on PosiChat dataset. The results show that the comments generated by PosiGPT have strong fluency, rationality and are always positive, achieving the goal of depression detection and positive response.

Keywords　Text generation　Depression detection　Positive comments　Pre-training

0　引　言

如今,随着工作和生活节奏的加快,越来越多的人由于忽视心理问题而患上抑郁情绪,在社交平台上倾诉他们的经历。其需要适当的心理干预,如朋友的鼓励或咨询师的治疗。由于羞怯和尴尬,这些人大多不愿寻求值得信赖的人或权威心理学家的帮助,从而导致更糟的情况,甚至患上抑郁症。

为了缓解这些问题,本文提出一个新颖的评论模型——PosiGPT,即基于预训练模型的积极情感评论模型。PosiGPT 使用中文预训练模型,在此之上使用本文定义的特殊场景的数据集,训练后得到正能量评论机器人,旨在帮助那些情绪低落的人渡过难关。

本文将详细阐述所使用的数据集和模型,包括数据的获取、处理,模型设计、训练和实验,以及本文的研究对社会的意义。

在对话机器人研究和应用方面,以往开放域对话系统[1]的工作主要集中于个性化聊天[2-3]或常识对话[4],本文认为积极情感的文本评论机器人作为情感型文本生成的一个应用,在社交评论情感检测和自动回复方面也具有重要意义。

收稿日期:2021-04-22。国家自然科学基金项目(61671157)。**卢晨耀**,硕士生,主研领域:自然语言处理。**李敏波**,副教授。

面对海量微博文本,PosiGPT 首先检测文本是否表现出消极情感,然后根据不同粒度的消极程度,对文本进行评论回复。为了确定文本的情感,本文使用开源的中文情感分析工具 bixin(https://github.com/bung87/bixin)对博文进行情感打分。如果文本的情感是消极的,那么 PosiGPT 将通过积极的正能量评论鼓励博文作者。

以往的工作都表明,对话系统的构建需要大量的训练数据和合理设置的语言模型。考虑到生成任务通常需要表现良好的通用语言模型作为基础,本文使用 Pretrain-finetune(预训练-微调)的方式来构建评论机器人。基于 Chinese GPT-2[5] 的 GPT2-chitchat[6] 预训练模型,使用大量中文对话数据集进行预训练后,得到一个通用对话模型,可以生成流畅的文本,在文本生成的流畅性和上下文相关性上具有出色的表现。本文使用 GPT2-chitchat 作为基础模型,在这之上进行微调和优化。在模型微调阶段,PosiGPT 使用 PosiChat 微博数据集,使模型在训练时梯度将朝着想要的特性方向移动。

根据本文的实验结果,PosiGPT 在评估任务上表现良好。此外,用少量的数据集进行微调即可实现出色的文本生成性能。为了定量评估模型的这些特性,本文建立不同的实验来进行本文模型与其他基线(Baseline)模型的对比,以及分析训练样本数量差别对实验的影响,并对结果进行分析。实验结果表明,本文模型不仅在评测任务上表现良好,还具有一定的迁移性,在小样本实验上表现出色。

综上,本文的主要工作如下:

(1) 提供一个名为 PosiChat 的大型数据集,其中包含高质量且细粒度的抑郁微博及其积极评论。

(2) 提供一个评论回复模型 PosiGPT,旨在检测发现抑郁文本并进行积极情感消息回复。在 PosiChat 数据集上进行评估。结果表明,与现有模型相比,本文模型产生的文本质量更高,情感更正面。

1 PosiChat 数据集

本节将详细介绍数据集,包括数据获取、数据清理和数据统计。首先,为了得到带有抑郁情感色彩的微博文本,本文定义不同粒度的抑郁情感色彩,分别为轻度、中度、重度,每种情感包含不同的表示抑郁情感的关键字。随后,将这些词语作为关键词,爬取相关微博及其评论。为了增加任务难度,除了爬取博文中带关键字的文本外,本文进一步获得了带有 hashtag 标记的微博。所谓 hashtag 标签,即用户在微博上发布博文时,使用两个“#”符号,中间带有关键字。分析 hashtag 中的词,可以解析博文表达的情绪,而提取正文时,将 hashtag 清除。这样获取的文本带有明显情感特征,而正文中未必出现情感特征关键字,增加了模型判别和生成的难度。如“#难过#我的朋友们都要走了,舍不得”一句中,正文里并未出现明显表示情感特征的词汇,但通过 hashtag“#难过#”可以判断出文本呈负面状态。获取该数据集后,其他的研究人员也可以根据具体情况选择使用何种文本来进行训练。

表 1 是数据集中的两个样本示例。它们之间的区别在于,两者都被归为负面微博,然而前者正文里包含关键字“难过”,而后者不包含情绪关键字。

表 1 两种不同的数据样本示例

微博正文	评论
喝酒最开心,宿醉最难过。	做回曾经最开心的你!
我的朋友们都要走了,舍不得。	要加油啊,等你考研结束回家好好玩儿!

为了防止模型通过判断关键字“作弊”,本文建议研究人员使用这两种文本的混合数据。

1.1 数据获取

本文使用新浪微博平台获取相关博文。为了获得多样化的微博,本文定义一些关键字,再使用关键字提取微博。这些关键字分别是:绝望、痛苦、悲伤、恐惧、失眠、焦虑、自杀、想死、抑郁、压抑、难过、伤心、失落、害怕、不开心等。除此之外,还包括这些词的同义替换词。

根据调查,这些词最有可能出现在负面情绪文字中。不同的关键字表达不同粒度的消极情绪,本文将不同的关键字进行粒度细分。表 2 显示了关键词粒度等级,以及在该程度下的关键字分类。其可以用作评估模型的指标。如应以温和的鼓励来回应略带消极的文本,而重度的消极情绪则需要更为强烈的安慰。

表 2 不同粒度的消极情感关键字分类

轻度	中度	重度
不开心、失落、难过、伤心	悲伤、害怕、失眠、焦虑	绝望、痛苦、恐惧、自杀、想死、抑郁、压抑

本文获取从 2020 年 1 月 1 日到 2020 年 11 月 1 日之间的微博及其评论。这是为了确保训练样本的时效性。然后,本文选择清理后的博客及其评论,并将它们保存到两个文件中:博客和评论文件。博客文件包含按 blog_id 区分的博客文本及博客的其他属性。评论文件包含以 comment_id 区分的评论相关属性,以及该评论对应的原博客的 blog_id。使用该数据集的研究

人员可以通过评论文件 comment_id 获取评论,并通过其 blog_id 定位到其原始博客。

本文研究保留了其他博客信息和用户信息,例如位置、设备类型和发文时间等,以供其他用途。

1.2　数据清洗

获取原始数据后,将其进行数据清洗,形成一轮博文-回复。清洗过程如下:

(1) 选择带有至少一个评论的微博,忽略不带任何评论的微博,并过滤系统机器人的评论。选择超过 10 个字且少于 200 个字的微博文本及其评论。

(2) 删除微博文本内的表情符号、特殊符号、URL、图片等非文本信息。如果用户提到另一个用户,删除所有这些标记。过滤日常生活中少见的特殊符号。删除乱码,删除其他无用的文本,例如"显示地图""位置"等。

(3) 删除所有其他语言的文本,删除不是文本中必要组成部分的数字。例如,在"我 3 点要开会"的情况下,不删除数字,而诸如"171 啊今天太糟糕了"情况,数字"171"为无意义信息,因此将其删除。最后,将所有标点符号替换为空格,以便进行训练。

(4) 将一条微博和一条它的评论放在一起,组成一对文本。忽视涉及三人及以上的用户交互评论。

(5) 仅保留正面评论,使用 bixin 工具包过滤中性或负面对话。

(6) 由众包人员重新人工检查数据集,仅保留高质量有意义的文本对。

经过以上工作,获得高质量的数据集样本。表 3 显示了部分数据示例。

表 3　清洗后数据集样本示例

微博内容	评论
喝酒最开心,宿醉最难过	做回曾经最开心的你
带病去工作已是很痛苦	不要这么拼啦,好好休息
普通又难过	抱抱,马上天冷了可以吃烤地瓜了
心里感到绝望是一种什么感觉	没啥,负能量也只是一段时间存在而已,不要怕,年轻人
我的朋友们都要走了,难过,舍不得	要加油啊,等你考研结束回家好好玩儿
挺失败,难过的时候突然发现除了自己就只剩马桶了,是啊,我只是一个人	世界这么大,你不可能永远都是一个人
一味想死,是因为太过认真地活着	每个问题都有一万种解决办法,为了把它们找全,你要活下去

1.3　数据统计信息

经过清洗的"博文-评论"总数量为 16 265。与其他数据集相比,该数据集在非预训练数据集中样本容量适中。考虑到数据集的质量比较高,数据集后续也可用于迁移学习(Transfer Learning)或少样本学习(Few-shot Learning)。表 4 统计了本文数据集 PosiChat 和其他数据集的比较信息。

表 4　不同数据集的样本数据、领域、语言统计信息

数据集	对话数量	领域	语言
PosiChat	16 265	开放型	中文
Persona-Chat[2]	10 907	开放型	英文
MultiWOZ2.2[7]	10 000	任务型	英文
CDialGPT[8]	6 800 000	开放型	中文
DialoGPT[9]	147 000 000	开放型	英文

将整个数据集分为训练集、测试集和验证集。这三个集合的样本数量分别为 15 451、407、407,所占比例分别为 95%、2.5%、2.5%。如果研究人员想要建立迁移学习模型或少样本学习模型,则这三个集合的数量可以根据任务的具体细节更改为合适的大小。

除了数据集的大小,本文进一步计算了样本评论的情感分数,利用 bixin 情感分析工具对评论情感进行打分。通常来说,分数的绝对值越高,文本的情感极性越强。正分数代表积极的情绪,负分数代表消极的情绪。分数范围为[-1,1]。

对训练集、测试集、验证集评论进行情感打分的定量分析,训练集、测试集、验证集中评论情感平均得分分别为 0.82、0.80、0.77。结果显示三个集合中,样本平均为正向情感且分数稳定在 0.8 左右,训练集、测试集和验证集的样本评分方差小,保证训练与测试时样本情感的一致性。

2　PosiGPT 模型

图 1 显示了 PosiGPT 模型整体架构。首先,需要一个通用中文语言预训练模型,来得到连贯、流畅的中文语言表述。在此,本文选择 Chinese GPT-2[5] 作为预训练模型。随后,使用中文闲聊对话数据在该模型上进行训练,得到在对话任务上性能更好的 GPT-chitchat[6],专门处理中文对话任务。最后,在 PosiChat 数据中训练 GPT-chitchat,得到 PosiGPT 模型。经过训练微调的 PosiGPT 模型在对话评论中能产生更正面情感的文本。

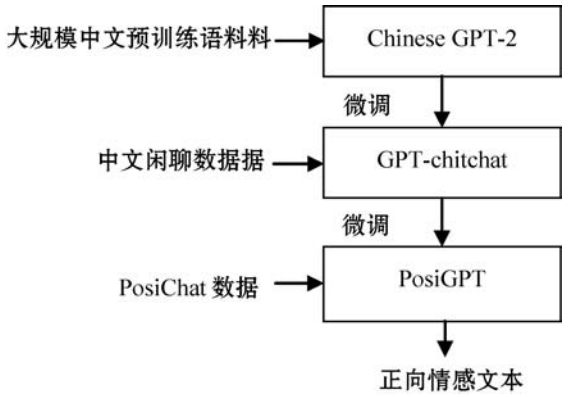

图 1 PosiGPT 模型结构

GPT-2 是 NLP 任务中近年来最著名的生成模型之一,在诸多任务上达到了 SOTA(State of the Arts)效果。本文用到的 Chinese GPT-2 模型继承了原始 GPT-2[10]的体系结构,在大规模中文预训练数据集上,如 wiki-zh、点评数据、新闻数据等语料上进行训练。Chinese GPT-2 具有良好的中文文本生成能力。在此基础上,GPT-chitchat 使用 50 万中文闲聊对话数据,对 GPT-chitchat 模型进行了微调,使得模型更适用于日常对话。

本文将 GPT-chitchat 作为在 PosiGPT 进行微调之前的预训练模型,并使用 Huggingface 的 Transformers[11] API 完成 PosiGPT 模型。以下部分将简要介绍 GPT-2 模型,然后主要介绍 PosiGPT 的模型细节。

2.1 GPT-2 和 Transformer 模型

GPT-2[10] 基于 Transformer[12] 结构。Transformer 在多项自然语言处理任务中均取得了卓越的性能。随后 Transformer 主要被用于自然语言生成领域,例如机器翻译、文本摘要、对话系统、小说生成等。最成功的基于 Transformer 的自回归模型是 GPT[10,13-14] 家族,这些模型使用了 6 ~ 24 层不等的 Transformer 层,在超大规模的语料上进行预训练,根据不同的具体任务进行微调,一经发布便刷新了自然语言处理的最好结果。

Transformer 的结构如图 2 所示。其使用多头自注意力机制(multi-head attention)[12],编码器在输入的序列符号(token)之间进行注意力打分,而解码器的每一步通过编码输出以及前一步所生成的内容,生成当前输出的符号。如样本"text:我的朋友们都要走了,舍不得。comment:要加油啊,等你考研结束回家好好玩儿!"中,编码器对"我的朋友们都要走了,舍不得。"逐位置进行编码,得到编码输出。而解码器以开始符号"SOS"(Start of Sentence)开始,一步步生成整个文本。如第一步以编码器输出和"SOS"为输入,解码器生成"要",然后以编码器输出和"要"作为输入,生成"加",以此类推,直到生成完毕。

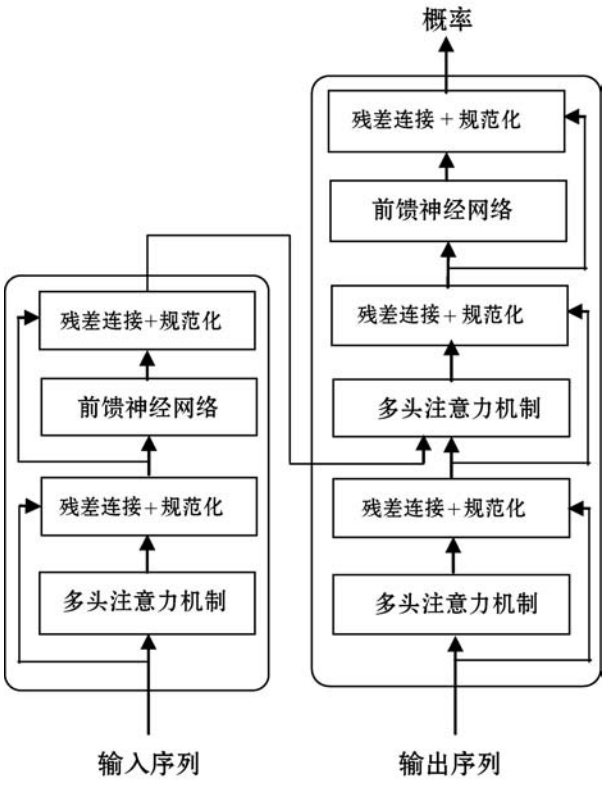

图 2 Transformer 结构示意图[12]

Transformer 编码器共有六层,每层有两个子块。第一个子块是多头注意机制层,第二个子块是前馈神经网络层,它们之间通过残差连接[15],以加快模型收敛速度和堆叠深层网络层。

解码器的结构与编码器大体上相似。然而不同于编码器使用自注意力(self-attention),解码器中的多头注意力模块,是将编码器的输出作为键(key)和值(value),而将解码器的输入作为查询(query),再进行注意力操作的,操作的公式为:

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{1}$$

$$h_{\mathrm{eadi}} = Attention(\boldsymbol{Q}\boldsymbol{W}_i^{Q},\boldsymbol{Q}\boldsymbol{W}_i^{K},\boldsymbol{Q}\boldsymbol{W}_i^{V}) \tag{2}$$

$$Multihead(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{Concat}(h_{\mathrm{eadl}},h_{\mathrm{ead2}},\cdots,h_{\mathrm{eadn}})\boldsymbol{W}^{o} \tag{3}$$

式中:$\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 分别代表 Query、Key 和 Value,即查询、键和值。在 Transformer 的编码器中,$\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 分别为编码器的输入序列经过转换的矩阵。而在解码器中,编码器的输出作为 $\boldsymbol{K}$ 和 $\boldsymbol{V}$,解码器的输入作为 $\boldsymbol{Q}$,进行 Attention 操作。式(2)中 h_{eadi} 代表第 i 个多头;$\boldsymbol{W}_i^{Q}$、$\boldsymbol{W}_i^{K}$、$\boldsymbol{W}_i^{V}$ 分别代表 h_{eadi} 中 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 的转换矩阵,式(2)表示每个多头 h_{eadi} 将对转换后的 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 进行式(1)中的 Attention 操作。式(3)中,Concat 操作将所有 n 个多头进行拼接,然后通过 $\boldsymbol{W}^{o}$ 转换矩阵后,输出多头自注意力机制的最终输出。

解码器这种操作可以更好地捕获对话输入和输出之间的关系,从而不仅将注意力集中在序列内部,还可以将注意力集中在序列与序列之间。

Transformer 的 Embedding 层包括 Word Embedding(词嵌入)和 Positional Embedding(位置嵌入)。Positional Embedding 确保文字的输入是按其位置进行“排序”的,而 Word Embedding 则给每个离散的词进行连续数值的向量化表示。

为了防止在训练过程中模型“看到”后面的单词进行“作弊”,Transformer 还需要进行掩码(attention-mask)操作,处于当前位置的词只能看到前面已经生成的信息,而不能看到后面的信息,从注意力机制上来说,即当前位置与后续位置的词之间的打分,分值为 0。

GPT-2 使用 12、24、36 或 48 层 Transformer 来构建模型,具体的层数取决于模型大小和任务的复杂度。同时它使用自回归方式的语言模型,对语料进行训练。而本文的 PosiPGT 也基本上遵循了 GPT-2 的配置。

2.2　符号规范

输入序列 x_i,其中 i 代表数据集 X 中的第 i 个样本。每个样本 x_i有一对文本对:源微博文本 c_i和微博评论 r_i。即,$x_i \in X$ 且 $x_i = [c_i, r_i]$。本文将 GPT-2 模型记为 $F(x)$,损失函数记 $L(y, y')$。其中:y 表示真实标签;y'表示预测标签。使用自回归方式来对序列进行建模,如式(4)所示。

$$P = \prod_{i=1}^{n} p(x_i \mid x_i) \tag{4}$$

式中:x_i表示第 i 个位置前的已知序列;x_i代表第 i 个位置的词。

2.3　Embedding,符号化,编码器

要训练类似 GPT 的模型,首先需要将一对文本对放在一起,随后对它们进行符号化(Tokenization)。本文使用与 BERT[16] 模型相同的 Word Embedding 和 Positional Embedding 方式。

经过 Embedding 后的序列将输入编码器,编码器一般为 6 ~ 12 层 Transformer 编码器层,通过多头注意力和自回归方式,将序列特征进行编码。

2.4　解码和损失函数

解码器结构与编码器类似。解码后模型的输出——logits——经过 Softmax 函数,生成[0, 1]之间的概率分布,该分布即为预测输出的单词在词汇表上的概率。然后选择最高概率的词作为本步输出的内容。

实验中也可选择在解码时设置 top-k [17] 参数。如果设置了 top-k 参数,则解码时模型选择 top-k 个最高概率的词,随后在这些词中选择一个作为本步输出的词。完成本步的输出后,输出的词与其他输入一起,再次进入解码器,进行下一时刻的词的生成。重复上述过程,直到生成结束。设置 top-k 参数的目的主要是避免少数高频词反复生成,同时保持生成文本的多样性和丰富性。

PosiGPT 使用与 GPT-2 论文中相同的交叉熵损失(Cross Entropy Loss,CE)来作为损失函数。原始的交叉熵损失函数为:

$$L_{\text{oss}} = -\sum_{k=1}^{N} p_k \times \log q_k \tag{5}$$

式中:p_k表示第 k 个词的真实概率;q_k表示第 k 个词的预测概率。由于真实标签是一个独热(one-hot)向量,因此式(5)也可以改写为:

$$L_{\text{oss}} = -\log q_m \tag{6}$$

式中:m 是真实标签在词表中的索引值。

模型将按序列顺序,将一个序列中的所有交叉熵损失相加,并求平均值,得到一个样本的最终损失值。

3　实验与结果分析

本文使用 PosiGPT 模型在 PosiChat 数据集进行实验。

3.1　实验环境

本文在 Linux CentOS 7.5 操作系统上进行实验,并使用 Intel Xeon E5-2620 V4 CPU。服务器的 DRAM 为 250 GB。使用 11 GB VRAM 的 Nvidia RTX 2080Ti 显卡进行深度学习训练。

3.2　实验设置

本文使用 10 层 Transformer 的 GPT-2 模型,每个 Transformer 中有 12 个多头(Head)。隐藏层的维度大小为 768。选用 768 作为隐状态维度有两个方面的考虑:一方面,它可以使模型更轻巧,有助于在训练阶段快速收敛。实验结果也证明了这一点,该模型使用 2080Ti 显卡可以在 3 个小时内完成 50 轮迭代,取得了很好的效果。另一方面,当模型中的参数过多且网络太大时,由于本文的数据集有限,容易引起过拟合的问题。实验中当设置 24 个编码层和 1 024 个多头时,确实出现了过拟合的情况。

表 5 展示了实验中的训练样本、测试样本和验证样本的数量分配,表 6 展示了本文具体的模型参数设置以及与原始 GPT-2 的参数比较。

表 5 训练集、测试集和验证集的样本数量

模型	训练集	测试集	验证集	总计
PosiGPT	15 451	407	407	16 265

表 6 PosiGPT 与 GPT-2 的模型参数设置比较

模型	Head 数量	层数	隐状态维度	词表大小
PosiGPT	12	10	768	13 317
GPT-2	12	12	768	50 257

3.3 评估指标

自然语言生成任务中有三个重要的指标来评估模型的性能,即回复的相关性、准确性和流畅性。在机器自动评估上,BLEU [18]是最常用的评估模型性能的指标。BLEU 衡量的是生成的语句出现的可能性。假设原始评论文本为 Reference,而模型生成的文本为 Candidate。假设 Candidate 一共有 M 个 n-gram,而 Reference 一共有 N 个 n-gram。首先计算 Candidate 中每个 n-gram 在 Reference 中出现的次数 c_{ntr},然后计算在 Candidate 中出现的次数 c_{ntc},则 $c_{ount} = \min(c_{ntr}, c_{ntc})$。BLEU 算法对模型的打分 p 为:

$$p = \frac{\sum_{\text{n-gram} \in \text{Candidate}} \min(c_{ntr}, c_{ntc})}{N} \tag{7}$$

BLEU 的总分为 100%,按照以往文献中的表示方法,本文实验中保留百分号前面的分数值。

此外,本文为正向情感文本生成,文本的情感也是一个重要的衡量指标。因此在实验阶段,对测试的结果使用 bixin 工具进行情感分析,分数的范围为[-1,1]。并汇总每个实验组的平均得分。

除了 BLEU 外,本文还对模型的结果进行人工评估。具体方式为:对于每个实验结果,招募 50 个志愿者进行对模型生成的回复,在相关性、流畅性上进行打分,总分为 5 分。并对打分的结果进行统计和取均值。打分的判断标准有:生成的文本是否流畅,评论与原博关联性是否足够。

3.4 结果分析

本文使用原始 Chinese GPT-2 模型和 PosiGPT 模型进行对比实验。表 7 和表 8 分别显示了在人类评估和在机器评估上的实验结果。

表 7 PosiGPT 与 Chinese GPT-2 人类评估实验结果对比

模型	流畅性	相关性
ChineseGPT-2	3.2	3.8
PosiGPT	**4.6**	**4.8**

表 8 PosiGPT 与 Chinese GPT-2 机器评估实验结果对比

模型	BLEU	情感打分
ChineseGPT-2	3.61	0.036
PosiGPT	**10.26**	**0.362**

表 7 显示了在人类评估上的实验结果。流利性和相关性这两个度量标准的满分为 5。流利度指标主要评估模型的生成是否自然,语言是否与真实的人类语言类似,没有机械感。相关性指标则衡量生成文本是否与微博文本相关,其关联度为多少。

表 8 显示了在 BLEU 分数和情感分数评估上的实验结果。可以看出,PosiGPT 比未经训练的 Chinese GPT-2 模型取得更好的效果。PosiGPT 在 BLEU 得分上显著高于 Chinese GPT-2,即模型生成的文本准确性更高,验证了在微调阶段,预训练模型由于 PosiChat 数据集样本的高质量性,模型“学会”了更好地对消极文本作出连贯流利的回复。

此外,PosiGPT 模型生成的文本的打分均分为 0.362,而 ChineseGPT-2 生成的文本情感均分为 0.036,即文本大多生成中性回复,或正面情感与负面情感抵消。由于没有本文“检测消极文本,生成积极评论”的任务约束,ChineseGPT-2 更偏向于无情感或中性情感表述。

PosiGPT 在这两项的打分,验证了其生成文本流畅、准确,且情感倾向基本为正向,性能远远超过了基准模型 ChineseGPT-2。

除了表 7 和表 8 显示的结果外,本文还在其他几种模型中进行实验,例如 BERT、BERT + RNN 和原始 GRU 模型等。尽管基于 BERT + RNN 的模型的人类得分不如本文模型好,但在实验期间仍然表现较好。这主要是归功于经过预训练的 BERT 已经具有良好的语言建模能力,加上 RNN 天然编码序列任务的优势,BERT 模型可以完成本文背景下的任务。但是由于 BERT 使用自编码方式进行训练,而 RNN 是自回归式生成,两者的训练方式有所差别,因此对于生成任务而言,BERT + RNN 模型训练的效果不如 PosiGPT。至于 GRU 模型,其得分比上述模型都要低。这是因为模型过于简单以至无法记住长序列中重要的信息,在诸如积极情感文本回复之类的可控文本生成任务中,模型缺少足够的记忆能力和控制能力。

除了上述实验之外,本文还进行少样本(Few-shot)实验。即在仅有几十或几百个样本的训练集上训练模型,在这种条件下,测试模型是否依然具有足够优秀的生成能力,以及衡量获取更多样本的成本与模型使用少量样本而损失的性能之间,对研究人员来说

哪种方式收益更大。

小样本实验阶段,模型仅使用 500 个样本来进行训练。为了保证对照实验的科学性,这 500 个样本的平均情感得分与之前的训练集一致,得分均分在 0.8 分。随后,模型再使用另外 500 个样本进行测试。这部分评估结果使用与上述类似的评估标准来判断模型的性能。

实验结果如表 9 所示。其中,PosiGPT 为标准模型,展示的是在 15 000 多个样本训练后的测试结果;而 PosiGPT-mini 代表了小样本模型,展示的是用同样大小的模型在 500 个样本下的测试结果。

表 9　小样本训练实验结果与正常训练实验结果对比

模型	样本数	测试数	BLEU	人类评分
PosiGPT	15 451	407	**10.26**	**4.8**
PosiGPT-mini	**500**	**500**	5.15	4.3

表 9 显示了在使用 500 个样本经过 50 个迭代之后,尽管模型没有获得与标准情况下相同的评分,但其表现效果尚可,与使用 15 451 个样本训练得到的 4.8 分相比,500 个训练样本能达到 4.3 分的平均得分。而在 BLEU 评分上,PosiGPT-mini 虽然不及 PosiGPT 的 10.26 分,但是仍然比 ChineseGPT-2 提升了不少。若在限制条件不严格的情况下,PosiGPT-mini 也可用于初步的情感生成。

以上结果表明高质量的对话可以减少训练样本的数量,仅仅需要少量的样本,即可得到较佳的学习效果。如果获取和收集训练数据成本很高,而增加大量的训练样本对模型效果的提升有限,或是模型在少量样本下的训练结果就可以满足研究人员的使用,则使用少量样本进行训练即可。这部分实验体现了 PosiGPT 也可在样本缺乏的情况下,仅仅用少数样本,就能在文本情感生成中达到出色的表现。

此外,如果以某些属性为条件,如生成的风格、情感、字数等,使用预训练-微调的训练方式,也可以有效地控制文本的生成。在样本不足的情况下,预训练-微调方式确实能够达到比传统的监督式学习更好的实验效果。

在实验期间,本文还调整了 top-k 参数对结果进行分析,使用不同的 top-k 参数,模型表现有所区别。实验显示,将参数 top-k 设置为取值在[1,4]时,模型的精度更高,在测试期间可以达到更好的结果,但会缺乏生成的多样性,同时容易产生某些高频字词重复出现的情况,若要解决这个问题,需要合理初始化模型参数,仔细调整模型参数初始值。而在 top-k 取值在[5,8]时,模型的自由生成和表达能力更强,可以生成更为丰富和灵活的语句,而在准确度上会有一定的损失。

根据不同的具体任务需求,设置不同的 top-k 参数,可以有效调整模型生成的性能。

4　结　语

本文提出基于生成式预训练模型的正向情感评论模型——PosiGPT。PosiGPT 旨在检测发现带有抑郁情感的文本,针对文本描述的背景产生积极情感的回复,以鼓励博文作者,给予其安慰和支持。同时本文发布一个名为 PosiChat 的新型聊天数据集,数据集的样本由带有抑郁情感的博文和积极正面的博文评论组成。为了测试模型性能,评估数据集质量,本文设置基线(Baseline)模型作对照实验,分析 PosiGPT 在任务中的性能和表现。

本文使用常用的生成模型 Chinese GPT-2 与 PosiGPT 进行实验对比。实验结果表明,PosiGPT 可以产生自然和流利的语言,以类似于人的方式对抑郁的微博文本作出积极情感的回复,试图鼓励和安慰微博作者,达到了本文检测消极评论、生成积极回复的目的。

尽管如此,本文的工作仍然有很多局限性,希望将来研究人员可以改进本文模型,丰富本文提出的数据集。本文的局限性包括:仅在作者认为相对重要的关键字上获取相关文本,并未包含所有消极情感的微博;对照实验使用的基线模型较少,对话数据不够庞大;构建的模型仅适用于单回合回复等。希望研究本方向的研究人员能够改进本文的局限和不足。

另外,感谢 GPT2-chitchat[6] 的模型参数和代码贡献,给本文的研究提供了很大的参考。

本文希望提供的数据集在将来能得到使用,PosiGPT 模型能帮助那些处于抑郁状态的人得到安慰和鼓励。

参　考　文　献

[1] Pan Z, F Bai K, Wang Y, et al. Improving open-domain dialogue systems via multi-turn incomplete utterance restoration [C]//9th International Joint Conference on Natural Language Processing, 2019:1824 - 1833.

[2] Zhang S Z, Dinan E, Urbanek J, et al. Personalizing dialogue agents: I have a dog, do you have pets too? [C]//56th Annual Meeting of the Association for Computational Linguistics, 2018:2204 - 2213.

[3] Madotto A, Lin Z J, Wu C S, et al. Personalizing dialogue agents via meta-learning[C]//57th Annual Meeting of the

Association for Computational Linguistics,2019:5454 - 5459.
[4] Zhang S Q, Stone P. CORPP: Commonsense reasoning and probabilistic planning, as applied to dialog with a mobile robot[C]//29th AAAI Conference on Artificial Intelligence, 2015:1394 - 1400.
[5] Gpt2-chinese: Tools for training GPT2 model in Chinese language[EB/OL]. [2021 - 04 - 20]. https://github.com/Morizeyao/GPT2-Chinese.
[6] Yang J X. CPT2-chitchat[EB/OL]. [2021 - 04 - 20]. https://github.com/yangjianxin1/GPT2-chitchat.
[7] Zang X, Rastogi A, Sunkara S, et al. MultiWOZ2.2: A dialogue dataset with additional annotation corrections and state tracking baselines[C]//2nd Workshop on Natural Language Processing for Conversational AI,2020:109 - 117.
[8] Wang Y D, Ke P, Zheng Y H, et al. A large-scale Chinese short-text conversation dataset[C]//CCF International Conference on Natural Language Processing and Chinese Computing,2020:91 - 103.
[9] Zhang Y Z, Sun S Q, Galley M, et al. Dialogpt: Large-scale generative pre-training for conversational response generation[EB]. arXiv:1911.00536,2019.
[10] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI blog, 2019, 1(8):9.
[11] Wolf T, Chaumond J, Debut L, et al. Transformers: State-of-the-art natural language processing[C]//Conference on Empirical Methods in Natural Language Processing: System Demonstrations,2020:38 - 45.
[12] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[EB]. arXiv:1706.03762v1,2017.
[13] Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training[EB/OL]. [2021-04-20]. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.
[14] Brown T B, Mann B, Ryder N, et al. Language models are few-shot learners[EB]. arXiv:2005.14165,2020.
[15] He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition[C]//IEEE Conference on Computer Vision and Pattern Recognition,2016:770 - 778.
[16] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding [EB]. arXiv:1810.04805,2018.
[17] Fan A, Lewis M, Dauphin Y. Hierarchical neural story generation[C]//56th Annual Meeting of the Association for Computational Linguistics,2018:889 - 898.
[18] Papineni K, Roukos S, Ward T, et al. Bleu: A method for automatic evaluation of machine translation[C]//40th annual meeting of the Association for Computational Linguistics, 2002:311 - 318.

(上接第 8 页)

[37] Lv B, Wang D, Wang Y, et al. A hybrid model based on multi-dimensional features for insider threat detection[C]//International Conference on Wireless Algorithms, Systems, and Applications,2018:333 - 344.
[38] Liu L, Chen C, Zhang J, et al. Insider threat identification using the simultaneous neural learning[J]. IEEE Access, 2019,7:183162 - 183176.
[39] Zhang D X, Zheng Y, Wen Y, et al. Role-based log analysis applying deep learning for insider threat detection[C]//1st Workshop on Security-Oriented Designs of Computer Architectures and Processors,2018:18 - 20.
[40] Lu J M, Wong R K. Insider threat detection with long short-term memory[C]//Australasian Computer Science Week Multiconference,2019:1 - 10.
[41] Sharma B, Pokharel P, Joshi B. User behavior analytics for anomaly detection using LSTM autoencoder insider threat detection[C]//11th International Conference on Advances in Information Technology,2020:1 - 9.
[42] Yuan F, Shang Y M, Liu Y B, et al. Attention-based LSTM for insider threat detection[C]//International Conference on Applications and Techniques in Information Security,2019: 192 - 201.
[43] Tian Z H, Shi W, Tan Z Y, et al. Deep learning and Dempster-Shafer theory based insider threat detection[J]. Mobile Networks and Applications,2020,28(3):213 - 221.
[44] Li G Q, Wu S X, Zhang S L, et al. Neural networks-aided insider attack detection for the average consensus algorithm [J]. IEEE Access,2020,8:51871 - 51883.
[45] Gayathri R G, Sajjanhar A, Xiang Y. Image-based feature representation for insider threat classification[J]. Applied Sciences,2020,10(14): 4945.
[46] 王毅,冯小年,钱铁云,等. 基于 CNN 和 LSTM 深度网络的伪装用户入侵检测[J]. 计算机科学与探索,2018,12(4): 575 - 585.
[47] Saaudi A, Al-Ibadi Z, Tong Y. Insider threats detection using CNN-LSTM model[C]//International Conference on Computational Science and Computational Intelligence,2018: 94 - 99.
[48] Tian Z H, Luo C, Lu H. User and entity behavior analysis under urban big data[J]. ACM Transactions on Data Science,2020,1(3):1 - 19.
[49] Karim F, Majumdar S, Darabi H, et al. LSTM fully convolutional networks for time series classification[J]. IEEE Access,2017,6:1662 - 1669.
[50] Veeramachaneni K, Arnaldo I, Korrapati V. AI^2: Training a big data ma-chine to defend[C]//2nd International Conference on Big Data Security on Cloud,2016:49 - 54.