为短文本分类而改进的 attention 层级网络

程芷怡, 马烨

2020.6

摘要: 文本分类一直是自然语言处理研究的重要领域。在调研时我们发现，著名的论文 Hierarchical Attention Networks for Document Classification 所描述的结构只适用于有很强结构性的长文本，而对短文本效果不太好。于是我们借鉴了论文中的结构，将其进行迁移，在推特的数据集上进行了测试。
关键词: 自然语言处理 文本分类 层级结构 attention 机制

目录

1 问题描述

文本分类领域一直是自然语言处理研究的大热领域。直到目前为止，在文本分类领域比较前沿的工作是带有 attention 机制的层级网络。但是，这个网络只能解决较长的且带有明显层级结构的文本，比如豆瓣、IMDB 影评等。而对于短文本的效果较差。我们尝试对这种层级网络进行改造，进而提升在短文本分类（比如推特、微博等）上的准确率。

输入：作为训练集的文本和分类标签，以及作为测试集的文本

输出：测试集的分类预测

2 相关工作

我们选用了在传统文本分类问题上的三个比较有代表性的基础模型进行参照。基础参照模型 1 是 2016 年 Zichao Yang 发表的 Hierarchical Attention Networks for Document Classification 中所描述的带有 attention 的层级网络模型。基础参照模型 2 是 2017 年 github 上的 project:Convolutional Neural Networks for Sentence Classification，基于 CNN 实现文本分类。基础参照模型 3 是 2018 年 github 上的 project:Character Level CNNs in Keras，实现了字母层次上的 CNN。

3 数据集和预处理

3.1 数据集 Sentiment140

Sentiment140 数据集是斯坦福大学的一个课堂项目产生的一个用于情感分析的数据集，数据抓取自 twitter, 该数据集包含了 160 万条从推特爬取的推文，主要用于情感分析相关的训练。数据集包含两个文件：测试集和训练集。数据格式为：推文标注，推文 ID，发布日期，Query，用户，内容。推文标注有三种：0 代表负面，2 代表中立，4 代表正面。我们主要用到的数据是推文标注和推文内容。该数据集还有一个特点是训练集并没有中性类，中性类只在测试集中出现。

3.2 数据处理

对于不同的模型，我们结合网络特点采用了不同的数据预处理方案。本模型和 baseline1 采用了相同的数据处理模式。都是先将数据集进行随机打乱处理，再提取前 10000 个数据和标签。baseline2 的数据处理是先选取前 10000 个推文内容，然后按负面正面分类。baseline3 的数据处理是选取前 10000 个推文内容，然后基于字母字典建立索引。

4 模型结构

4.1 以 GRU 为基础的序列 encoder

GRU 用了一种门机制去追踪序列的状态而不需要花销额外的内存来记录。GRU 中有两种门，一种是重设门 r_t，一种是更新门 z_t。它们共同控制了信息在状态之间的流动。在时刻 t，GRU 计算更新后的状态：

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot h_t$$

。这是基于老状态 $ht-1$ 和新状态 h_t 的线性插值。z_t 门确定过去的信息中有多少留下来，而加入了多少新信息。z_t 被更新为：

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

。这里，x_t 是在 t 时刻的序列向量。候选状态 h_t 像 RNN 那样被计算：

$$h_t = tanh(W_h x_t + r_t \odot (U_h h_{t-1}) + b_h$$

。这里 r_t 是重设门，控制过去的状态对现在的状态贡献了多少信息。如果 r_t 为 0，则忘记之前的状态。重设门的更新方程为：

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

4.2 层级式 attention

我们在这里更加注重短文本的单词分类。假设一个文本有 L 个单词 s_i，每一个单词包含了 T_i 个字母。c_{it}（t∈ [1,T]）代表了第 i 个单词中的字母。这个模型将推特文本映射为一个向量表示。基于向量表示我们做文本分类。

4.2.1 字符 encoder

给定一个单词，由字符 c_{it}（t ∈ [1,T]）组成。我们用嵌入矩阵 W_e 先将字符映射成为向量，$x_{ij} = W_e\, c_{ij}$。我们使用双向 GRU 从不同方向总结字符上下文的表示，将语境信息嵌入字符的表示中。双向 GRU 包括前向的 GRU $\overrightarrow{f}$ 和后向 GRU $\overleftarrow{f}$，分别从前往后和从后往前依次输入字符：

$$x_{it} = W_e c_{it}, t \in [1, T]$$
$$\overrightarrow{h_{it}} = \overrightarrow{GRU}(x_{it}), t \in [1, T]$$
$$\overleftarrow{h_{it}} = \overleftarrow{GRU}(x_{it}), t \in [1, T]$$

。我们将得到的两个 h 拼起来得到隐藏状态。

4.2.2 字符 attention

并不是所有字符在对单词意思的贡献都是相等的。比如说 goooooood，o 的数目很多，体现了原作者的一种愉悦的心情，而 g 却没有这个作用。因此我们借鉴了原文之中的 attention 机制提取有重要意义的字符，把它们聚集起来形成单词的向量表示。特别地，

$$u_{it} = tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{exp(u_{it}^{\mathrm{T}} u_w)}{\Sigma_t exp(u_{it}^{\mathrm{T}} u_w)}$$

$$s_i = \Sigma_t \alpha_{it} h_{it}$$

即，我们首先将字符表示输入网络之后得到 u_{it} 作为隐藏状态 h_{it} 的表示，然后用 u_{it} 和语境向量 u_w 的相似性衡量重要性，输入 softmax 函数得到参数 α_{it}。之后计算单词向量 s_i 作为字符向量的加权和。

4.2.3 单词 encoder

和上面的字符 encoder 相似，用双向 GRU 编码单词，再粘合起来得到隐藏状态 h_i。

4.2.4 单词 attention

和字符 attention 相似。使用 attention 机制编码整条推特，得到最终的向量 v。

4.3 文本分类

v 是推特的高层表示，可以作为这条推特的 feature 用于文本分类。落在类别中的概率

$$p = softmax(W_c v + b_c)$$

。使用负的对数似然函数作为 loss 函数。

$$L = -\Sigma log p_{dj}$$

，j 是推特 d 的标记。

5 实验部分

5.1 模型参数

原数据集自然分割成为了训练集和测试集。我们用出现最多的前 20000 个词作为词典。其他的词直接忽略。我们设置字符编码维度为 200，GRU 维度为 50，双向 GRU 维度为 100. 语境向量随机化初始化，维度也是 100. 在训练中，我们的 batch 尺寸为 50. 随机取样 batch。我们使用了 rmsprop 的优化器，学习率为 0.005。

5.2 baseline 1 的参数

baseline 1 我们遵从了原论文中的数据。单词编码维度数为 200，GRU 维度数为 50，双向 GRU 维度数为 100. 语境向量同样是随机化初始化，维度 100. batch 尺寸 64，用随机梯度下降的优化器。学习率默认值。

5.3 baseline 2 的参数

epoch	filter_sizes	batch_size	embedding_dim	optimizer	dropout	loss
10	(3,8)	64	50	adam	(0.5,0.8)	binary_crossentropy

5.4 baseline 3 的参数

epoch	batch_size	embedding_size	optimizer	dropout	loss
5000	128	128	adam	0.5	categorical_crossentropy

5.5 训练结果比较

:

模型名称	loss	acc	val_loss	val_acc
本文模型	0.2257	0.9046	5.2870	0.4839
HAN	0.1997	0.9253	4.7457	0.5120
wordcnn	0.7058	0.4978	0.6956	0.4960
charcnn	0.0253	/	4.4988	/

5.6 结果分析和 Future work

经过对实验数据进行分析，我们发现虽然测试集上的效果提升不太明显。但是比起 HAN 模型，初值和峰值的提升已经达到了百分之七。但是遗憾的是，实验数据在 loss、准确率和收敛速度上都不如 HAN 模型。对于另外两个模型，我们惊喜地发现，在训练集的准确率上有较大的改进。由于我们只是在 10000 个数据上进行了训练，可能有过拟合倾向。未来有条件时，我们可能会尝试在 16000000 个推特文本的数据集上进行大规模训练，测试结果。另外，由于推特的文本短，句子划分不明显（一般只有 2-3 句），但是一个单词可能常常会有 20 个字母，因此我们的模型网络层数可能还是太深，在梯度上产生了一些问题。最后，我们还可能将 GRU 换成 LSTM，进行结果的对比和测试。

6 总结

在这篇论文中，我们提出了一种将 Hierarchical Attention Networks for Document Classification 论文中的层级结构和 attention 机制进行迁移从而处理情感较丰富、单词拼写较为多样的短文本的方法。注意到推特的单词拼写较为灵活，蕴含感情，以及一些字符组合表情包的存在，我们将单词和句子中的 encoder 和 attention 机制迁移到了字符和单词两层。但遗憾的是，相对于 HAN 结构，我们的方法并没有取得更加优秀的性能和结果，可能还需要进一步训练和改进。

小组成员和分工：

程芷怡 撰写模型代码，复现 HAN 的实验。

马烨 复现 WordCNN 和 charCNN 模型的实验。

项目 github 地址：https://github.com/Murphy-OrangeMud/TextClassifier

参考文献

[1] *Character Level CNNs in Keras*：https://github.com/mhjabreel/CharCnnKeras

[2] *Convolutional Neural Networks for Sentence Classification*:https://github.com/alexander-rakhlin/CNN-for-Sentence-Classification-in-Keras

[3] Zichao Yang，Diyi Yang，Chris Dyer:*Hierarchical Attention Networks for Document Classification* 2016 10.18653/v1/N16-1174

[4] Kamran Kowsari,Kiana Jafari Meimandi,Mojtaba Heidarysafa,Sanjana Mendu,Laura Barnes and Donald Brown: *Text Classification Algorithms: A Survey.* 23 April 2019

[5] *Text classifier for Hierarchical Attention Networks for Document Classification*:https://github.com/richliao/textClassifier