语言分析与机器翻译实践报告

一、实践内容

自拟题目：基于情感的对话文本生成（Emotional Conversation Generation Machine，简称ECGM）。首先介绍模型使用的数据集，其次介绍模型结构、模型伪代码，最后介绍可视化界面。

1、EmpatheticDialogue数据集、GloVe词向量

该模型所用到的数据集均为公开数据集，复制地址粘贴到浏览器搜索框，按下回车获取，保存到系统内部指定文件夹。具体地址见表1.1：

表 1.1 数据集下载地址

|  |  |
| --- | --- |
| 名称 | 下载地址 |
| EmpatheticDialogues  glove词向量 | https://dl.fbaipublicfiles.com/parlai/empatheticdialogues/  empatheticdialogues.tar.gz  https://nlp.stanford.edu/projects/glove/ |

EmpatheticDialogue是facebook AI实验室发布的公开数据集，对话出自于810个接受雇佣的工作人员，他们先根据不同的情感词生成多个与之对应的场景，再围绕不同的场景做出对话。已有许多研究课题采用该数据集，均取得较好的结果，尤其在情感表示的研究方面效果显著。该数据集包括对话轮次、情感标注、对话内容等属性，详见表1.2。

表 1.2 EmpatheticDialogue数据集属性及内容

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 属性 |  |  | 内 | 容 |  |  |  |
| Conv\_id | hit:0\_conv:1 | hit:0\_conv:1 | hit:0\_conv:1 | hit:1\_conv:2 | hit:1\_conv:2 | hit:1\_conv:3 | hit:1\_conv:3 |
| Utterance  \_idx | 1 | 2 | 3 | 1 | 2 | 1 | 2 |
| context | sentimental | sentimental | sentimental | afraid | afraid | proud | proud |
| prompt | I remember going to the fireworks with my best friend. | I remember going to the fireworks with my best friend. | I remember going to the fireworks with my best friend. | i used to scare for darkness | i used to scare for darkness | I showed a guy how to run a good bead in welding class. | I showed a guy how to run a good bead in welding class . |

续表 1.2 EmpatheticDialogue数据集属性及内容

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 属性 |  |  | 内 | 容 |  |  |  |
| Speaker  \_idx | 1 | 0 | 1 | 2 | 3 | 3 | 2 |
| utterance | I remember going to see the fireworks with my best friend. It was the first time we ever spent time alone together. | Was this a friend you were in love with\_comma\_ or just a best friend? | This was a best friend. I miss her. | it feels like hitting to blank wall when i see the darkness | Oh ya? I don't really see how | Hi how are you doing today | doing good.. how about you |
| selfeval | 5|5|5\_2|2|5 | 5|5|5\_2|2|5 | 5|5|5\_2|2|5 | 4|3|4\_3|5|5 | 4|3|4\_3|5|5 | 3|5|5\_4|3|4 | 3|5|5\_4|3|4 |

对于普通的序列到序列模型，在数据预处理阶段将情感标注剔除，只保留其对话内容；对于融合情感编码的序列到序列模型，则将情感标注与对话内容一并保留。

2、模型结构

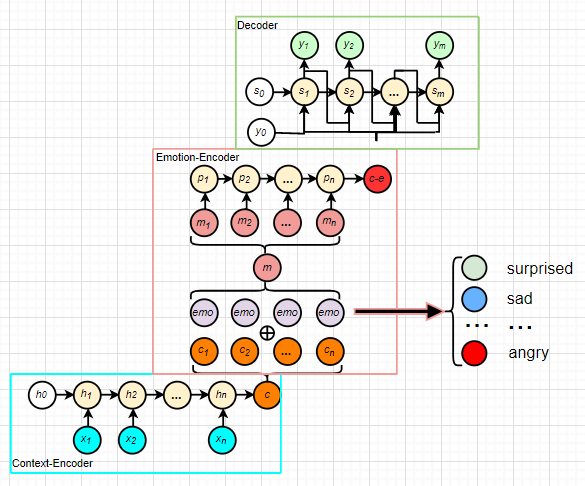
该模型分为三个模块：上下文编码器（Context-Encoder）、情感编码器（Emotion-Encoder）、解码器（Decoder），其结构见图1.1。

图 1. 1 融合情感编码的序列到序列模型

上下文编码器采用门控循环单元作为循环函数，将输入文本按单词顺序输入，经过*n*个时间步得到上下文向量*c*。

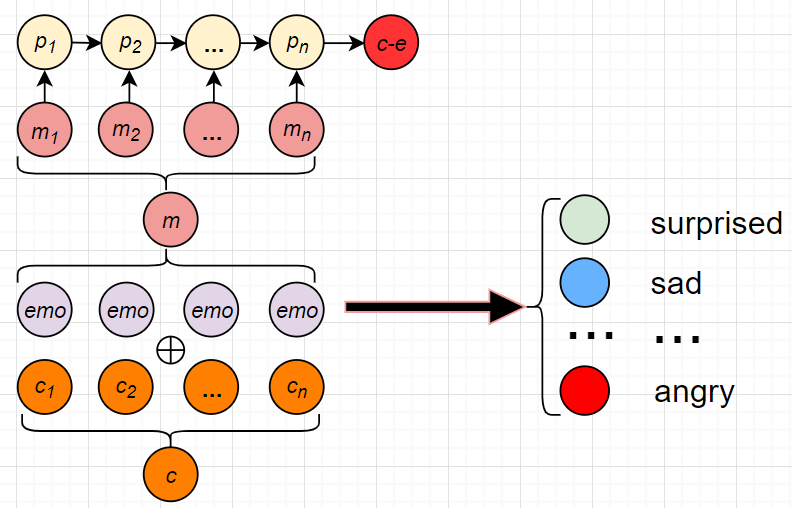
情感编码器结构见图1.2。

图 1. 2情感编码器结构图

采用GloVe词向量对情感标注进行编码，位于上下文编码器与解码器之间，结构上采用门控循环单元作为循环函数，从上下文编码器获取上下文向量c，将其在第一维度展开，得到；再从GloVe词向量文本文件中获取情感标注的词向量表示，将其复制n次，得到。二者在第二维度首尾拼接，展开后得到，作为情感编码器的输入，在时间步*t*，编码状态由GRU的隐藏状态表示，见公式（1.1）。

这样经过连续*n*个时间步后，输入文本的上下文内容以及情感标注所包含的全部信息被编码到向量*c-e*中。解码器采用门控循环单元作为循环函数，对*c-e*向量解码得到目标文本，既包含了上下文内容，又有丰富的情感表示。

序列到序列模型在生成文本的过程中，输入文本*X*位置越靠后的词对解码器生成文本的影响越大，反之位置越靠前的词在解码时越容易被忽略，这是因为普通序列到序列模型的编码器机制使得输入文本的最终编码状态更容易记住位置靠后词的信息。

为解决这一问题，该模型加入了注意力机制，使解码器在解码时的每个时间步都计算输入文本*X*中每一个词对生成当前词的影响权重，从而使解码器在对每个词进行预测时可以关注到输入文本*X*的各个部分。

采用注意力机制的解码器在计算每一时间步的解码状态时，模型都要接受上一步的解码状态以及上一步通过解码生成的词，同时也要融入注意力机制基于编码器的多个编码状态计算出的上下文向量。采用注意力机制的解码器运算方式见公式（1.2）。

其中表示上下文向量，由所有编码隐藏状态{}经过动态加权求和后得出。运算方式见公式（1.3）、（1.4）、（1.5）。

其中、、是可训练的参数；是*t*时刻的解码状态受编码状态的影响程度。公式（1.4）是一个softmax函数，通过此函数可将*t*时刻的解码状态受编码状态的影响程度进行归一化，从而得到对所有时刻的编码状态的注意力权重。公式（1.5）是一个对齐函数，表示了上下文间的对齐关系。被称为查询（Query）向量，{}被称为键（Key）向量以及值（Value）向量。对于注意力机制可以这样理解：键向量表示文本中的每一个词，查询向量对每一个键向量的影响程度进行分别计算，计算结果也就是注意力分布，然后再将其归一化得到权重值，最后基于这个权重值对值向量进行加权求和操作，得到的结果就是基于注意力的上下文向量。在得到解码器的隐藏状态之后，计算条件概率的方式与普通的序列到序列模型相同。通过注意力机制，解码器在生成每一个词时都动态地关注到了不同位置上的编码状态。

3、模型伪代码

融合情感编码的序列到序列模型算法的伪代码如表1.3所示。

|  |
| --- |
| **算法4.1 融合情感编码的序列到序列模型** |
| **Input:** 输入文本序列 |
| **Output**:目标文本序列 |
| **Begin** |
| 1 将输入文本序列*x*拆分为情感内容*emo*和两部分  2 **For** *i* = 1，…，*m* **Do**  3 **If**  //判断*emo*是否在32类情感标注之中  4 //是则保留  5 **Else**  6 //否则赋给默认值  7 **End IF**  8 **End For**  9 *x*通过上下文编码器中的GRU，利用公式（4.12）、（4.13）得到，令向量*c =* |
| 10 将*c*与在第一维度上进行拼接，得到  11 通过情感编码器中的GRU得到，令向量*c-e* =  12 *c-e*通过解码器中的注意力机制，利用公式（4.17）、（4.18）、（4.19）得到  13 通过解码器中的GRU得到 |
| 14 **For** *i* = 1，…，*m* **Do**  15 //通过softmax函数得到输出单词的概率 |
| 16 通过概率分布预测单词 |
| 17 **End For** |
| **END** |

表 1.3 融合情感编码的序列到序列模型预测效果

4、可视化界面

本系统的可视化界面基于Flask搭建。Flask是由python编写的轻量级Web应用框架，因为其使用简单的核心并通过extension增加其他的功能，故也被称为“microframework”。Flask的基本模式是在程序中把一个视图函数分配给一个URL，每当用户访问这个URL，系统就会执行为那个URL分配好的视图函数，获取函数的返回值，然后将返回值在浏览器上显示。

Flask具有灵活、轻便、高效等特点，同时拥有许多开源库、内置服务器以及单元测试，支持安全的cookies，而且官方文档内容十分完整，便于学习和掌握。Flask的内核简单，复杂功能全部以扩展组件的方式完成，使用这种微小型框架可以更为灵活地改变方案。

可视化界面搭建工作主要分两部分，一是界面设计，二是脚本编写。

首先建立一个名为style.css的文件，.css后缀是Cascading Style Sheet的缩写，即译为“层叠样式语言”。层叠样式语言属于样式表语言，没有变量和数据类型，完全依附于HTML。本系统定义标签选择器body，作用于所有标签，再定义类选择器.bg、.chat、.chat\_title、.messages、.message-box等，对多个标签进行同时控制。

然后建立一个名为index.js的文件，.js后缀是Java Script的缩写，是一种脚本语言，即事件驱动型编程语言，通常以发生某个事件为驱动条件，从而触发某段代码的执行。所有事件的句柄都以标签属性的形式存在，例如一个鼠标点击按钮就需要鼠标点击事件诸如onclick、mouseover作为标签属性，只要用鼠标点击此按钮，onclick事件句柄中的代码就会被执行。与编译型语言不同，脚本语言的目标程序以普通文本形式保存，嵌入于HTML中，在浏览器中运行。本系统在前端主要定义以下几个响应函数：insertMessage()、interact()、updateScrollbar()、setDate()。insertMessage()实现输入信息功能，点击发送按钮后更新滚动栏；interact()实现系统交互功能，接收insertMessage()的输入信息，结合训练好的模型进行预测，最终返回输出信息，更新滚动栏；updateScrollbar()实现滚动栏更新功能，将各部分模块的最新状态显示于界面；setDate()实现设置时间功能，用于计算系统停止操作的时长，从而向后台发送提示。

二、实践结果

通过前端可视化界面接受用户输入，并回复结果将展示出来。

结果展示模块会先判断用户所选择的是否为普通模型，如果是，则加载普通的序列到序列模型；如果不是，则加载融入情感编码的序列到序列模型。接下来用户在输入框中输入英文文本，点击“SEND”发送给系统，最终在聊天窗口中展示模型预测结果，见图2.1。基于情感的序列到序列模型的预测结果见表2.1。

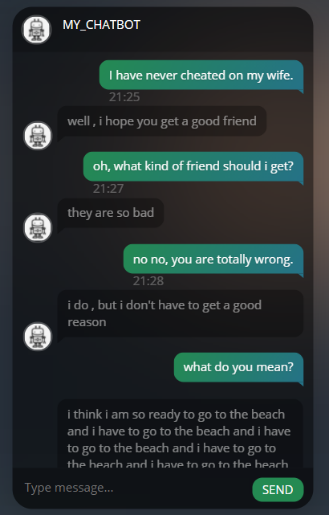
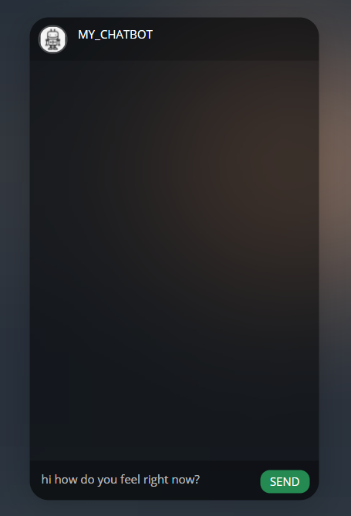


图 2. 1 结果展示图

表 2.1 融合情感编码的序列到序列模型预测效果

|  |  |  |
| --- | --- | --- |
| 输入文本 | 情感标签 | 输出文本 |
| I have never cheated on my wife. | Ashamed | oh no, I hope you were able to do. |
| I have never cheated on my wife. | Surprised | Oh no, I hope you have a good time. |
| I lost my job last year. | Angry | Oh no, I hope you have to be a good reason. |
| I lost my job last year. | Sad | Oh no, I am sorry to hear that. |
| I lost my job last year. | Sentimental | Oh no, I am sure you will. |
| I never thought things would go like this. | Sentimental | I hope you can do. |
| I never thought things would go like this. | Angry | I hope you will get it. |
| I never thought things would go like this. | Surprised | I hope you will be a good time. |

使用情感分类器判断输出结果的情感标签，计算其准确率、f1值。

准确率：0.7328，f1值：0.7211

三、心得体会

在完成该项目的过程中，熟悉了tensorflow框架的环境配置及使用，学会搭建一条数据预处理、构造模型、模型训练的pipeline，并基于此实现了一个基于情感的序列到序列模型。其中使用了Glove词向量，以及GRU、Attention等经典算法，对矩阵运算以及各种函数的计算公式有了更深入的理解。同时了解了各种损失函数以及优化算法，学习调参技巧。最后使用Flask搭建了一个简易的可视化界面，积累了一部分前端经验。