基于深度学习的中文谣言检测方案

1. 引言

在现代信息物理空间中，各种社交网络服务已成为日常生活中获取信息不可或缺的一部分 。这使得社会服务用户面临着不断增长的新闻和信息量无处不在。但与此同时，各种真假新闻混杂在一起，总是让社会服务用户难以很好地辨别其本质。目前的情况无疑给网络物理社会服务的运营商带来了很大的挑战。没有有效的技术监管手段，广大用户可能会受到威胁。一个典型案例是 2020 年早期 COVID-19 相关虚假新闻的广泛传播，带来了严重的经济和社会危害。因此，研究实用的虚假新闻检测方法，对于网络物理社会服务的运营商具有重要意义。

现有的网络物理社会服务假新闻检测研究大致可分为两类：统计检索和语义计算。前者主要是对句子或段落进行分词后，搜索与异常特征相关的关键词。然后，建立一个评分模型来评估句子的真实性作为结果。然而，以这种方式拆分句子的想法很难捕捉到句子的语义特征。当没有明确的异常关键字时，很难得到理想的结果。后者更侧重于建立数学模型来以向量化形式表示句子的语义。然后，利用机器学习中的分类或回归方法输出情感评价结果。然而，从语言学的角度来看，几乎任何一种语言都有大量的词单元，这使得向量模型的构建更加困难，具有挑战性。

目前，深度学习因其先进的信息处理结构已成为语义建模的主流思想，对算法效率提出了挑战。因为在现实世界的大规模数据流中，算法的实时性对于现实实践至关重要。

2. 技术方案

现实世界的大规模数据流中，算法的实时性对于现实世界的影响巨大，为弥补之间的差距，以基于中文文本的谣言数据为对象，该项目训练了基于深度学习的社交网络假新闻高性能检测模型。将中文短文本中的每个字直接作为基本处理单元。使用特征提取滤波器扫描句子中的每个单元字符，通过卷积运算从一个字符和相邻字符中提取联合特征表征。最后，通过全连接层映射最终获得检测结果。

利用基于卷积的神经计算结构实现了高性能假新闻检测，对从移动社交媒体收集的真实数据集进行了一组实验，以验证其效率。

为了快速将字符级特征表示合并为句子级特征表示，选择了 CNN。 CNN 是一种通过卷积运算自动从目标实体中学习隐藏特征的神经网络模型。卷积算子可以看作是一个过滤器，扫描目标实体的全局特征，从而获得新的抽象特征。 CNN 最常见的应用领域是计算机视觉和图像处理。由于像素级特征和词级特征都是向量的形式，因此，CNN 也可以用于语义建模。每个字的 embedding 都可以看作是一个垂直方向的向量。对于每个字符 Xn,m，假设它与其相邻的几个字符相关联。相邻字符的范围在 Xn,m 的前 d 个字符和后 d 个字符。也就是说，Xn,m-d 与其相邻的 2d 个词组合成了一个滑动窗口。它从 Xn,m-d 开始到字符 Xn,m+d 结束。在第一个字符 Xn,1 和最后一个字符，他们没有直接相邻的字符。为了构建一个完整的滑动窗口，选择 Xn 的最后 d 个字符来交替 Xn,1 之前的 d 个字符。这种循环邻接机制保证了滑动窗口的全面性。

3. 实验

1. 环境配置

CPU/GPU 均可，框架使用百度 paddlepaddle

2.数据集

1.解压数据，读取并解析数据，生成 data.txt

2.生成数据字典，即 dict.txt

3.生成数据列表，并进行训练集(train.txt)与验证集(val.txt)的划分

4.定义训练 Dataset 和 DataLoader

3.加载模型

CNN 和 Transfomer

4. 超参数的设置：

maxlen	200
Seq_len	200
Batch_size	32
Epochs	100
Embed_dim	128
Num_heads	2
Feed_dim	128

5. 模型训练

6. 模型评估

7. 使用测试集进行预测

4.实验结果分析

1.Transformer:

Learning_rate=0.0005

acc:0.8616,pre:0.8646,rec:0.8671,f1:0.8658

	precision	recall	f1-score	support
0	0.8585	0.8558	0.8571	326
1	0.8646	0.8671	0.8658	346
accuracy			0.8616	672
macro avg	0.8615	0.8614	0.8615	672
weighted avg	0.8616	0.8616	0.8616	672

Learning_rate=0.001

acc:0.8497,pre:0.8708,rec:0.8493,f1:0.8599

	precision	recall	f1-score	support
0	0.8259	0.8502	0.8379	307
1	0.8708	0.8493	0.8599	365
accuracy			0.8497	672
macro avg	0.8484	0.8497	0.8489	672
weighted avg	0.8503	0.8497	0.8498	672

2. CNN:

Learning_rate=0.0005

```
acc:0.8631,pre:0.8646,rec:0.8696,f1:0.8671
              precision    recall  f1-score   support

           0     0.8615    0.8563    0.8589       327
           1     0.8646    0.8696    0.8671       345

    accuracy                         0.8631       672
   macro avg     0.8630    0.8629    0.8630       672
weighted avg     0.8631    0.8631    0.8631       672
```

Learning_rate=0.001

```
acc:0.8527,pre:0.8691,rec:0.8571,f1:0.8631
              precision    recall  f1-score   support

           0     0.8339    0.8474    0.8406       308
           1     0.8691    0.8571    0.8631       364

    accuracy                         0.8527       672
   macro avg     0.8515    0.8523    0.8518       672
weighted avg     0.8529    0.8527    0.8528       672
```

5.总结

实验结果表明 CNN 和 Transformer 都能够在该任务中取得良好效果的模型。

CNN 的优势在于对局部特征的提取较为擅长，适合处理序列中的局部模式。所以对于较小的数据集，CNN 通常比较容易训练，因为它们的参数共享机制可以在小数据集上更好地泛化。同时 CNN 可以有效地捕捉文本中的 n-gram 特征，这对于一些文本分类任务（很有帮助。

Transformer 的优势在于具有全局注意力机制，能够捕捉长距离依赖关系，对于处理长文本序列有一定优势。Transformer 在大规模数据集上通常表现出色，但在小数据集上可能会有过拟合的问题。如果预训练的语言模型（如 BERT、GPT）在大数据集上进行了预训练，然后在小数据集上进行微调，效果可能会更好。

因此，由于本次任务所使用的数据集较小，理论上 CNN 会取得一个更好的效果，并且实验数据也表明，CNN 的确要优于 Transformer。

所以如果任务较为简单，数据集较小，可以首选 CNN，因为其相对较轻量且容易训练。如果任务涉及到更复杂的语言建模或者需要捕捉长距离依赖关系，可以考虑使用预训练的 Transformer 模型，然后在小数据集上进行微调，这样的效果会更好。