|  |
| --- |
| **机器学习** |
| Quora Insincere Questions Classification |
| 主 研 人：岳天昕  参 研 人： |
| 审 核 人：  方 向：工业算法案例研究  版 本 号：A |
|  |

**版本更新**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **更新描述** | **作者** |
| 2018/10/31 | A | 初稿 |  |

**选题表格**

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 竞赛名 | 竞赛背景描述（50字以内） | 类型（分类/回归） |
| 2019/3/11 | Quora Insincere Questions Classification | 当今主流网站都存在一个问题：如何处理含有害与分裂言论的内容。Quara（类似知乎）希望参赛者开发算法来过滤虚假或含虚假背景或含误导性内容的问题。 | 分类 |

目录

[1. 背景描述 4](#_Toc25575_WPSOffice_Level1)

[1.1 竞赛赛题描述 4](#_Toc5940_WPSOffice_Level2)

[1.2 评估指标描述 5](#_Toc12357_WPSOffice_Level2)

[2. 数据来源及描述性统计分析 5](#_Toc5940_WPSOffice_Level1)

[2.1 大赛数据来源 5](#_Toc22683_WPSOffice_Level2)

[2.2 数据的描述性统计 5](#_Toc16174_WPSOffice_Level2)

[2.2.1 数据基本情况描述： 5](#_Toc5940_WPSOffice_Level3)

[2.2.2 数据字段介绍： 5](#_Toc12357_WPSOffice_Level3)

[2.2.3 数据描述性统计 6](#_Toc22683_WPSOffice_Level3)

[3. 优秀算法思路 6](#_Toc12357_WPSOffice_Level1)

[3.1 方案一 6](#_Toc32293_WPSOffice_Level2)

[3.1.1 方案一数据预处理及特征工程部分方案 6](#_Toc16174_WPSOffice_Level3)

[3.1.2 方案一模型设计、建立部分方案 6](#_Toc32293_WPSOffice_Level3)

[3.1.3 方案一结果、排名等 6](#_Toc29922_WPSOffice_Level3)

[3.1.4 方案一算法流程图 6](#_Toc4787_WPSOffice_Level3)

[3.2 方案二 7](#_Toc29922_WPSOffice_Level2)

[… 8](#_Toc4787_WPSOffice_Level2)

[4. 算法比较 8](#_Toc22683_WPSOffice_Level1)

[表4-1 算法比较 8](#_Toc7253_WPSOffice_Level2)

[5. 总结与展望 8](#_Toc16174_WPSOffice_Level1)

[5.1 总结 8](#_Toc23198_WPSOffice_Level2)

[5.2 建模思路 8](#_Toc16619_WPSOffice_Level2)

1. 背景描述

今天任何一个主要网站的存在问题是如何处理有害和含分裂言论的内容。 Quora希望能直接解决这个问题，让他们的平台成为用户可以安全地与世界分享知识的地方。  
Quora是一个让人们相互学习的平台。 在Quora上，人们可以提出问题，并与提供独特见解和高质量答案的人们有交流机会。 此次竞赛的一个关键挑战是如何筛选掉虚假问题 - 那些建立在虚假前提下的问题，或者打算发表声明而不是寻求有用答案的问题。

在本次比赛中，参赛者将开发识别和标记虚假问题的模型。 到目前为止，Quora已经使用机器学习和人工审查来解决这个问题。 在您的帮助下，他们可以开发更具可扩展性的方法来检测有害和误导性内容。  
  
这是你大规模对抗在线巨魔的机会。 帮助Quora坚持“善良，尊重”的方针，使其继续成为分享和发展知识的地方。

**1.1 竞赛赛题描述**

在本次比赛中，您将预测Quora上提出的问题是否真诚。  
  
一个虚伪的问题被定义为一个旨在发表声明而不是寻找有用答案的问题。一些可以表明问题不真实的特征：  
  
    具有非中性色调  
        夸张的语气强调了一群人的观点  
        是修辞的，意味着暗示关于一群人的陈述  
    是贬低或煽动性的  
        建议针对受保护阶层的人提出歧视性观点，或寻求确认陈规定型观念  
        对特定的人或一群人进行贬低的攻击/侮辱  
        基于关于一群人的虚假前提  
        贬低无法明确衡量的特征  
    不是基于现实  
        基于虚假信息，或包含荒谬的假设  
    使用性内容（乱伦，兽交，恋童癖）来获得震撼价值，而不是寻求真正的答案

**1.2 评估指标描述**

指标：F1-score

在二元分类的统计分析中，F1分数（也是F分数或F-分数）是测试准确度的度量。 它考虑了测试的精度p和召回率r来计算得分：p是正确的阳性结果的数量除以分类器返回的所有阳性结果的数量，r是正确的阳性结果的数量除以 所有相关样本的数量（所有样本应该被确定为阳性）。 F1得分是精度和召回的调和平均值，其中F1得分在1处达到其最佳值（完美精确度和召回率），在0处达到最差值。

在预测目标和观察目标之间的F1分数上评估提交。

1. 数据来源及描述性统计分析

**2.1 大赛数据来源**

大赛数据由quora公司提供

**2.2 数据的描述性统计**

**2.2.1 数据基本情况描述：**

训练数据包括被询问的问题，以及是否被识别为不真实（目标= 1）。

由于是真实数据，其中包含了一些噪音点。  
请注意，数据集中问题的分布不应被视为代表Quora上提出的问题的分布。部分原因是由于采样程序和已应用于最终数据集的筛选措施的组合。

**2.2.2 数据字段介绍：**

做成表格形式（如果有多个数据表，请做多个数据字段介绍表格），格式如下：

**表2-1 训练数据表字段介绍**

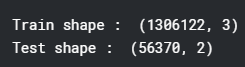
|  |  |  |  |
| --- | --- | --- | --- |
|  | 变量含义 | 变量类型 | 缺失率 |
| **qid** | 唯一问题标识符 | **文字描述** | 0% |
| **question\_text** | Quora 问题文本 | 文字描述 | 0% |
| **target** | 目标值：标记为“insincere”的问题的值为1，否则为0 | **离散** | 0% |

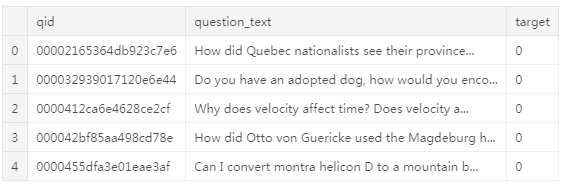
**表2-1 测试数据表字段介绍**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 变量含义 | 变量类型 | 缺失率 |
| **qid** | 唯一问题标识符 | 文字描述 | 0% |
| **question\_text** | Quora 问题文本 | 文字描述 | 0% |

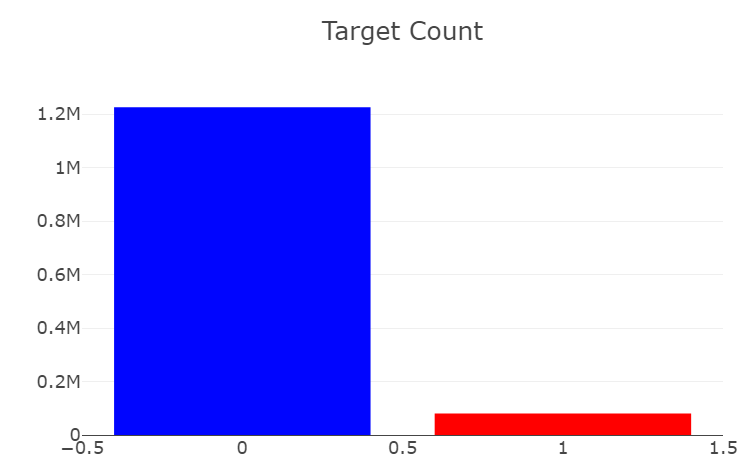
**2.2.3 数据描述性统计**

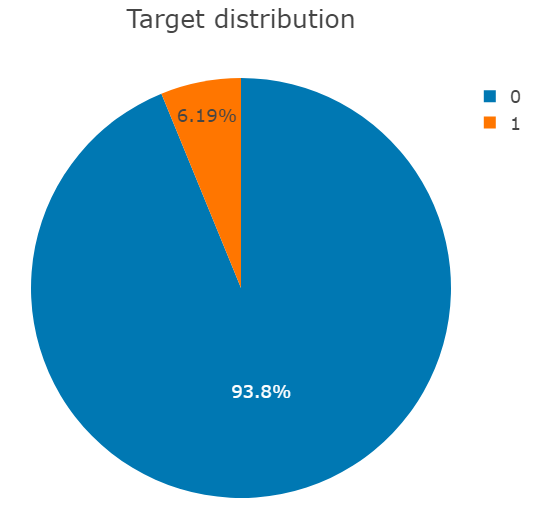
数据集总览：





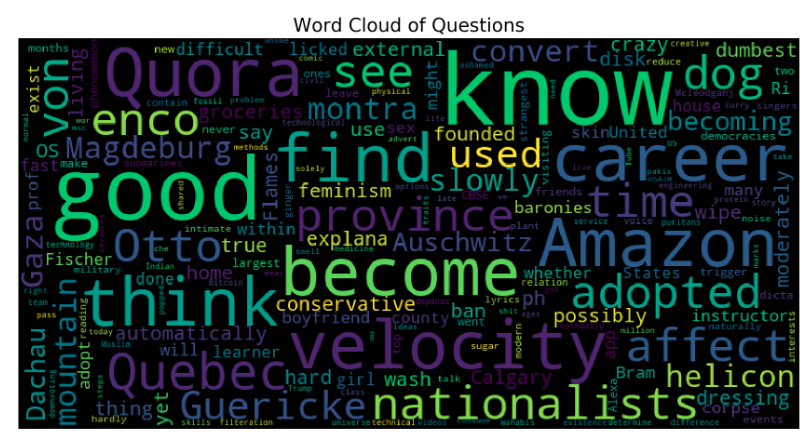
目标值分布：



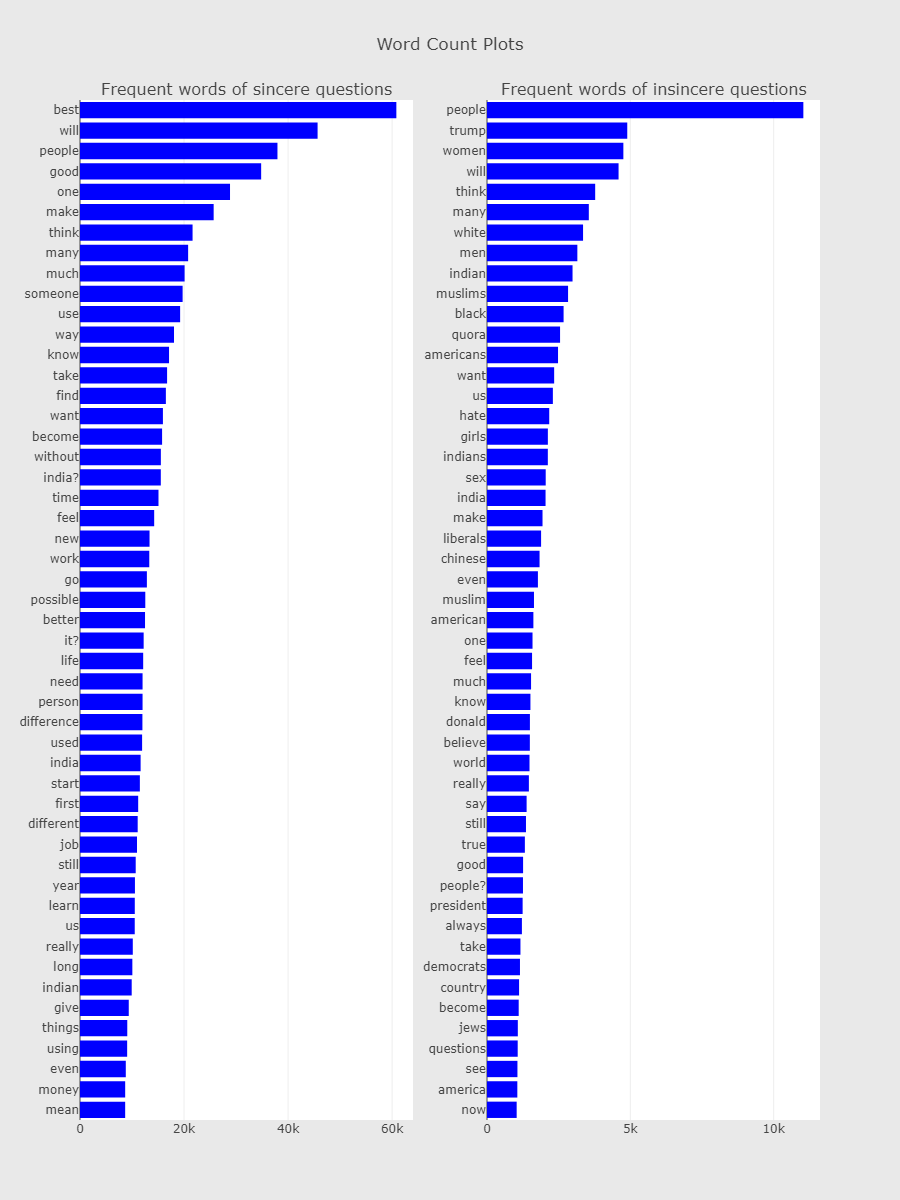


因此，大约6％的训练数据是需被筛选掉的问题（目标= 1）

**World cloud：**  
现在让我们通过在'question\_text'列上创建一个词云来查看数据中经常出现的单词。



这里似乎有各种各样的词。可能分别查看每个类中最常用的单词是个好主意。

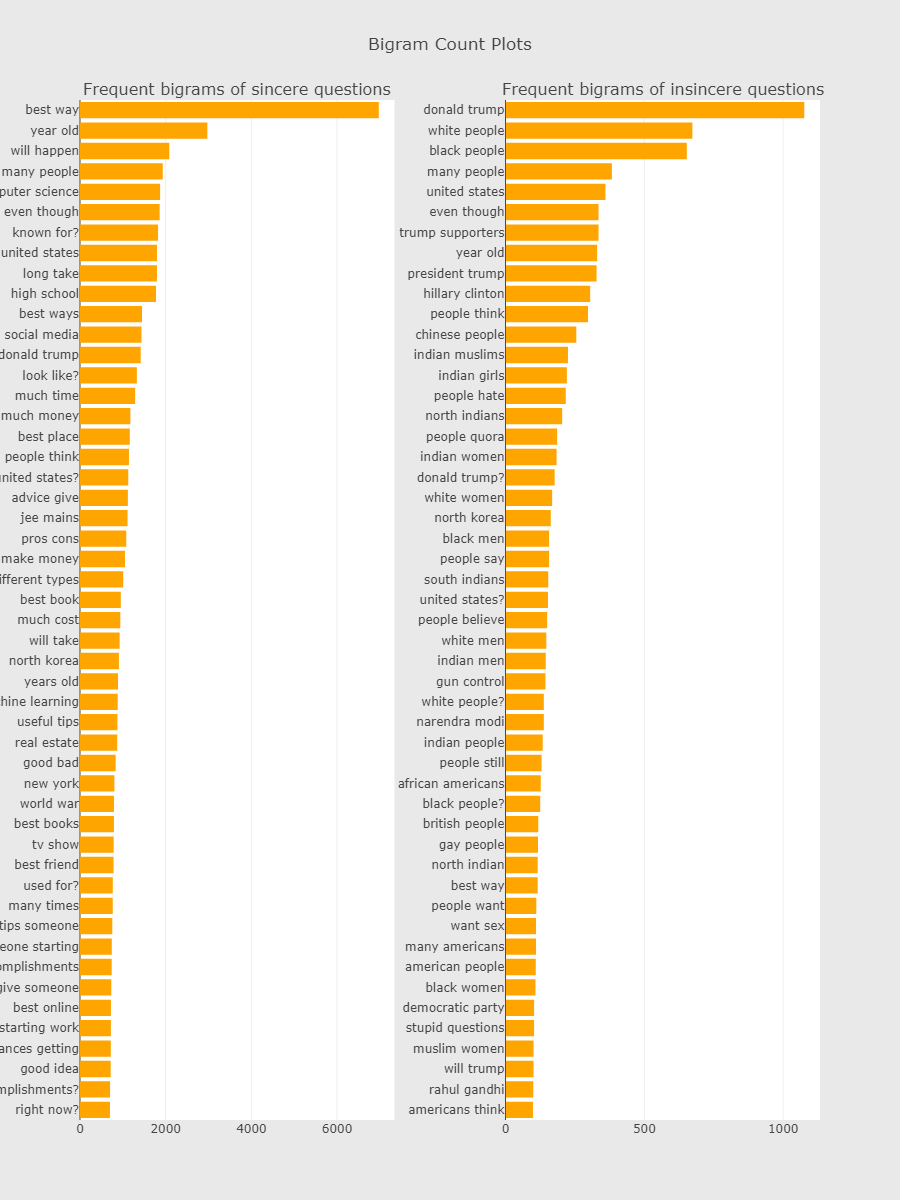


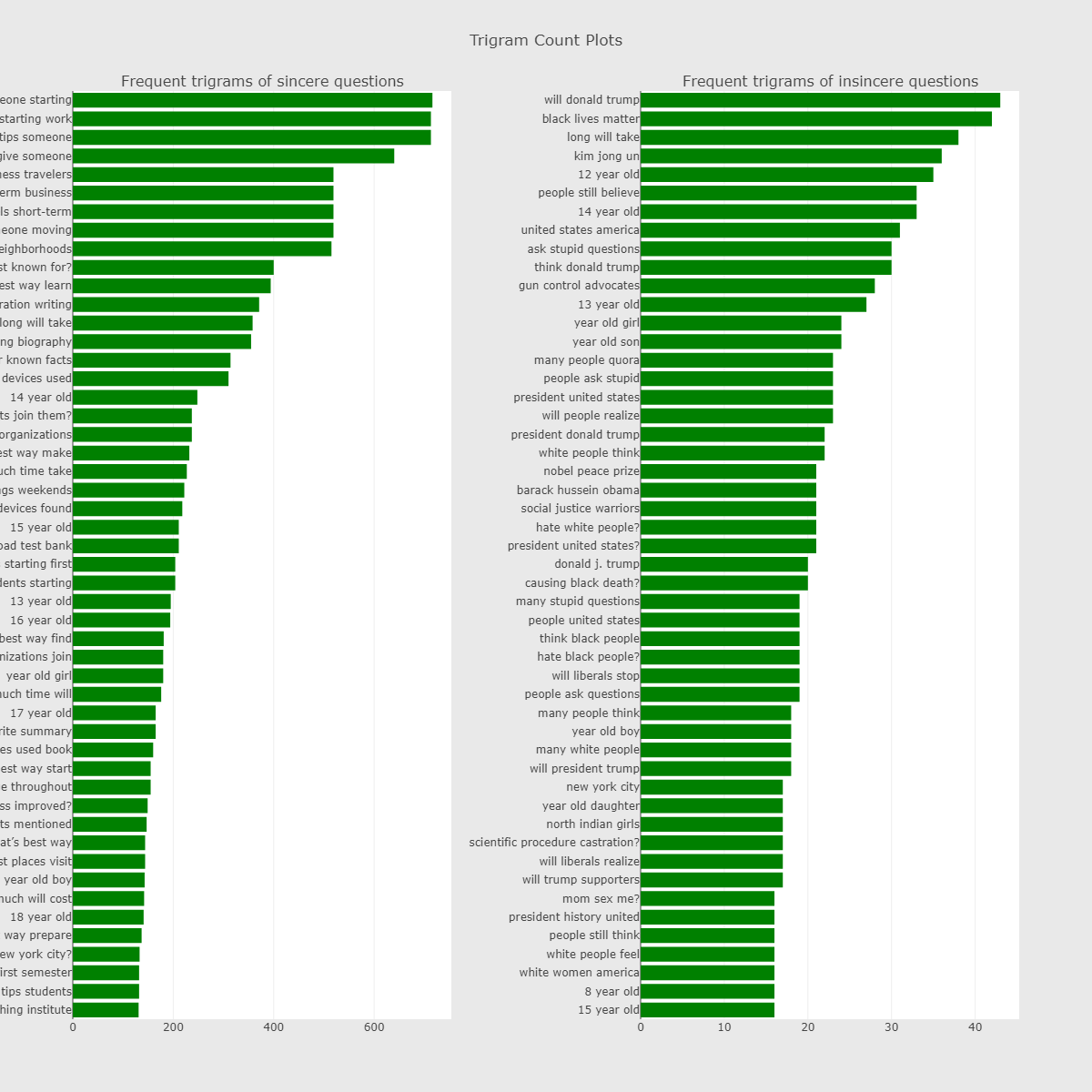
观察到：

一些顶级单词在“人”，“将”，“思考”等类中都很常见

在排除最常见的常见问题后，真诚问题中的其他热门话题是“最好”，“好”等

在排除常见问题之后，在虚假问题中的其他顶级词是“特朗普”，“女性”，“白色”等





3. 优秀算法思路

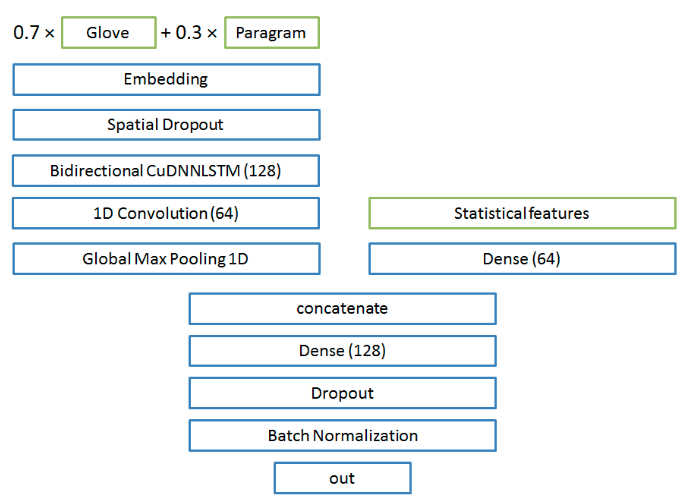
**3.1 方案一**

**3.1.1** 方案一数据预处理及特征工程部分方案

答主未解释

**3.1.2** 方案一模型设计、建立部分方案

**Model structure 模型结构**  
这个模型基本上是单个Bi-LSTM 128，后面跟着一个内核大小为1的Conv1D，之后是GlobalMaxPooling，加上额外的丢失率最小的丢失层。我们还使用了一些统计功能。



**Embeddings 嵌套**

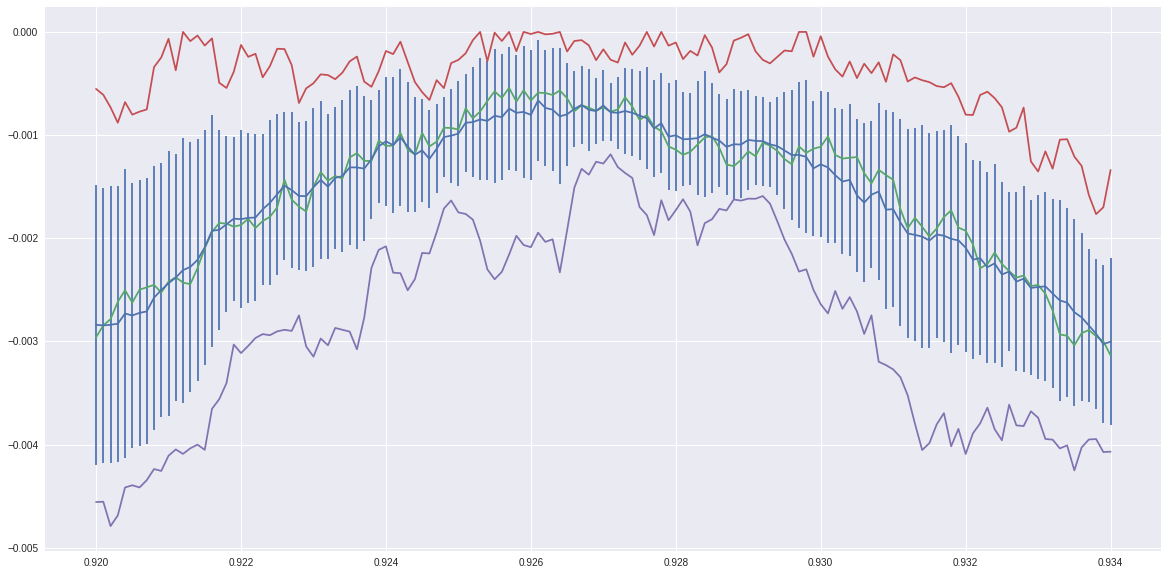
首先在训练集和测试集中提取出来文字词组。在比赛前我们进行可简单的数据预处理，然后按空格分割（spacy和nltk导致类似的性能）。我们没有采用小写，而是保持大写，且没有限制词汇。对于嵌入，我们使用glove和para且提高了一点glove的权重。现在最重要的是为我们的单词找到尽可能多的嵌入，我们做了以下步骤：例如检查单词的单复数、检查小写字母嵌套、删除特殊标记等。对于公共测试数据，我们仍有大约50万的词汇不能找到相应的嵌入。尽管我们尝试了一些不同的策略来处理OOV标记，最后还是使用了一个带有单个随机嵌入向量的OOV。

**Threshold阈**

我们花了很多时间试图找出选择良好分类阈值的好策略。随着时间的推移，我们看到估计验证数据的阈值然后将其应用于测试数据并不真正起作用。最佳阈值存在很大差异。所以我们所做的是尝试在CV上找到一个固定的阈值，使该阈值产生与最佳阈值的最小偏差。为了最终提交，我们选择了最佳的CV阈值。这也使我们能够将模型拟合到完整数据上，而无需依赖随机分割和较少的训练数据。

以下是可视化过程。

在x轴上，我们绘制了不同的固定阈值，在y轴上，我们看到使用此固定阈值与折叠的最佳F1分数的偏差（参见下面的CV章节）。蓝线是平均值，绿色是中位数，紫色是最小值，红色是最大值，条形是标准。因此，例如，如果我们选择0.927范围内的阈值，我们预计与选择最佳阈值（我们无法对测试数据做出）相比，F1得分不会更差（约0.001）。在实践中，这当然可能会进一步偏离，我们也可能看到PLB的偏差更大。



**Runtime tricks 运行时的小技巧**

我们的目标是尽可能多地组合模型。为此，我们需要改进运行时间，且实现以下重要事情。

我们不根据整个数据将序列填充到相同的长度，而只是在批处理级别上。这意味着我们分别在数据生成器级别为每个批次时执行填充和截断，以便批处理中句子的长度可以变化。此外，我们不通过基于批次中最长序列的长度截短，而是基于序列内95％百分比的长度截短进一步改进了这一点。这大大改善了运行时间，并且在单个模型级别上保持了非常强大的准确性，并通过能够平均更多模型来改进它。

**Fitting 拟合**

我们使用Nadam优化器的一个周期策略（你可以通过将步长更改为总迭代的一半来实现典型的CyclicLearningRate）。我们选择批量为512。即使采用10或20倍更高的批量，我们也可以获得类似的结果，这与最近的快速收敛研究密切相关。有了这些更大的批量，我们甚至可以容纳近20个模型，但结果稳定接近10个模型，这就是为什么我们选择最终采用较小的批量。

但是，如果有人适当地调整它，可能仍然会留下一些发展空间。

**Multiple models 多种模型**

最后，我们通过前面提到的运行时技巧，在完整的训练数据集上设置了10多个模型。我们最好的最终私人得分甚至只有6000秒的运行时间（我认为他们使用了更好的硬件来运行），因此估计可以再容纳1-2个以上的模型。如上所述，然后我们平均了每个模型的排名预测，并使用我们指定的预测阈值。

**Embrace the randomness 考虑随机性**

在本次比赛中有必要使用CUDNN图层，因此有一些随机性可能会不时令人沮丧。我看到很多人试图修复种子等等，有些人声称他们可以通过使用Pytorch完全消除随机性（我仍然不相信这个，因为CUDNN有原子操作）。然而，如前所述，本次比赛中的一个良好的解决策略是结合多个模型并最终得到一个良好的整体，这些模型应该彼此有点不同。因此，具有不同的随机初始化等可能是有帮助的。看到人们将种子设置为超参数我们觉得很奇怪。

**CN Evaluation CN评估**

我看到很多人在这场比赛中做错了，我们也只是在一段时间后才想到这一点，就是相信他们的单一评估。但是，在本次比赛中，将（平均）多个模型（在我们的例子中是同一模型）结合起来至关重要。这意味着我们的CN评估如下所示。我们进行k倍分割（大多数是10倍）并且在相同的训练分割中将相同的模型拟合到v次，然后在单次折叠时连续评估它。因此，对于第一次拆分，我们首先拟合一个模型并对其进行评估，然后是第二个模型并评估平均值等等。我们重复这10次折叠，例如，整体上100个模型拟合，然后我们可以看一下v模型集合的所有折叠的中值或平均值。这样做的原因是f1分数在你的分割上会非常不同。对于一个10％的分割，最终可能最多为0.72，另一个最终可能为0.705。因此重复分割10次，为每次分割拟合相同的模型v次，然后查看大图给我们最好的整体评价。这个例程帮助我们将各个解决方案相互比较。

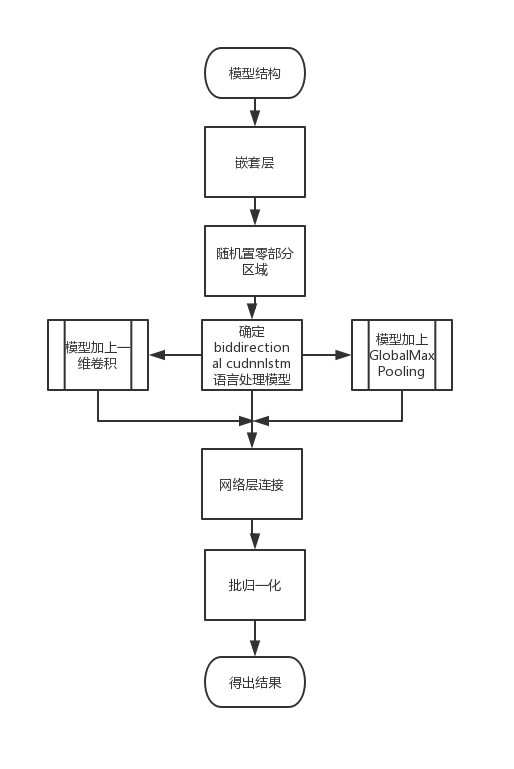
**underfitting欠拟合**  
在最终提交前大约2周，我们的结果变得非常稳定，以至于改变过程并没有改变结果。像寻找更多OOV嵌入向量之类的东西导致相同的结果，使用稍微不同的层最终变得相似，以及其他东西。这有点令人沮丧，但最终事情已经解决了。最后，重要的是在过拟合和欠拟合之间找到一个良好的平衡。过拟合导致良好的单一模型表现，但是对于组合模型更糟糕，反之亦然。例如，如果你的模型过度拟合，可以有许多不同的解决方案来解决这个问题，例如，添加辍学，或减少词汇大小，或降低模型复杂性等。

**3.1.3** 方案一结果、排名等

TOP1

0.71323

**3.1.4** 方案一算法流程图

（这个部分需要自己整理一下绘制，流程图可以用processon画）**图3-1**

**3.2 方案二**

**3.2.1** 方案一数据预处理及特征工程部分方案

我在字符周围插入了空格，除了字母和数字。

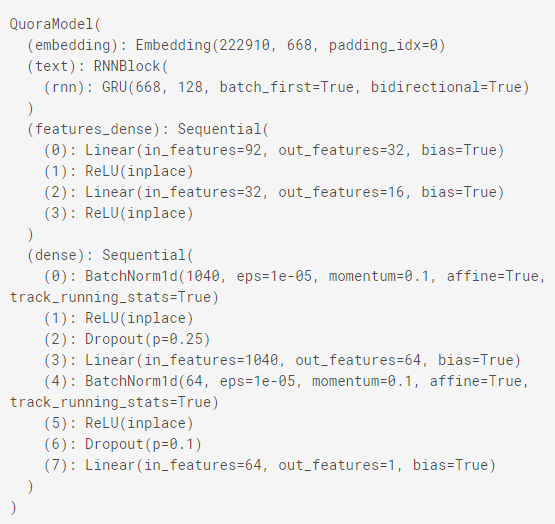
然后，我使用了eras tokenizer，它只用空格分割。

在标记化之后，我对OOV单词应用了拼写校正。拼写校正算法的粗略思想是在忽略案例时找到具有0或1 levenshtein距离的单词。

在我的情况里，设计包括上述拼写校正在内的预处理并没有如此多地改变CV得分。

**3.2.2** 方案一模型设计、建立部分方案

主要部分是1层Bi-GRU，其隐藏层大小为128，后接最大池，平均池和第一/最后位置输出的串联。另一部分是用于统计特征的密集层。将2个网络部分的输出连接起来，然后馈送到密集层。



**嵌套：**

我使用glove和维基新闻词预训练词嵌入。我还使用fastText在竞赛数据集（训练+测试）上训练64维字嵌入。

除了它们之外，我还为字嵌入添加了4个二进制特性。最后，我连接了所有这些操作。在训练期间，字嵌入的参数是被冻结的。

**统计特征：**

单词数量

唯一单词的数量

字符数

大写字母的数量

一袋字符：由CountVectorizer实现（ngram\_range =（1,1），min\_df = 1e-4，token\_pattern = r'\ w +'，analyzer ='char'）

**输入RNN的序列长度：**

为了加快培训速度，我调整了每批次的序列长度。

在训练时，如果最大长度超过55，我通过应用预截断来使用批次中的最大序列长度或55长度。当预测测试时，如果长度超过70而不是55，则应用截断。

多亏了这个技巧，我能够在内核上训练6个模型。

**模型训练：**

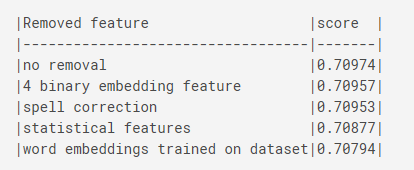
我将最初学习率定为0.001，每个迭代后学习率乘以0.8。  
我获得了批量大小为256的最佳CV分数。但是，批量大小在分数和训练时间之间需要进行权衡。 随着我增加批量大小，CV得分变差，训练却变得更快。 我通过权衡内核上的CV分数和训练时间来最终决定批量大小为320。

**模型集成：**

我做了6个模型的平均种子。 我训练了6种不同种子的模型。 每个模型经过5次迭代。 每个模型都在整个训练数据集上进行训练，换句话说，我没有使用k-fold分割来训练不同的模型。  
我平均了6个模型的预测。 然后，我做了最终的二进制预测，阈值为0.36。

**本地验证：**

我为本地验证做了5折CV。对于每个折叠，我使用整合后的模型的预测而不是1个模型的预测，以获得更稳定的CV，和更接近的CV得分与LB得分以及更优化的超参数搜索。

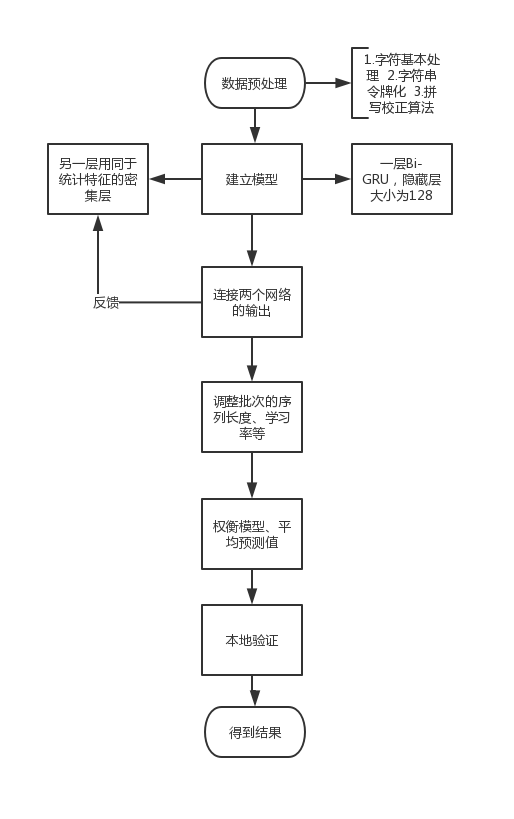
**CV得分：**  


**3.1.3** 方案一结果、排名等

TOP2

0.71275

**3.1.4** 方案一算法流程图



4. 算法比较

**表4-1 算法比较**

|  | **评估指标** | **特征工程** | **基础算法** | **基本库** |
| --- | --- | --- | --- | --- |
| **算法1** | **F1-Score** | **Wordcloud** | **Bi-LSTM128** | **Keras**  **sklearn** |
| **算法2** | **CV评估** | **keras tokenizer，**  **CountVectorizer** | **RNN** | **Keras** |

5. 总结与展望

**5.1 总结**

两种方案都提到在数据预处理方面应该尽可能保留数据的原始特征，并且使用多种模型训练。第一名提到，自己队伍获胜的一个原因是在权衡最终结果时对模型运用十次折叠100次拟合的CN评估对多个方案进行比较，最后选出最优结果，而不是相信单一评估。

一二名队伍都在模型的调整上有独特的策略，比如调整RNN序列长度、学习率，在多个相反趋势的参数之间进行很好的权衡，最终训练出优秀的模型。

**5.2 建模思路**

