小样本数据分类任务 综合实践

装

订

线

目录

装 订 线

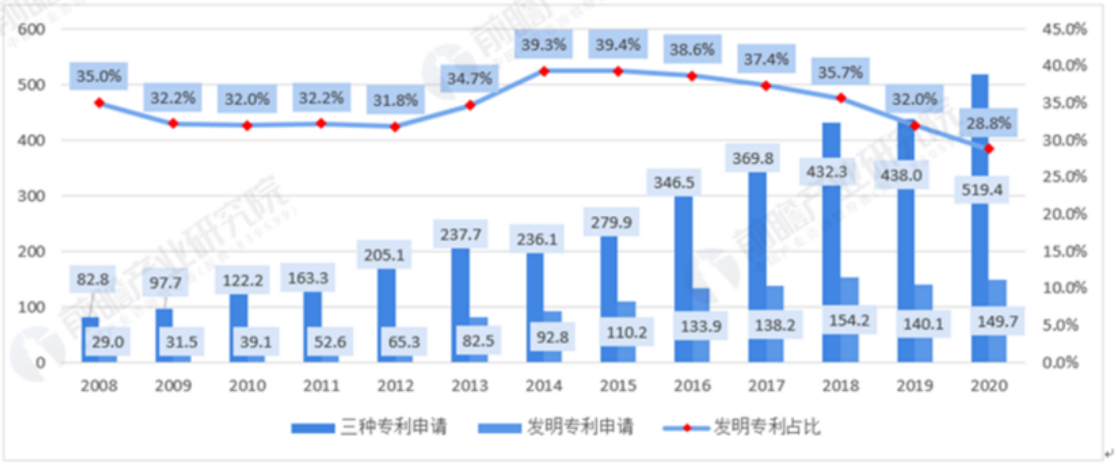
1 选题任务描述

1.1 题目介绍

1.1.1 赛题背景

创新是国家发展、民族复兴的不竭动力。近年来，随着政策扶植、国家工业化水平和国民教育水平提高，我国的专利申请量快速增长，专利检索、查新、管理等需求也不断增加。为了满足以上需求，提升专利服务质量，通常需要建立多个维度的专利分类体系。常见的分类体系有国际专利分类(IPC)、联合专利分类(CPC)、欧洲专利分类(ECLA)等，但是这些分类体系比较复杂，专业性强，对非 IP 人员而言使用有一定的困难。智慧芽作为国际领先的知识产权 SaaS 平台，根据用户的搜索习惯等因素，制定了一套新的专利分类体系，从而提升用户的使用体验。

图表1：2008-2020中国专利申请量走势及发明专利情况(单位：万件，%)



1.1.2 赛题任务

比赛方公开 958 条专利数据，包括专利权人、专利标题、专利摘要和分类标签，其中分类标签经过脱敏处理，共 36 类。要求选手设计一套算法，完成测试数据的分类任务。本次赛题公布的训练数据量较小，属于基于小样本训练数据的分类任务。小样本分类任务作为近年来研究的热点问题，学界提出了远程监督、数据增强、预训练模型、PET 范式等方案。

1.2 关键技术与难点

（1）小样本学习

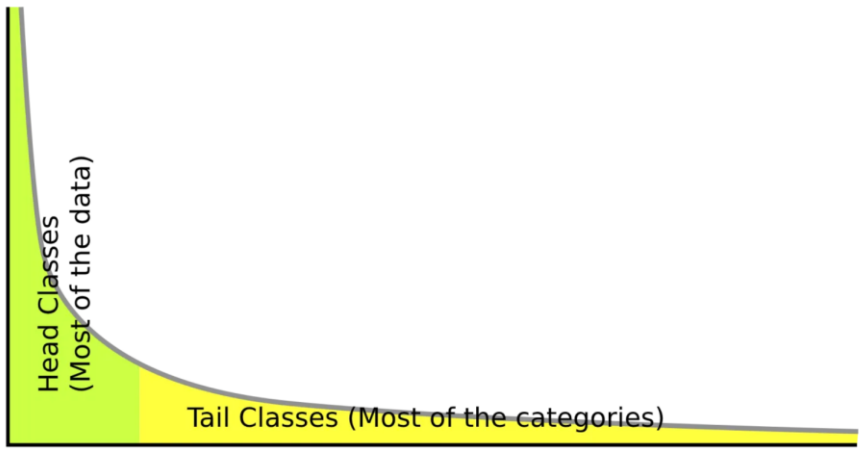
小样本学习，又称少样本学习（few-shot learning），是在只有目标少量训练样本的环境中，如何训练一个可以有效地识别这些目标样本的一类模型。在现实场景中，某些类别只有少量数据或少量标注数据，而对无标签数据进行标注将会消耗大量的时间和人力。而人类只需要通过少量数据就能做到快速学习，例如,一个五六岁的小孩子从未见过斑马,但如果给他看过一张斑马的图像或者告诉他斑马是带有黑白条纹的马,在当他看到真正的斑马时,就会马上认出这是“斑马”，而这就是传统机器学习与人类学习之间的差异，也因此启发了小样本学习领域的技术方向，即如何从

少量的样本中学习到解决问题的方法。

小样本学习最大的技术难点在于训练过程中可能产生的过拟合问题，究其原因是因为少量的数据并不能很好地反映大量数据的真实分布情况，导致模型过多地在高维空间中训练，从而产生“维度灾难”。在本赛题背景下，此问题最典型的特征是在训练过程中的 accuracy 很高，而一旦提交测试准确率就会骤降。因此，如何有效地缓解过拟合问题，是本赛题中的最为关键的技术难点，较为有效的方法是采用基于数据增强（Data Augmentation）、基于预训练模型（迁移学习）等。

（2）长尾问题

后来，我们还发现本项目中隐藏了一个问题，就是训练集数据标签不平衡，也称之为长尾问题。也就是说有部分类别的数据较多，而相对地，一些类别的数据较少，可能只有 3、4 条。这导致训练出的模型会很大程度上依赖于分类标签的分布进行预测，少类标签生存空间小，从而降低了模型的准确性。



长尾分布

本次报告将在上一次报告的基础上针对上述两个问题的解决方案和技术路线进行详细地介绍，并会阐释我们的实践和思考。

2 任务进度安排

2.1 赛题的时间安排

- 2022/08/29-11/09，初赛阶段
- 2022/08/29（12:00），发布大赛赛题，选手可登录大赛官网报名；
- 2022/09/05，开启初赛线上评测，选手可在线提交结果文件至竞赛平台，每日每队最多可提交 3 次，测评系统将自动评测得分并同步更新至排行榜。排行榜上将记录选手的最高成绩，相关团队必须自行保存最高成绩作品的源代码以备审核；
- 2022/11/04（12:00），截止报名组队；
- 2022/11/07（24:00），截止初赛 A 榜作品提交；初赛 A 榜评测结束后，所有选手均可进行初

赛 B 榜评测；

▪ 2022/11/08，公布 B 榜测试集，本赛题所有参赛选手务必在 11 月 08 日 24:00 前，在本赛题“赛题数据”页下载 B 榜测试集；

▪ 2022/11/09（00:00-24:00），初赛 B 榜作品提交，参赛者可在 B 榜当天提交 3 次，但仅以每支团队当天最后一次提交进行评测，决赛入围资格以 B 榜线上最终成绩为准（B 榜排行榜展示的成绩），若团队没有进行 B 榜提交，则无法晋级后续比赛，B 榜 TOP5 团队经复现审核后入围决赛。

▪ 2022/11/10-12/31，决赛阶段

▪ 2022/11/10-11/20，复现及决赛资料提交，拟入围决赛的团队按照要求提交复现资料、决赛答辩评审资料等;

▪ 2022/11/26-11/27，决赛评审（线上）；

▪ 2022/12 月中旬，决赛嘉年华系列活动（时间、场地待定，请关注通知）

装

2.2 组队的进度安排

小组已经完成打榜，最终 A 榜排名 122/1426，B 榜排名 41/1426。

3 小组成员与分工

订

3.1 成员分工

线

4 解题思路

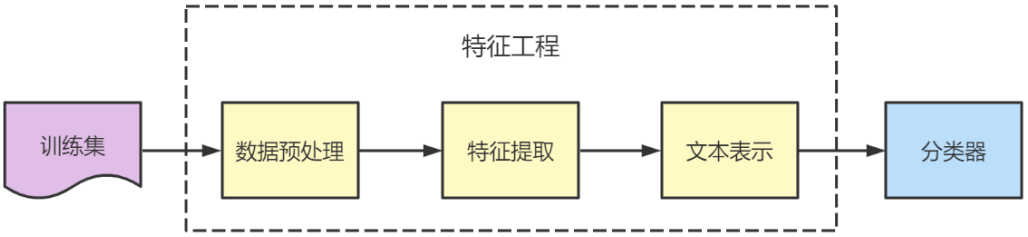
4.1 核心问题汇总

本次项目是一个针对于专利文献的小样本分类任务。其基础是一个“文本分类”任务，但是赛事方给选手添加了两层限制：较为明显的是“小样本”，即指本项目的训练集样本小，需要大量数据的普通模型很难在小样本中很好地提取出特征，导致模型准确性不高；而隐藏背后的是可能存在的“长尾”和“过拟合”问题，即模型很可能被较小的训练样本所误导，从而提取出错误的特征，降低了模型的准确性。以下将先简要介绍一下“文本分类”任务及其两个限制：

4.1.1 “文本分类”任务

文本分类是，是指计算机将载有信息的一篇文本映射到预先给定的某一类别或某几类别主题的过程，实现这一过程的算法模型叫做分类器。文本分类问题算是自然语言处理领域中一个非常经典的问题。根据预定义的类别不同，文本分类分两种：二分类和多分类，多分类可以通过二分类来实现。从文本的标注类别上来讲，文本分类又可以分为单标签和多标签，因为很多文本同时可以关联到多个类别。根据上述定义和题目要求，我们可知：本项目是一个对于段落级别文本，基于其主题的单标签多分类任务，共有 36 个标签类别，且每条数据只能有一个标签。

传统来看，“文本分类”任务可以被划分成以下几个部分，即数据预处理、特征提取、文本表示、分类器（如下图所示）。



4.1.2 小样本学习

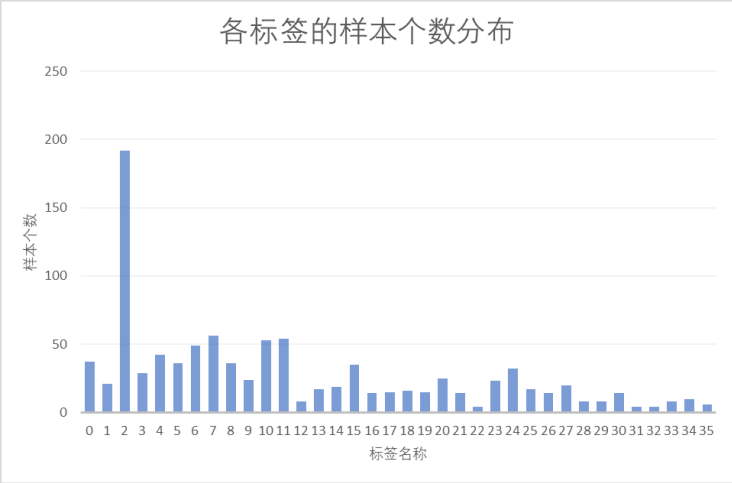
小样本学习也叫做少样本学习(low-shot learning)，其目标是从少量样本中学习到解决问题的方法。随着大数据时代的到来，深度学习模型在图像分类、文本分类等任务中取得了卓越的成果。但深度学习模型的成功，很大程度上依赖于大量训练数据。而在现实世界的真实场景中，某些类别只有少量数据或少量标注数据，而对无标签数据进行标注将会消耗大量的时间和人力。因此我们需要小样本学习，让模型在较少的训练样本下也能够有很好的准确率。

对于本项目，我们一共有 979 条专利数据，其中每条数据长度大约是 100~300 字。而分类类别有 36 个，因此对于每个训练类别的专利数据条数很少，使用通常方式模型很难达到好的效果。

4.1.3 数据不平衡

数据不平衡通常反映出数据类别的不均衡，其指的是：对于某些类别数据，其在训练集中的数量较多；而对于另一些类别的数据，其在训练集中的数据量较少。这会导致训练时，模型会把训练集的数据分布也纳入模型训练的特征中，然后由于某个类别的数据量很大，模型在进行分类时，就会自然而然地扩大该数据标签的范围，从而将很多不是该标签的数据分成该类。

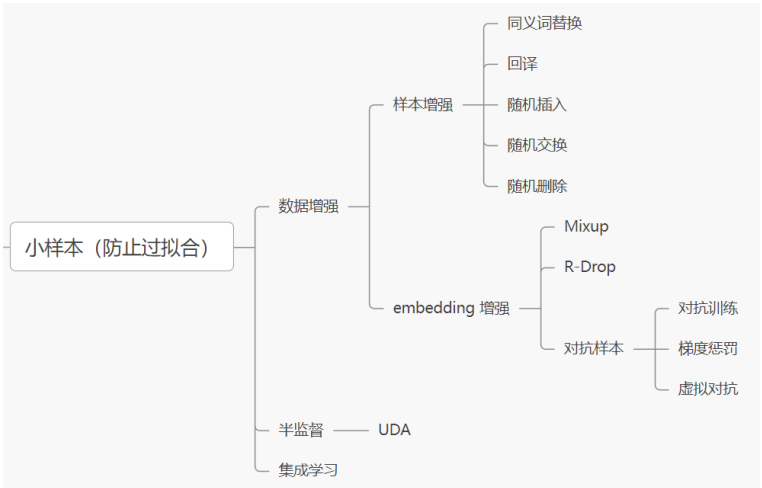
在本项目中，我们对于各个标签的样本个数分布进行了统计，结果如下图所示：



其中，标签编号为 2 的类别数据量接近 200，而部分类别（如标签编号为 31、编号为 32）数据量只有个位数条。很显然，该数据分布是不平衡的，形成我们常说的“长尾”问题。对于该问题，我们也进行了一些探究，形成了一套具有实践性的方案。

4.2 小样本学习

小样本学习有很多方法 tricks 值得尝试，我们列出如下框架：



这个框架并不包含所有处理小样本的方法，我们所列出的只是我们认为值得尝试且易于完成的。其他方法（例如：PET 范式、迁移学习、元学习等）我们了解不深，没有深入研究。以下是我们整理的小样本学习方法的详细介绍。对于各个方法，我们是顺序介绍的，没有按照各个方法的逻辑和关系介绍，具体的逻辑和关系见上图。

4.2.1 样本增强

样本增强中我们主要使用了回译和同义词替换。

（1）回译

回译做法很简单，把中文翻译成英文或法文或日文，再翻译回中文，有时回译能得到语法结构不同的句子，如被动句变成主动句，有效增强了样本的多样性。当然，对于长文本的回译会导

致语句不通顺的问题，产生负面效果，因此回译一般用于短文本，或是随机对长文本中的单句进行回译，而不是把整个长文本进行回译。

（2）同义词替换。

同义词替换是维护一个同义词表，如哈工大的发布的同义词词典。在每次训练的时候，样本有一定的概率对里面的词语进行替换。

还有种做法是用词向量进行替换，如我们对"驾驶"一次进行同义词替换，发现在词向量表中，离"驾驶"余弦距离最近的一词是"行驶"，所以就把"驾驶"替换成"行驶"。最好用目前手头任务领域的文本来训练词向量。

无论使用哪种数据增强方式，增强后的样本要和实际预测的样本分布要相似，这样才能得到比较好的正向效果。

4.2.2 Mixup

Mixup 是一种非常规的数据增强方法，其源于 CV 领域，利用线性插值的方式产生更多的训练样本。假设有两个训练样本(x^i, y^i)与(x^j, y^j)，那么可以分别对输入与 label 进行插值产生训练样本$(\hat{x}^{ij}, \hat{y}^{ij})$

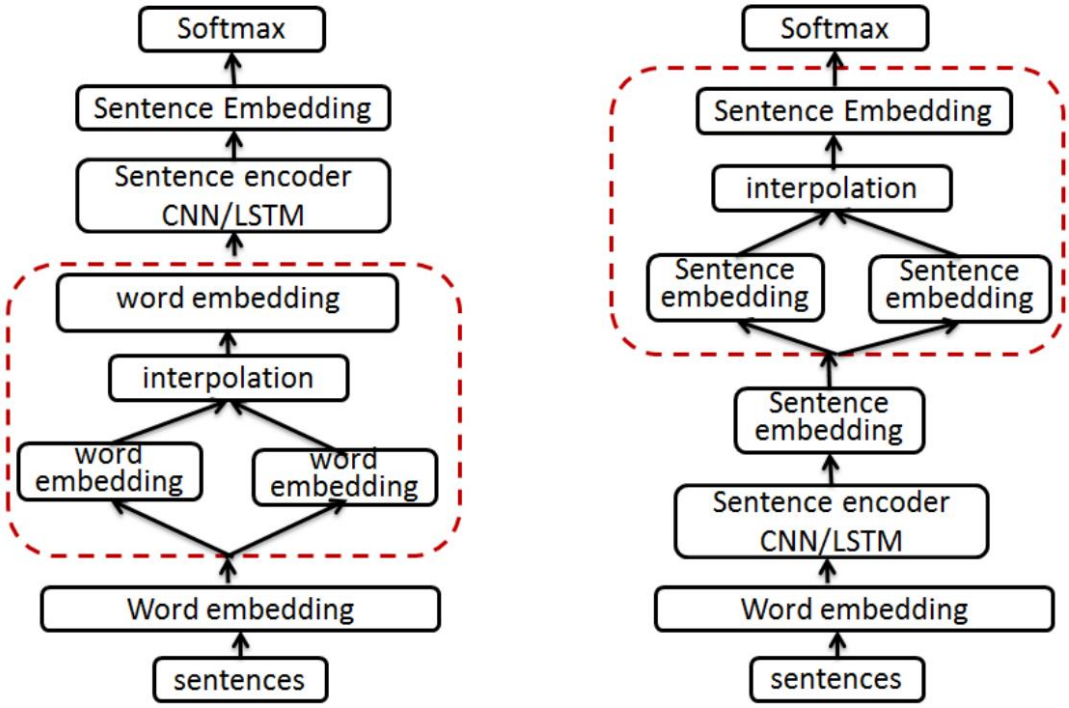
$$\hat{x}^{ij} = \lambda \mathrm{x}^{\mathrm{i}} + (1-\lambda)\mathrm{x}^{\mathrm{j}}$$
$$\hat{y}^{ij} = \lambda \mathrm{y}^{\mathrm{i}} + (1-\lambda)\mathrm{y}^{\mathrm{j}}$$

在 CV 任务中，模型的输入信息为 RGB 矩阵，因此可以直接进行该操作。但在 NLP 中，input 一般都是文本，故若是希望实现插值，则至少需要先将其变为 token 或者 embedding。两种常见的文本 mixup 方式为：

WordMixup：用 zero padding 的方式将所有句子 padding 到相同的长度，然后将两个训练样本的 word embedding 序列与 label 用上面的公式进行插值得到增强样本。这一增强发生在模型的 embedding 层。

SenMixup：在句子经过 LSTM、CNN 或者 Transformer 处理后会得到整句话的 sentence encoding，然后对两个样本的 sentence encoding 与 label 进行插值，最后将插值的结果送入网络最末端的 softmax 层进行处理。相较于前者，此类插值发生在网络的末端，因此可以复用先前 encoding 的结果，计算效率较高。

$$\hat{B}^{ij} = \lambda \mathrm{f}(\mathrm{B}^{\mathrm{i}}) + (1-\lambda) f(\mathrm{B}^{\mathrm{j}})$$
$$\hat{y}^{ij} = \lambda \mathrm{y}^{\mathrm{i}} + (1-\lambda)\mathrm{y}^{\mathrm{j}}$$

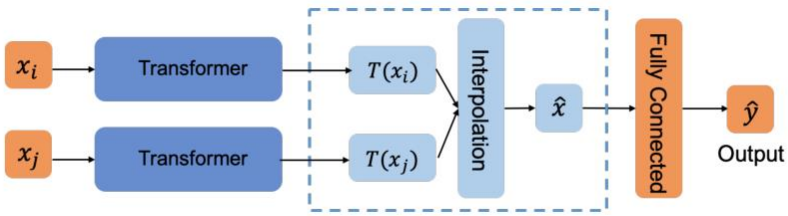


此方法在面对小样本数据时是较为有效的。Hongyu Guo 等在 2019 年就已提出了 Mixup 在 NLP 领域的运用方法，上述所介绍的便是此文中所提出的方法。而 Lichao Sun.等在 2020 年进一步将其运用于 transformer，其网络结构如下图。

4.2.3 R-Drop

R-Drop 是一种更强的正则化 Dropout 手段。正则化的一些方法在深度学习中很常见，它很好的缓解了模型过拟合的问题，提高了模型的泛化能力。那在这些正则化方法中，dropout 是应用最广泛的，dropout 提出的灵感是为了避免 co-adaptation 就是神经元之间的联合适应性。那 dropout 的操作很简单，就是在训练中 drop 掉一定比例的神经元，每个 mini-batch 学习一个瘦小的网络，最后得到一个更加强大的网络。

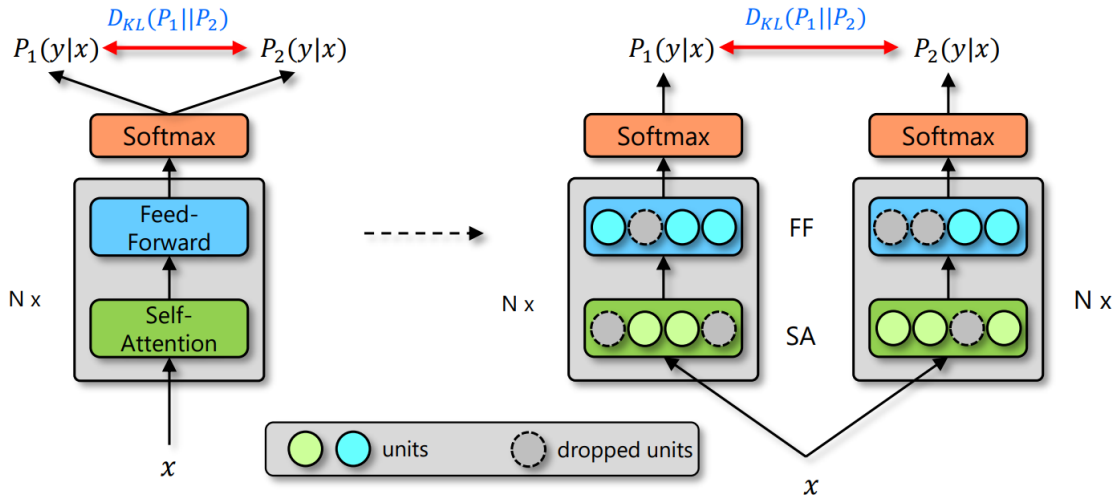
我们在训练时用到 dropout，而推理时不用，这里有个副作用就是这会造成训练和推理的不一致。由于训练时，每个瘦小的模型组成是随机的，输出分布也会不一样，那么我们可不可以在每个 mini-batch 训练时，forward 两次，而我们通过约束这两次的 forward 的输出分布，就是所说的



R-Drop。

如下图所示，每个数据样本重复经过带有 Dropout 的同一个模型，再使用 KL 散度约束两次

输出，使其尽可能一致，由于 Dropout 的的随机性，可以近似地把输入 X 走过的两次路径当作两



个略有不同的模型。

接下来，我们看下 loss 计算，我们需要计算两部分地 loss：

第一部分就是模型自有的损失函数交叉熵：

$$L_{NLL}^{i} = -\log(P_1^w(y_i|x_i)) - \log(P_2^w(y_i|x_i))$$

第二部分是两次输出之间的 KL 散度：

$$L_{KL}^{i} = (D_{KL}(P_1^w(y_i|x_i)||P_2^w(y_i|x_i)) + D_{KL}(P_2^w(y_i|x_i)||P_1^w(y_i|x_i)))/2$$

总的损失函数为

$$L_i = L_{NLL}^{i} + \alpha \cdot L_{KL}^{i}$$

具体训练过程如下所示

Algorithm 1 R-Drop Training Algorithm

Input: Training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$.
Output: model parameter w.

1: Initialize model with parameters w.
2: **while** not converged **do**
3: randomly sample data pair $(x_i, y_i) \sim \mathcal{D}$,
4: repeat input data twice as $[x_i; x_i]$ and obtain the output distribution $[\mathcal{P}_1^w(y_i|x_i), \mathcal{P}_2^w(y_i|x_i)]$,
5: calculate the negative log-likelihood loss $\mathcal{L}_{NLL}^{i}$ by Equation (3),
6: calculate the KL-divergence loss $\mathcal{L}_{KL}^{i}$ by Equation (2),
7: update the model parameters by minimizing loss $\mathcal{L}^{i}$ of Equation (4).
8: **end while**

然后我们看下训练 Algorithm 的部分，其实他就是在正常的训练基础上再 forward 一次，多算一种 kl 散度 loss。下图是他的代码，很简单，只需要再加一行 forward，我们得到两个 logits，不需要改变原有模型的代码和结构，训练也很简单。

如下为 R-drop 代码的具体实现：

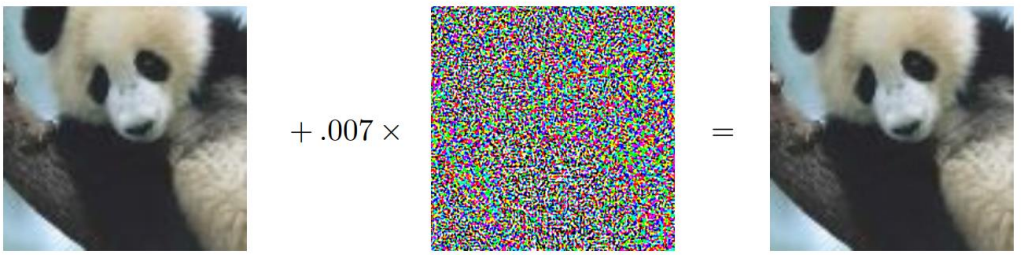
```
import torch.nn.functional as F

# define your task model, which outputs the classifier logits
```

```
model = TaskModel()

def compute_kl_loss(self, p, q, pad_mask=None):

    p_loss = F.kl_div(F.log_softmax(p, dim=-1), F.softmax(q, dim=-1), reduction='none')
    q_loss = F.kl_div(F.log_softmax(q, dim=-1), F.softmax(p, dim=-1), reduction='none')

    # pad_mask is for seq-level tasks
    if pad_mask is not None:
        p_loss.masked_fill_(pad_mask, 0.)
        q_loss.masked_fill_(pad_mask, 0.)

    # You can choose whether to use function "sum" and "mean" depending on your task
    p_loss = p_loss.sum()
    q_loss = q_loss.sum()

    loss = (p_loss + q_loss) / 2
    return loss

# keep dropout and forward twice
logits = model(x)

logits2 = model(x)

# cross entropy loss for classifier
ce_loss = 0.5 * (cross_entropy_loss(logits, label) + cross_entropy_loss(logits2, label))

kl_loss = compute_kl_loss(logits, logits2)

# carefully choose hyper-parameters
loss = ce_loss + α * kl_loss
```

4.2.4 对抗训练

4.2.4.1 对抗训练

要认识对抗训练，首先要了解对抗样本。对抗样本是指在原先样本的基础上增加一些“人类所察觉不出来”的改变以提升模型的预测效果。以下借用 Ian Goodfellow 等人的论文《Explaining and Harnessing Adversarial Examples》来加以说明：



$\boldsymbol{x}$

“panda”
57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

“nematode”
8.2% confidence

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
“gibbon”
99.3 % confidence

如图所示，左图的熊猫图片来自 ImageNet 数据集；右边熊猫图片则是经过处理的，作者向原本图片添加一个不易察觉的小向量——这个向量等于相对于输入的损失函数的梯度元素的符号，从而生成了右侧图片。显而易见，人的肉眼几乎无法发现两者的区别。但是作者使用了 GoogLeNet 分别对两张图片进行了图像分类，发现两者的分类置信度存在很大差异。右侧的样本就被称为对抗样本。

显然，我们希望自己的模型能识别更多的对抗样本，这种思想被称为对抗防御。而所谓对抗训练，则是属于对抗防御的一种，它构造了一些对抗样本加入到原数据集中，希望增强模型对对抗样本的鲁棒性。而对于我们这个小样本的 NLP 任务，添加对抗样本能帮助我们补充样本的缺失，进而可能提高我们模型的泛化能力和鲁棒性，是值得一试的策略。

下图比较清晰地向我们解释了对抗训练如何帮助提升模型的鲁棒性。引自论文《Towards Deep Learning Models Resistant to Adversarial Attacks》

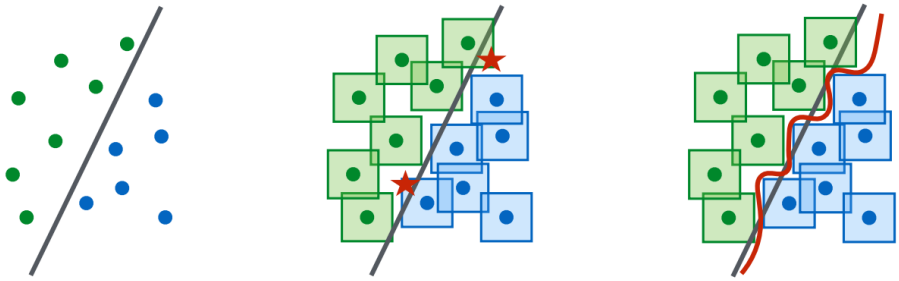


Figure 3: A conceptual illustration of standard vs. adversarial decision boundaries. Left: A set of points that can be easily separated with a simple (in this case, linear) decision boundary. Middle: The simple decision boundary does not separate the ℓ_∞-balls (here, squares) around the data points. Hence there are adversarial examples (the red stars) that will be misclassified. Right: Separating the ℓ_∞-balls requires a significantly more complicated decision boundary. The resulting classifier is robust to adversarial examples with bounded ℓ_∞-norm perturbations.

由于 NLP 任务中所输入的文本多数是独热向量，其欧氏距离相差太大，不存在所谓小扰动。所以我们可以采用论文《Adversarial Training Methods for Semi-Supervised Text Classification》中的方法，将扰动加入到 Embedding 层，如下图所示：

有实验证明，在很多任务中，在 Embedding 层进行对抗扰动能有效提高模型的性能。

针对于 NLP 项目，我们使用到了两种较为常用的对抗训练算法，分别是快速梯度法（FGM）和迭代梯度下降（PGM），以下将简要介绍一下这两个方法：

（1）FGM

FGM 是 Goodfellow 等人在 2017 年提出的，它增加了如下形式的扰动：

$$r_{adv} = \varepsilon \cdot g/\|g\|_2$$

$$g = \nabla_x L(\theta, x, y)$$

新增的对抗样本是：$x_{adv} = x + r_{adv}$；

其核心算法是：

a) 计算前向 loss，然后反向传播计算 grad（这里的梯度 grad 先保留，不更新）；

b) 拿到 embedding 层上的梯度，计算正则值；然后根据上述公式计算出扰动r_{adv}，再将r_{adv}

累加到原始的 embedding 样本上，得到对抗样本x_{adv}；

c) 根据对抗样本x_{adv}计算出新 loss，再 backward 得到对抗样本的梯度；

d) 将被修改的 embedding 恢复到原始状态 x；

e) 使用步骤 c)计算出的梯度，对模型参数进行更新。

（2）PGD

PGD 是在 FGM 之上演化而来的。不同于 FGM 一步对抗到位，PGD 采用了小步多走的策略进行对抗，即 PGD 做多次迭代，每次走一小步，每次迭代都会将扰动投射到规定范围内。其增加的扰动形式如下所示：

$$r_{adv}^{t+1} = \prod_{\|r_{adv}\|_F \le \varepsilon} r_{adv}^{t} + \alpha g(r_{adv}^{t}) / \|g(r_{adv}^{t})\|_2$$

$$g(r_{adv}^{t}) = \nabla_{r_{adv}} L(f_\theta(X + r_{adv}^{t}), y)$$

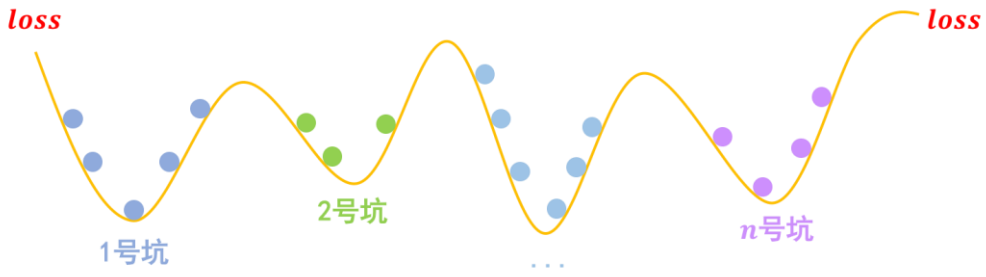
其中，$S = r \in \boldsymbol{R}^{d}, \|r\|_2 \le \varepsilon$为扰动的约束空间，$\alpha$为小步的步长，然后这里是在对这个扰动作投影。

具体来说，其执行过程就是一次次地进行前后向传播，一次次地根据 grad 计算扰动 r，一次次地将新的扰动 r 累加到 embedding 层的 grad 上，若超出一个范围，则再映射回给定范围内。最终，将最后一步计算得到的 grad 累加到原始梯度上。即以累加过 t 步扰动的梯度对应的 grad 对原梯度进行更新。

4.2.4.2 梯度惩罚

梯度惩罚其实是对于对抗训练的衍生，本质上是要提高模型的泛化能力。

我们可以从简单的几何图像中理解：即对于一个常规的分类问题，假设有 n 个类别，那么模型相当于有 n 个坑，然后让同类的样本放到同一个坑中，如下图所示：



梯度惩罚所要做的是：希望同类样本不仅要在同一个坑中，还要放在坑底；坑底意味着极小值点（梯度为零），这样能保证模型更加稳定、鲁棒。

4.2.4.3 虚拟对抗

虚拟对抗（VAT）是在对抗训练的基础上提出的一个半监督学习正则化方法。它是一种对于给定条件标签分布$p(y|x)$的数据度量该分布局部光滑性的一种方法。其实就是对于每一个数据点，它的条件标签分布对于局部的扰动鲁棒性怎么样，是否数据的一点小变化，就会导致预测的其标签的大变化。

它的优点也是它与对抗训练的不同之处在于：虚拟对抗在对抗训练时，不需要标签信息，所以可以应用于无监督学习。下图是论文作者对于 VAT 工作原理的描述：

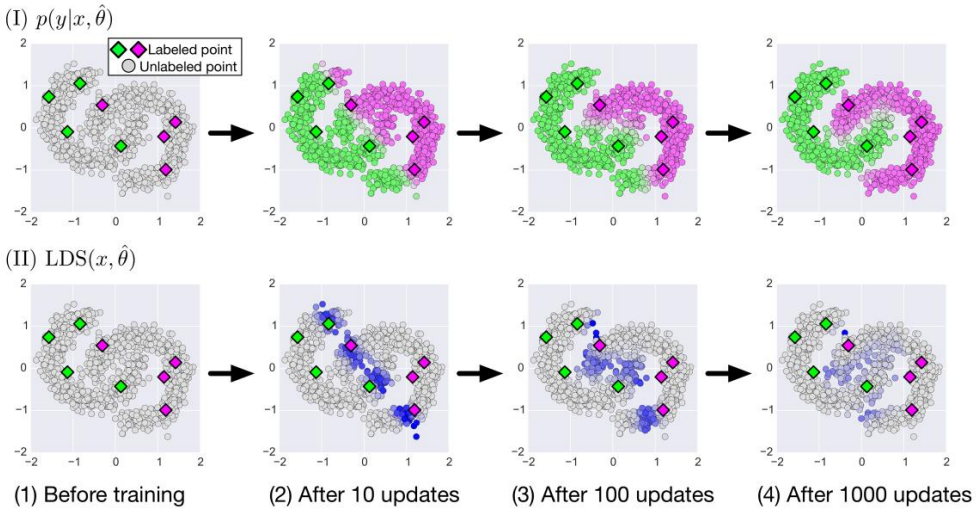


Fig. 1: Demonstration of how our VAT works on semi-supervised learning. We generated 8 labeled data points ($y = 1$ and $y = 0$ are green and purple, respectively), and 1,000 unlabeled data points in 2-D space. The panels in the first row (I) show the prediction $p(y = 1|x, \theta)$ on the unlabeled input points at different stages of the algorithm. We used a continuous colormap to designate the predicted values of $p(y = 1|x, \theta)$, with Green, gray, and purple respectively corresponding to the values 1.0, 0.5, and 0.0. The panels in the second row (II) are heat maps of the regularization term LDS$(x, \hat{\theta})$ on the input points. The values of LDS on blue-colored points are relatively high in comparison to the gray-colored points. We used KL divergence for the choice of D in Eq.(5). Note that, at the onset of training, all the data points have similar influence on the classifier. After 10 updates, the model boundary was still appearing *over* the inputs. As the training progressed, VAT pushed the boundary away from the labeled input data points.

4.2.5 UDA

现有 NLP 的数据增强主要有两条思路，一个是加噪，另一个是回译，均为有监督方法。加噪即为在原数据的基础上通过替换词、删除词等方式创造和原数据相类似的新数据。

Easy Data Augmentation for Text Classification Tasks （EDA）提出并验证了几种加噪的技巧，分别是同义词替换、随机插入、随机交换、随机删除。

虽然传统的数据增广方法有一定的效果，但对于渴求大量训练数据的深度学习模型，传统的方法效果始终有限。而 Unsupervised Data Augmentation（UDA）无监督数据扩增方法的提出，为大量数据缺失打开了一扇大门。

UDA（Unsupervised Data Augmentation 无监督数据增强）是 Google 在 2019 年提出的半监督学习算法。该算法超越了所有现有的半监督学习方法，并实现了仅使用极少量标记样本即可达到使用大量标记样本训练集的精度。在 UDA 论文中，效果体现在 IMDb 数据集上，通过仅仅 20 个标记样本与约 7 万余个无标记样本（经过数据增强）的 UDA 算法学习，最终达到了与有 2.5W 标记数据集更好的效果。

UDA 算法的大致框架见下图

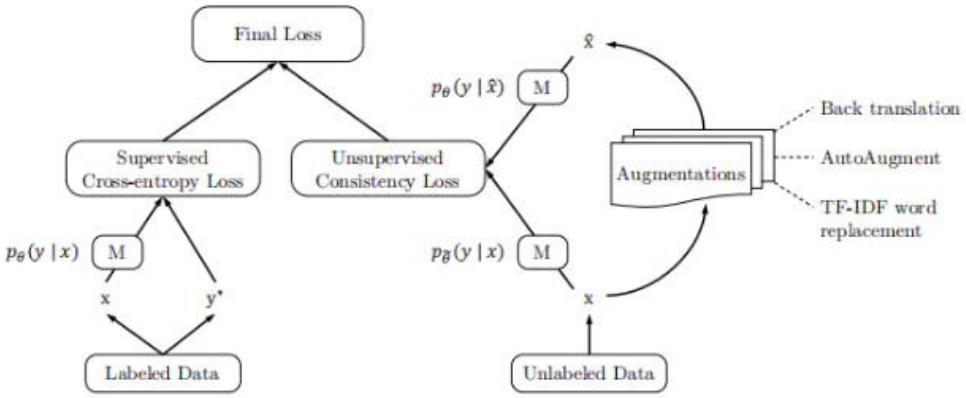


图 4.2.9

相比较于传统的加噪方法，例如：高斯噪声、dropout 噪声、或者简单的仿射变换，对不同任务进行针对性的数据增强能够生成更加有效的噪声。它具有以下优点：

（1）扰动的有效性：让增强数据和原始数据使用相同标签在监督学习中取得了良好性能，因此，对于无标签数据的增强也是类似有效的。

（2）扰动的多样性：由于可以对输入数据进行多种方式的改动而不改变数据标签，所以数据增强具有更强的多样性，而例如高斯噪声和 Bernoulli 噪声改变了局部信息，因此多样性不好。另外，由于是在一组增强数据集上进行平滑操作，所以数据增强拥有较高的效率。

（3）定向归纳偏差：不同的任务需要不同的归纳偏差。如自动增强，数据增强策略可以直接优化以提高验证性能 每项任务。这种面向性能的增强策略可以学会在原始标记集中找出缺少的或最想要的归纳偏差。虽然自动数据增强策略是应用于监督学习任务中的，但是在本文半监督数据增强中，同样有效。

另外，UDA 优秀的另一个重要的突破是采用了 Training Signal Annealing（TSA）方法在训练时逐步释放训练信号。

当收集了少量的标注的数据和大量未标记的数据时，可能会面临标记数据和未标记数据相差很大的情况。比如标记的数据都和保险相关，但未标记的数据都是热点新闻。因为需要采用大量的未标记数据进行训练，所需的模型会偏大，而大模型又会轻松的在有限的有监督数据上过拟合，这时 TSA 就要逐步的释放有监督数据的训练信号了。

作者对每个 training step 都设了一个小于等于 1 的阈值 ηt，当一个标签例子的正确类别 P 的概率高于阈值 ηt 时，模型从损失函数中删除这个例子，只训练这个 minibatch 下其他标记的例子。

4.2.6 集成学习

集成学习（Ensemble Learning）是通过构建并结合多个机器学习器来完成学习任务的机器学习方法，由于集成学习能高效地解决实际应用问题，所以在机器学习领域内备受关注。最初，集成学习旨在提高自动决策系统的准确性，而现如今此方法已经能够成功解决各种机器学习问题，

其核心思想在于先训练若干个个体学习器，再通过一定的结合策略，就可以最终形成一个强学习器，以达到博采众长的目的。

不同的集成学习算法之间主要的区别在于以下三个方面：提供给个体学习器的训练数据不同；产生个体学习器的过程不同；学习结果的组合方式不同。虽然集成学习领域不断有新的算法推出，但是这些算法大都是由一些经典算法如：Bagging、Boosting、Stacking 等改编得到的，这些经典算法具有良好的效果且被广泛应用于各个领域。

（1）Bagging 算法

Bagging 算法是最早的集成学习算法之一，它虽然结构简单，但是表现优越。该算法通过随机改变训练集的分布产生新的训练子集，然后分别用不同的训练子集来训练个体学习器，最后将其集成为整体。该算法中，由于使用自助采样法来产生新的训练子集，一些实例会被多次采样，而其他实例会被忽略，因此，对于特定的子空间，个体学习器会具有很高的分类精度，而对于那些被忽略的部分，个体学习器难以正确分类。但是最终的预测结果是由多个个体学习器投票产生的，所以当个体学习器效果越好且它们之间的差异越大时，该集成算法的效果就会越好。由于不稳定的学习算法对于训练集比较敏感，训练集只要产生一些微小的变化，就会导致其预测结果发生很大的改变，所以 Bagging 算法对于不稳定学习算法非常有效

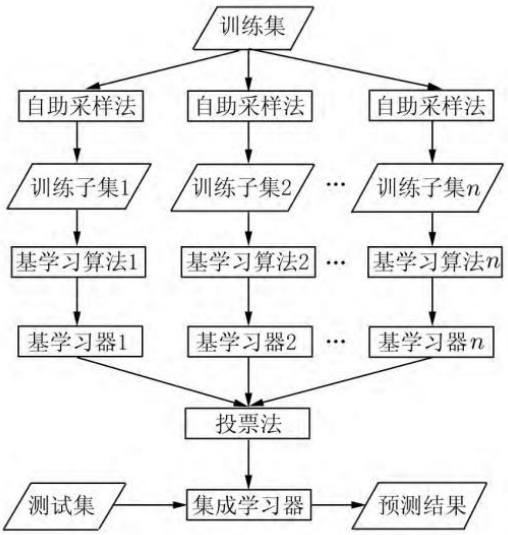


图 1 Bagging 算法流程图

Fig.1 Procedure of Bagging algorithm

（2）Boosting 算法

Boosting 算法是一种将弱学习器转换为强学习器的迭代方法，它通过增加迭代次数，产生一个表现接近完美的强学习器。其中，弱学习器是指分类效果只比随机猜测效果稍好的学习器，即分类准确率略高于 50%。在实际训练中，获得一个弱学习器比获得一个强学习器更加容易。因此，对于 Boosting 系列算法的研究意义非凡。Boosting 算法除了具有良好的实际性能外，还具有强大的理论基础和算法特点。

对于监督学习，该算法在第一个分类器之后产生的每一个分类器都是针对前一次未被正确分类的样本进行学习，因此该算法可以有效地降低模型的偏差，但随着训练的进行，整体模型在训练集上的准确率不断提高，导致方差变大，不过通过对特征的随机采样可以降低分类模型间的相关性，从而降低模型整体的方差. 当主分类器不能被信任，无法对给定对象进行分类时，例如，由于其结果中的置信度低，则将数据输送到辅助分类器，按顺序添加分类器。

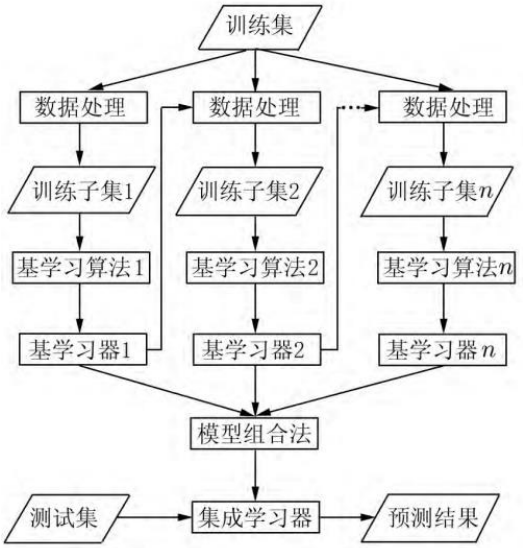


图 2 Boosting 算法流程图

Fig.2 Procedure of Boosting algorithm

（3）Stacking 算法

Stacking 也称 Stacked Generalization，它是指训练一个用于组合所有个体学习器的模型，即首先训练多个不同的个体学习器，然后再以这些个体学习器的输出作为输入来训练一个模型，从而得到一个最终的输出。 该算法的具体流程为：首先在整个训练数据集上通过重采样方法得到多个训练子集，然后使用这些新产生的训练集训练得到一系列分类模型，称之为 Tier1，然后将 Tier1 的输出组合后用于训练 Tier2 的元分类器. 除了重采样方法，交叉验证也经常在训练 Tier1 分类器时使用，即首先将训练集分成 N 等份，然后 Tier1 中的每个个体学习器根据前 N-1 份训练集进行训练，最后在第 N 份训练集上测试。

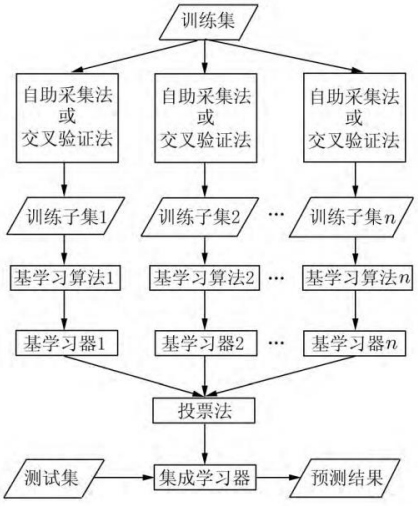


图 3 Stacking 算法流程图

Fig.3 Procedure of Stacking algorithm

（4）三种算法优缺点总结：

Bagging：与不稳定的学习算法相结合通常能产生一个强大的学习模型，并且具有良好的抗噪能力，而且各个基学习器可以并行生成，提高运行效率；

Boosting：能够显著提高弱学习器的学习效果，但很容易受到噪声的影响产生过拟合现象，并且每个基学习器只能顺序生成，训练效率相对较差；

Stacking：使用多个强大而不同的初级学习器并且使用类标概率代替预测类标作为次级学习器的属性会产生更好的结果，并且次级学习器选择简单的模型会降低过拟合的风险。

（5）基学习器的组合策略

对于个体学习器的组合策略主要分为两种：

首先是投票法，简单的投票法可以细分为绝对多数投票（majority voting）和相对多数投票（plurality voting）两种，区别在于是否考虑得票数是否超过个体学习器数量的一半。由于个体学习器之间的学习能力不同，投票过程中如果能将个体学习器的学习能力考虑进去，将可以进一步的提高整个系统的性能，因此产生了加权投票法。先估计出个体学习器的误差，然后令权重大小

与误差大小成反比，最终 T 个个体学习器加权投票的结果可以表示为:

$$H(x)=\sum_{i=1}^{T} w_i h_i(x)$$

其中 w_i 为第 i 个个体学习器的权值。

然后是平均法，这是一种组合连续数值型输出常用的方法，它主要分为简单平均法(simple averaging)和加权平均法(weighted averaging)。简单平均法可以表示为：

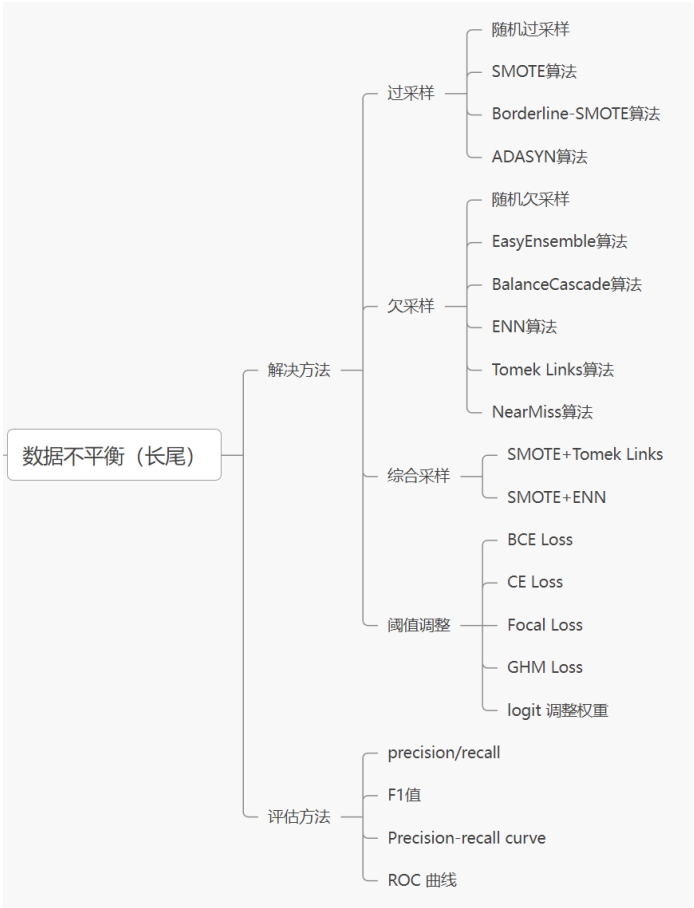
$$H(x)=\frac{1}{T}\sum_{i=1}^{T} h_i(x)$$

加权平均法结合了简单平均法和加权投票法，与加权投票法不同的是加权平均法中的权值不是用于类标，而是直接应用于实际的连续输出值．加权平均法的权值可以在集成系统生成期间作为训练的一部分获得，如 AdaBoost 算法中权值的产生，或者可以通过单独的训练来获得．加权平均法可以表示为：

$$H(x)=\sum_{i=1}^{T} w_i l_i(x)$$

4.3 数据不平衡

遇到数据不平衡问题有很多通用的解决方法，其中较为传统的是使用过采样和欠采样等方式尽可能平衡数据，从而降低数据的不平衡分布对模型所造成的影响。基本框架图如下所示：



4.3.1 过采样

过采样又称为上采样，其核心思想就是增加训练集中的少数类数据，使得正、反例数目接近，然后再进行学习。这样会使得训练的数据量增大，不仅能一定程度上解决小样本问题，而且还能尽可能使得数据类别分布平衡。以下将简单介绍一些过采样方法：

（1） 随机过采样算法

随机过采样是指从少数类的样本中进行随机采样来增加新的样本，即增加少数类样本数量。可以事先设置多数类与少数类最终的数量比例，在保留多数类样本不变的情况下，根据比例随机复制少数类样本，在使用的过程中为了保证所有的少数类样本信息都会被包含，可以先完全复制一份全量的少数类样本，再随机复制少数样本使得满足数量比例。

该算法的优点是：可以强行增加数据，增加训练样本；也能很好地平衡数据，确保模型对小众样本的识别率。

但是它的缺点也尤为明显。因为随机过采样是简单的对初始样本进行复制采样，这就使得学习器学得的规则过于具体化，不利于学习器的泛化性能，造成过拟合问题。

（2）SMOTE 算法

SMOTE 全称是 Synthetic Minority Oversampling 即合成少数类过采样技术。其主要思想是利用特征空间中少数类样本之间的相似性来建立人工数据，即对少数类样本进行分析并根据少数类样本人工合成新样本添加到数据集中。该算法的流程如下所示：

a) 对于少数类中的每一个样本x_i，以欧式距离为标准计算它到少数类样本集S_{min}中所有样本的距离，得到K近邻。

b) 根据样本不平衡比例设置一个采样比例以确定采样倍数 N，对于每个少数类样本x_i，从其K近邻中随机选择若干个样本，假设选择的近邻为$\tilde{x}$；

c) 对于每一个随机选出的近邻$\tilde{x}$，分别与原样本按照如下公式构建新的样本：

$$x_{new} = x + rand(0,1) \times (\tilde{x} - x)$$

该算法是对上面随机过采样方法的改进。由于不是简单地复制少数类样本，因此可以在一定程度上避免分类器的过度拟合，在实践上也可以证明该算法提高了模型的性能。

当然，该算法也有相应的缺陷，即由于对每个少数类样本都生成新样本，因此容易发生生成样本重叠的问题。而且在 SMOTE 算法中，出现了过度泛化的问题，这主要归结于产生合成样本的方法。特别是，SMOTE 算法对于每个原少数类样本产生相同数量的合成数据样本，而没有考虑其邻近样本的分布特点，这就使得类间发生重复的可能性增大。也就是说，合成的样本尽管很接近原样本但其实可能和原样本不属于同类，这将导致训练出的模型容易错误地区分位于边界上的样本数据。

（3）Borderline-SMOTE 算法

原始的 SMOTE 算法对所有的少数类样本都是一视同仁的，但实际建模过程中发现那些处于边界位置的样本更容易被错分，因此利用边界位置的样本信息产生新样本可以给模型带来更大的提升。Borderline-SMOTE 便是将原始 SMOTE 算法和边界信息算法结合的算法。算法流程如下：

a) 首先，对于每个$x_i \in S_{min}$确定一系列 K 近邻样本集，称该数据集为S_{i-kNN}，称$S_{i-kNN} \subset S$；

b) 其次，对每个样本x_i，判断出最近邻样本集中属于多数类样本的个数，即：$|S_{i-kNN} \cap S|$；

c) 最后，选择满足下面不等式的$\frac{k}{2} \leq |S_{i-kNN} \cap S| \leq k$，将其加入易被错分开的样本集合（这里也成为危险集）

上面的式子表明，只有最近邻样本集中多数类多于少数类的样本x_i才会被选中形成“危险集”。

该算法的好处是可以加强边界处模糊样本的存在感，使得新增样本更加靠近真实值，从而避免出现由于 SMOTE 算法过度泛化的问题。

（4）ADASYN 算法

ADASYN 算法的全称是 Adaptive Synthetic Sampling Approach for Imbalanced Learning，即被称为自适应合成。它能根据不同的少数样本其学习的困难程度，使用权重分布，为少数较难学习的类别样本合成更多的训练数据。它采用了两种改进数据分布的学习方式：首先是减少类不平衡所带来的偏差；其次是它能自适应地将分类决策边界向困难的实例移动。该算法的流程如下所示：

a) 假设有m个训练样本$\{x_i, y_i\}, i = 1, \ldots, m$，其中$x_i$是$n$维特征空间$X$中的一个实例，$y_i \in y = \{-1,1\}$与$x_i$相关的类别标签，将$m_s$和$m_l$分别定义为少数样本的数量和多数类样本的数量。

b) 可以计算出不平衡度$d = m_s/m_l, d \in (0,1]$；

c) 如果 d 小于类不平衡比率的预设阈值，计算少数类需要合成的样本数量：

$$G = (m_l - m_s) \times \beta$$

其中，$\beta \in [0,1]$是用于在生成合成样本后制定所需的平衡级别的参数。$\beta = 1$表示操作之后形成一个完全平衡的数据集。

d) 对于少数类中的每个样本x_i，根据 n 维空间中的欧几里得距离找到 k 近邻，并计算r_i：

$$r_i = \triangle i/K, \qquad i = 1, \ldots, m_s$$

其中，$\triangle i$为x_i的 k 近邻中多数类的占比，$r_i \in [0,1]$。

e) 之后对于r_i进行标准化处理，可得每个样本x_i在其中的占比：

$$\hat{r}_i = \frac{\hat{r}_i}{\sum_{i=1}^{m_s} r_i}$$

f) 然后计算少数类中每个样本x_i需要生成的合成样本个数：$g_i = \hat{r}_i \times G$。

g) 最后根据 SMOTE 算法生成样本生成合成样本：$s_i = x_i + (x_{zi} - x_i) \times \lambda$，其中$(x_{zi} - x_i)$是 n 维空间的差分向量，$\lambda \in [0,1]$是一个随机数。

4.3.2 欠采样

欠采样指的是去除掉一些多数类中的样本使得正例和反例的数目接近，然后再进行学习。这种采样方式显然可以使得样本均衡，但其造成的问题是将训练集变小。在本次的小样本学习任务中，该种采样方式可能造成训练数据量进一步减少，从而导致模型的准确性无法提升。我们在实践种，没有使用到这类模型，这里仅作介绍：

（1）随机欠采样

随机欠采样的思路和随即过采样是类似的，都是从多数类中随机选择一些样本组成样本集 E，然后将样本集 E 从训练集中删除，得到新的数据集，再让模型在该训练集上进行训练。

其优点是可以均衡样本的分布；但是其缺点更加明显，就是信息缺失，即由于采样的样本集合要少于原来的样本集合，导致将多数类样本删除有可能会导致分类器丢失有关多数类的重要信息，导致模型对多数类样本的准确性不佳。当然，我们可以在其基础上加上一些集成（ensemble）的方式来改进该模型。

（2）EasyEnsemble 算法

EasyEnsemble 算法是对于随机欠采样的改进，它的核心思想是多次随机欠采样，从而尽可能全面地覆盖所有信息。该算法的流程如下：

a) 从多数类中有放回的随机采样 n 次，每次选取与少数类数目相近的样本个数，那么可以得到 n 个样本集合，记作$\{S_{1maj}, S_{2maj}, \ldots, S_{nmaj}\}$。

b) 然后将每一个多数类样本的子集与少数类样本合并并训练出一个模型，则可以得到 n 个模型。

c) 最后将这些模型组合形成一个集成学习系统，最终的模型结果是这 n 个模型的投票值。

该算法的特点是利用到集成学习的方法，使用 boosting 算法来减小片偏差，使用 bagging 算法来减小方差（集成分类器），实际应用过程中还可以尝试选用不同的分类器来提高分类的效果。

（3）BalanceCascade 算法

BalanceCascade 算法的前半部分与 EasyEnsemble 算法相同，都需要对大众样本进行下采样训练出一个分类器，对于那些分类正确的大众样本不放回，然后对剩余的大众样本重新下采样产生训练集，训练第二个分类器，以此类推。其核心思路是：

a) 在每一轮训练时都使用多数类与少数类数量相等的训练集，训练出一个 Adaboost 基分类器。

b) 然后使用该分类器对全体多数类进行预测，通过控制分类阈值来控制假正例率（FPR），将所有判断正确的类删除。

c) 最后，进入下一轮迭代，继续降低多数类数量。

该算法的详细流程如下所示：

算法 2.8 BalanceCascade

输入：一个包含少数类阳性样本集 P 和多数类隐形样本集 N 的训练集 D，T-从 N 中抽取的子集个数，s_i-训练 Adaboost 基分类器H_i时的循环次数。

输出：一个组合分类器H(x)

1. $f = \sqrt[(T-1)]{\left(\frac{|P|}{|N|}\right)}$
2. for i = 1 to T:
3. 从 N 中随机抽取一个子集N_i, $|N_i| = |P|$
4. 使用 P 和N_i训练一个 Adaboost 分类器H_i（H_i由s_i个基分类器h_{ij}及其权重$\alpha_{i,j}$构成，θ_i是H_i的调节参数）

$$H_i(x) = sgn(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \theta_i)$$

5. 调节θ_i令H_i的 FP 率为 f
6. 移除 N 中所有被H_i正确分类的样本
7. 输出一个集成分类器

$$H(x) = \mathrm{sgn}(\sum_{i=1}^{T} \sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \sum_{i=1}^{T} \theta_i)$$

可以看出，该算法是基于 Adaboost 的。在该算法中，每一次迭代过程都会造成数据集中大众的数量减少，而且每一步的集成分类器都是由平衡的数据集训练得到的。只有当所有的$H(i)$都预测为正例的时候，最终的分类器才会预测为正例。

该方法可以有效减少欠采样造成的大众样本信息损失的问题，且在每次迭代过程中，在有限的样本空间里可以充分利用尽可能多的信息，但同样未考虑小众样本极度欠缺的情况。

（4）ENN 方法

对于属于大众样本集的一个样本，如果其 K 个邻近点有超过一半都不属于大众样本，则这个样本会被剔除。这个方法的另一个变种是所有的 K 个邻近点都不属于大众样本，这样的样本会被剔除。

这个方法的好处是能有效去除一些不重要的样本信息，可以减少不同类别之间的样本重叠，但能剔除的大众样本比较有限，故最好作为数据清洗的方法，结合其他方法使用。

（5）Tomek Links

Tomek Links 指的是，数据集中的两个样本彼此是对方的最近邻，同时他们的类别不同。这时候我们可以删除两个点中，属于样本较多的那一类的那个点，这样能一定程度上减轻两类数据的不平衡。

Tomek Links 的想法其实是，如果两个样本是 Tomek Links 的，那么分类器处理这两个样本的时候一定不太容易，那干脆删除一个减轻压力。不过这个方法比较危险，毕竟这样很可能导致信息缺失。这项方法也常常被用于数据清洗，结合其他方法使用。

（6）NearMiss 算法

NearMiss 本质上是从大众样本中选取最具代表性的样本用于训练，主要是为了缓解随机欠采样中的信息丢失问题，总结起来有以下几类：

NearMiss-1：在大众样本中选择与最近的 K 个小众样本的平均距离最小的样本。

NearMiss-2：在大众样本中选择与最远的 K 个小众样本的平均距离最小的样本。

NearMiss-3：对于每个小众样本，选择离它最近的 K 个大众样本，目的是保证每个小众样本都被大众样本包围。

优缺点：NearMiss-1 考虑的是与最近的 K 个小众样本的平均距离，是局部的，易受离群点的影响；NearMiss-2 考虑的是与最远的 K 个小众样本的平均距离，是全局的。而 NearMiss-3 方法则会使得每一个小众样本附近都有足够多的大众样本，显然这会使得模型的精确度高、召回率低。有论文中证明了在某些数据下 NearMiss-2 方法的效果最好。但三种方法的计算量普遍都很大。

4.3.3 综合采样

上述的方法都是针对于一类样本进行重采样，即对多数类样本进行欠采样或者对少数类样本进行过采样。其实我们也可以综合上述的过采样和欠采样的方法，将两者结合起来，来解决样本类别分布不平衡和过拟合的问题。本部分会简单介绍两种综合采样的方式。

（1）SMOTE+Tomek Links

SMOTE+Tomek Links 方法的算法流程非常简单：首先，利用 SMOTE 方法生成新的少数类样本，得到扩充后的数据集 T。然后，剔除 T 中的 Tomek Links 对，得到最终的训练数据。

普通的 SMOTE 方法生成的少数类样本是通过线性插值得到的，在平衡类别分布的同时也扩张了少数类的样本空间，产生的问题是可能原本属于多数类样本的空间被少数类“入侵”，引起数据泛化的问题，容易造成模型的过拟合。

而 Tomek Links 对寻找的是那种噪声点或者边界点，可以很好地解决“入侵”的问题，使得多数类样本的空间不会被少数类样本“入侵”，明确了模型的边界。因此，使用这种方式可以很好地平衡过采样和欠采样的关系，发挥出其训练优势。

（2）SMOTE+ENN

SMOTE+ENN 算法的思想和 SMOTE+Tomek Links 算法是类似的。它的过程是：首先，使用 SMOTE 算法生成新的少数类样本，得到扩充后的数据集 T；然后，对数据集 T 中的每一个样本使用 KNN 方法预测，如果预测结果与实际类别标签不符，则剔除该样本。

这种方式也可以使得模型避免因 SMOTE 而带来的数据泛化的问题，引导训练模型明确不同类别样本的边界，从而提升模型的准确率。

4.3.4 阈值调整

阈值调整的思想就是不去改变训练样本的数据集，而是从样本数据的距离和权重出发，在样本空间中强化少数类的样本而弱化多数类的样本，使得模型对少数类样本也能有一定的判别能力，从而达到数据均衡的目的。而从实践上来看，“阈值调整”的核心就是去改变其 Loss 函数。

首先，我们先介绍一下 BCE Loss：

在自然语言处理领域，BCE Loss 在多标签文本分类中是最常用的。如果对于给定的一个数据集$\{(x^1,y^1),(x^2,y^2),\dots,(x^N,y^N)\}$，其中有 N 个样本，其中每个样本都有多个确切的标签$y^k=\{y_1^k,\dots,y_C^k\}\in\{0,1\}^C$（C 表示一共有 C 个类别标签），它的分类器输出是$z^k=\{z_1^k,\dots,z_C^k\}\in R$。那么 BCE Loss 如下所示：

$$L_{BCE}=\begin{cases}-log(p_i^k), & if\ y_i^k=1\\ -log(1-p_i^k), & otherwise\end{cases}$$

其中，可以使用 sigmoid 函数$p_i^k=\sigma(z_i^k)$来计算p_i^k。我们可以看到，由于该 Loss 函数使得多数类样本占主导，导致了单纯的 BCE 其实很容易被数据不平衡所影响。

以下将介绍一些我们实践过或者搜寻到的可用的 Loss Function。

（2）FL（Focal Loss）

通过在 BCE 的基础上乘上一个调节因子（其中可调聚焦参数$\gamma\geq 0$），Focal Loss 将较高的 loss 权值置于那些“难以分类”的实例样本上。这样就能使得这些少数据的样本权重增加，进而提升模型对该种样本的预测能力。因此 Focal Loss 函数可以表示为：

$$L_{FL}=\begin{cases}-\alpha(1-p)^\gamma log(p), & if\ y=1\\ -(1-\alpha)p^\gamma log(1-p), & if\ y=0\end{cases}$$

我们会发现 Focal Loss 不仅实现上较为简单，而且还往往更加灵活。它可以从基于类别分布的反向加权，到不需要知道类别，直接根据分类的可信度进行的困难样本挖掘。

（3）GHM Loss

GHM(gradient harmonizing mechanism) 是一种梯度调和机制。Focal Loss 虽然强调对 hard example 的学习，但不是所有的 hard example 都值得关注，有的 hard example 很可能是离群点，

过分的关注就可能导致错误。GHM 定义了梯度模长 g：

$$g = |p - p^*| = \begin{cases} 1-p, & if\ p^* = 1 \\ p, & if\ p^* = 0 \end{cases}$$

经过统计我们可以发现，梯度模长 g 接近于 0 的样本数量最多，随着梯度模长的增长，样本数量迅速减少，但是在梯度模长接近于 1 时，样本数量也比较多。

因此，GHM 的出发点是；既不要关注那些容易学习的样本，也不要关注那些离群点特别难分的样本。为此，作者定义了梯度密度$GD(g) = \frac{1}{l_\varepsilon(g)} \sum_{i=1}^{N} \delta_\varepsilon(g_k, g)$，其物理含义是：单位梯度模长 g 部分的样本个数。最终 GHM Loss 为：

$$L_{GHM} = \sum_{i=1}^{N} \frac{L_{CE}(p_i, p_i^k)}{GD(g_i)}$$

（4）logit 调整权重

这个方法的实质是将互信息的概念纳入到解决类别不平衡问题。我们可以从“互信息”的角度来考虑某个分类问题数据是否均衡。我们会发现，对于不均衡的数据样本，从整个训练集中统计各个类别的频率$p(y)$，频率$p(y)$一定会集中在某几个类别中。而消除数据不平衡就是将这个频率从模型中消除。Google 的一篇文章《Long-tail learning via logit adjustment》就通过这种方式达到了 SOTA 的效果。

4.3.5 评价方式

其实，对于数据不平衡问题，除了相应的解决方案，对于模型的评价方式同样重要。因为假如一个二分类问题的数据是不平衡的，它的正向样本占比 99%，反向样本占比 1%。那么对于同样数据分布的测试集，一个模型很可能全猜“正向”，则它的准确率就会达到 99%。尽管这样看起来它的准确性很高，但是实际上这种模型的稳定性很差，也没有实际应用的价值。因此，如何来评价模型也是一个值得讨论的问题，以下将简单介绍一下评价方式和定义。

精确率(Precision)：在所有系统判定的“真”的样本中，确实是真的的占比，就是 TP/(TP+FP)。

召回率（Recall）：在所有确实为真的样本中，被判为的“真”的占比，就是 TP/(TP+FN)。

TPR（True Positive Rate）：同 Recall 一样。

FPR（False Positive Rate）：又被称为“Probability of False Alarm”，就是所有确实为“假”的样本中，被误判真的样本，或者 FP/(FP+TN)。

F1 值：为了综合考量精确率和召回率而设计的一个指标，一般公式为取 P 和 R 的 harmonic mean:2*Precision*Recall/(Precision+Recall)。ROC=Receiver Operating Characteristic，是 TPR vs FPR 的曲线；与之对应的是 Precision-Recall Curve，展示的是 Precision vs Recall 的曲线。

我们在实践中都是使用 F1 值对我们的模型进行评价，衡量该模型的性能。

5 采用的技术路线及成果

因为原赛题结束时间较早，并且结束后无法提交，因此无法得知模型修改后的正确率。为了深入研究小样本问题和长尾问题，对模型进行有效评估，我们从网上选取了新的多类别分类数据集，并人工划分训练集和测试集，让训练集拥有 few-shot 和 long-tail 这两个特征，并且每一个类别的样本数都与比赛的类别样本数相近，复刻了赛题情景。评估模型的方法也与赛题相同，即使

用 Macro-F1 作为评估指标。

如下是我们使用的新数据集：

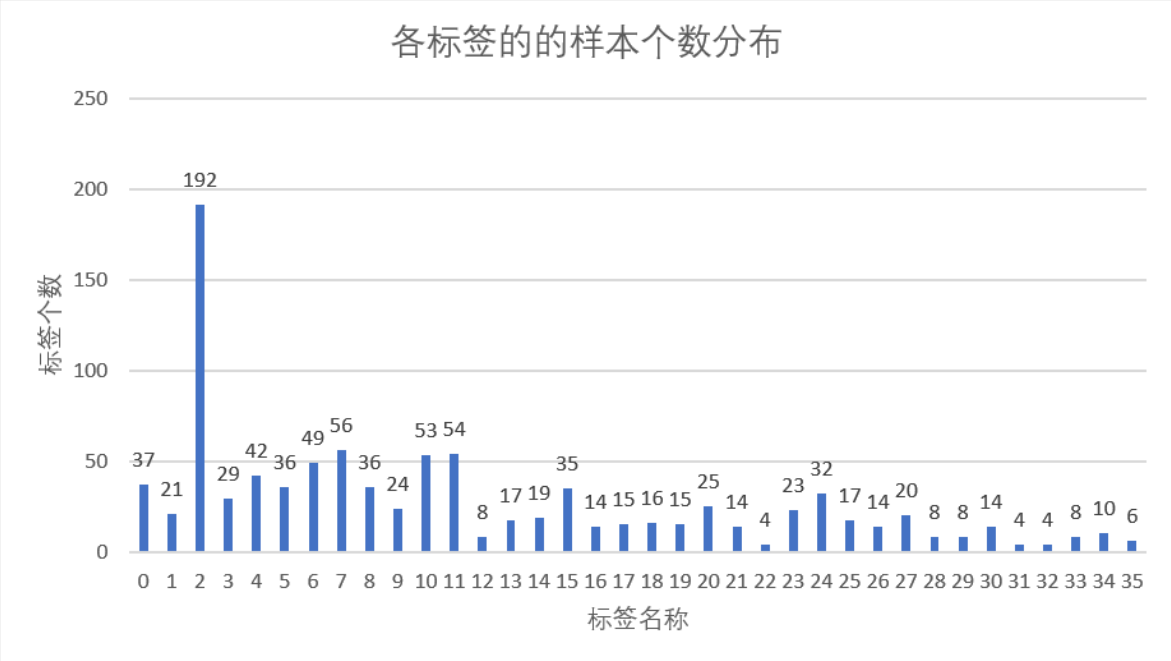
TNEWS'数据集下载

3.IFLYTEK' **长文本分类** Long Text classification

该数据集共有1.7万多条关于app应用描述的长文本标注数据，包含和日常生活相关的各类应用主题，共119个类别："打车":0,"地图导航":1,"免费WIFI":2,"租车":3,...,"女性":115,"经营":116,"收款":117,"其他":118(分别用0-118表示)。

```
数据量：训练集(12,133)，验证集(2,599)，测试集(2,600)
例子：
{"label": "110", "label_des": "社区超市", "sentence": "朴朴快送超市创立于2016年，专注于
每一条数据有三个属性，从前往后分别是 类别ID，类别名称，文本内容。
```

然后是新的标签样本分布，与原来的分布基本相同，可以很好地复现比赛训练集。



新的数据集相较于赛题数据集的准确率整体上升了 8%左右，评估效果较为良好，因此以下采用技术都是在新的数据集上生成的准确率，从而评估模型。

为了对比，首先说明模型 baseline 的 Macro-F1：63.99%

5.1 简单样本增强

（1）使用了最简单的数据增强方式，即将训练集复制一次然后添加如原训练集中，此方法的 Macro-F1：64.41%。

（2）然后我们选取了两种文本数据增强的常用方法：回译和同义词替换

回译：将训练集中的每条数据翻译成英文后再翻译回中文，将回译后的集合添加到原训练集中，此方法的 Macro-F1：65.74%。

同义词增强：在尝试同义词替换后，我们发现替换后的句子与与原句子差别很大，训练出来的效果也较差，原因是文本中有许多专业名词，然而我们使用的哈工大词库并不能很好地去识别并替换，因此我们放弃使用此方法。

可以见样本增强缓解了小样本问题的困境，但对于长尾问题无明显优化。

5.2 Mixup

正如上文所述，Mixup 方法是一种极好的数据增强方法，对于小样本任务有很大的帮助，因此我们应当在本题中使用该方法。

为了在 NLP 中使用 Mixup，首先应当将输入转化为 token。将输入转化为 token 的过程成为 tokenization，也叫 word segmentation，这是一种按照特定需求，将文本切分成一个字符串序列的操作。本质而言，这是一种将文本映射为数值的过程，是对于文本的数值化。

我们首先将训练集 mixup 一倍，也就是训练集数量为原来的两倍，此方法的 Macro-F1：64.92%，随后将训练集 mixup 两倍，也就是训练集数量为原来的三倍，此方法的 Macro-F1：65.57%，随后将训练集 mixup 三倍，也就是训练集数量为原来的四倍，此方法的 Macro-F1：63.56%，准确度较较低，说明 mixup 次数过多，样本分布已经与原数据集的分布产生巨大差异，训练效果不佳。

经过实验得知 Mixup 生成了有效的新样本，缓解了小样本问题的困境，但对于长尾问题无明显优化。

5.3 Rdrop

R-Drop 是一种更强的正则化 Dropout 手段。正则化的一些方法在深度学习中很常见，它很好地缓解了模型过拟合的问题，提高了模型的泛化能力。

此方法的 Macro-F1：64.07%

5.4 对抗训练

对于小样本的 NLP 任务，添加对抗样本能帮助我们补充样本的缺失，进而可能提高我们模型的泛化能力和鲁棒性。

由于 NLP 任务中所输入的文本多数是独热向量，其欧氏距离相差太大，不存在所谓小扰动。所以我们可以采用论文《Adversarial Training Methods for Semi-Supervised Text Classification》中的方法，将扰动加入到 Embedding 层。

使用了 FGM 和 PGD 两种对抗训练的方法：

FGM：64.76%

PGD：64.22%

5.5 过采样

（1）随机过采样：因为随即采样方式稳定性差，因此未尝试。

（2）从原始中文语句出发过采样，即复制少样本类别，添加到数据集中，使少样本类别的样本数与多样本类别的样本数近似，使少样本类别得到生存空间，此方法的 Macro-F1：66.43%.

虽然方法较为暴力，但还是缓解了长尾问题的困境。

随后，我们更换添加方式，从复制原数据添加，更改为将回译后的数据添加，此方法的 Macro-F1：66.86%。

此方法被证明是有效的，但是由于没有生成新的样本，而只是相似数据的堆叠，导致过拟合，线下测试数据准确率过高，线下线上不匹配。

（2）SMOTE：使用 SMOTE 对少样本类别进行过采样，使少样本类别的样本数与多样本类别的样本数相同，通过 embedding 层生成新的数据，这与 mixup 方法相同，只不过 SMOTE 是特定增加少样本类别的样本数量，而 mixup 是随机添加，此方法的 Macro-F1：66.20%.

（3）Borderline-SMOTE：SMOTE 改进版，此方法的 Macro-F1：66.32%.

（4）SVM-SMOTE：SMOTE 改进版，此方法的 Macro-F1：65.85%.

（5）Adasyn：与 SMOTE 类似，此方法的 Macro-F1：65.06%.

（6）KMeans-SMOTE：SMOTE 改进版，因为少样本类别数实在太小，无法形成聚类，因此此方法无法使用。

总结：SMOTE 类似方法都是通过现有少类别样本的 embeddings 生成新的数据，因此不会产生过拟合的情况，但是由于样本数实在过小，因此生成的新样本比较局限，该类的预测效果还是存在欠缺。总体来说较好地缓解了长尾问题。

5.6 欠采样

欠采样优随机欠采样，EasyEnsemble，BalanceCascade，NearMiss 等方法，其思想是对多样本类别进行欠采样，去符合少样本类别，这样会进一步削减训练集的样本个数。但是因为该训练集不止有长尾特性，还有小样本的特性，因此欠采样只会适得其反，训练出来的效果也较差，此处便不多讨论。

5.7 综合采样

综合采样是结合过采样和欠采样的方法

（1）SMOTE+Tomek Links

SMOTE 对小样本类别进行过采样，Tomek Links 进行欠采样，起到数据清洁的作用，此方法的 Macro-F1：65.56%.

（2）SMOTE+Tomek Links

更改了数据清洁的方式，此方法的 Macro-F1：64.78%.

结论：综合采样是在过采样的基础上进行了数据清洁，即删除了部分数据，防止出现类与类边界模糊的情况，但原因与上述相同，此问题还有小样本属性，因此删除数据可能产生了负面影响，因此效果不如单纯的过采样。得出的结论是欠采样不适用于小样本数据集。

5.8 重加权

重加权(阈值调整)的思想就是不去改变训练样本的数据集，而是从样本数据的距离和权重出发，在样本空间中强化少数类的样本而弱化多数类的样本，使得模型对少数类样本也能有一定的判别能力，从而达到数据均衡的目的。而从实践上来看，“阈值调整”的核心就是去改变其 Loss

函数。

Baseline 采用的 CE，即综合交叉熵，下面我们采用新的 loss 计算方式

（1）BCE loss

此方法的 Macro-F1：62.47%.此方式适用于二类别分类，因此效果差是肯定的，此处只是进行对比。

（2）Focal loss

此方法更加注重少样本类别的权重计算，优化了权重计算方式，此方法的 Macro-F1：64.43%

（3）CB loss

在 Focal loss 基础上进一步优化，此方法的 Macro-F1：64.13%

（4）DB loss

在 CB loss 基础上进一步优化，此方法的 Macro-F1：63.37%

（5）GHM loss

此方法的 Macro-F1：63.47%

（6）Dice loss

此方法的 Macro-F1：63.73%

总结：通过实验，发现重加权的方式对于模型并没有产生可观的优化，甚至有些产生了负优化，并且越高级的计算方式效果更差，具体原因我在网上搜索了很多，但没有一个确切的答案，下面分享一下搜索的结果。

在这篇论文中提出，对于长尾数据集，当类别少、不平衡度较低的情况下，重加权确实有效，但是随着类别数增多、不平衡度加剧，重加权的效果并不理想。

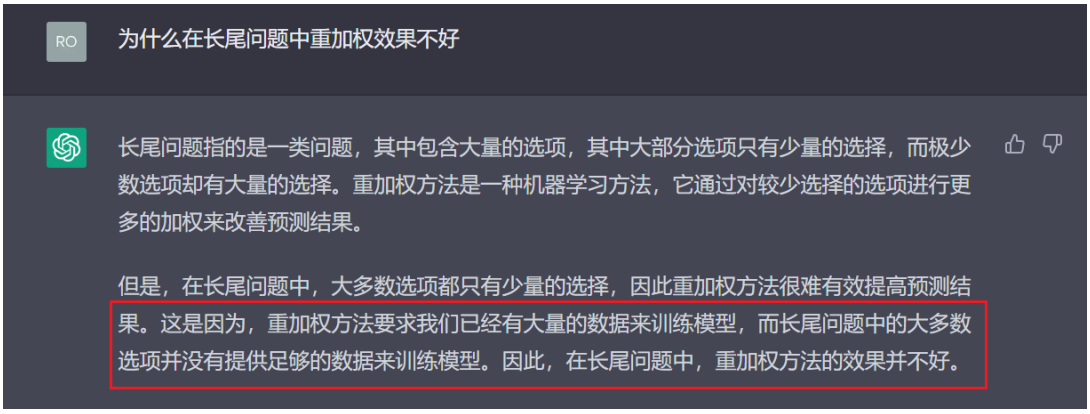
实验结果

我们在长尾CIFAR数据集上评估了重加权方法，见表3，我们发现重加权在CIFAR-10-LT上能得到更小测错误率，但是在CIFAR-100-LT上变差了。这表明直接用重加权并不是理想的选择，特别是类别数增加不平衡度增加的时候。

Table 3: Top-1 error rates of re-weighting methods. It shows directly applying re-weighting is inappropriate, especially when the number of classes increases.

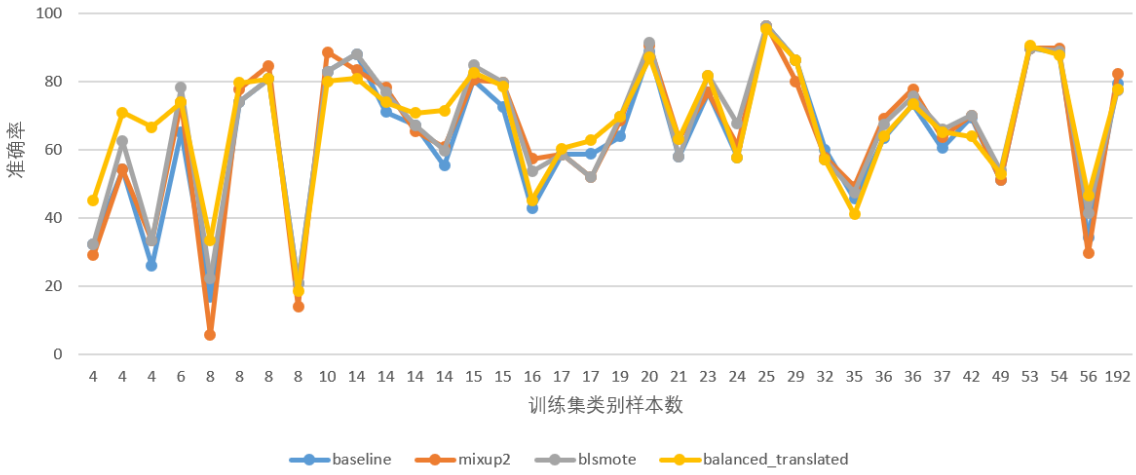
Datasets	CIFAR-10-LT		CIFAR-100-LT	
Imbalance factor	100	50	100	50
CE	30.18	24.78	**61.73**	57.90
CB_CE	28.26	22.76	66.40	63.48
CS_CE	29.07	23.74	70.92	63.78
Focal loss	29.62	24.75	61.90	57.56
CB_Focal	**27.02**	**22.03**	62.36	**57.24**

同时，我也问了 ChatGPT，重加权需要大量的数据去训练模型，而小样本问题限制了重加权的作用。



综上所述，重加权效果不佳与小样本特性和数据集自身原因（类别、不平衡度等）密切相关。

我们也输出了每个类别的准确率，可以看到少样本类别的准确率普遍较低，过采样和样本平均的方法可以有效缓解少类样本生存空间小的困境。



6 亮点价值

1、从重采样和重加权两个方面，对长尾问题进行了透彻深入的研究与代码实践

2、着重从文本增强和 embedding 增强，对于小样本训练进行了细致的研究与代码实践

7 总结与思考

1、理论与实践结果不对应，当样本数极小的情况下生成有效样本的局限性较大

2、tricks 的堆叠不一定能提升模型的准确率

3、在处理小样本训练问题时，主要涉及数据增强的手段，对于半监督学习、迁移学习、元学

装 订 线

习、PET 范式等方法并未进行代码实践

8 参考文献

[1] Sun, L., Xia, C., Yin, W., Liang, T., Yu, P.S., & He, L. (2020). Mixup-Transformer: Dynamic Data Augmentation for NLP Tasks. COLING.

[2] Guo, H., Mao, Y., & Zhang, R. (2019). Augmenting Data with Mixup for Sentence Classification: An Empirical Study. ArXiv, abs/1905.08941.

[3] https://zhuanlan.zhihu.com/p/370774948

[4] https://zhuanlan.zhihu.com/p/549605526

[5] https://blog.csdn.net/qq_35812205/article/details/119865130

[6] https://zhuanlan.zhihu.com/p/46016518

[7] https://zhuanlan.zhihu.com/p/164502624

[8] Goodfellow, I.J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. CoRR, abs/1412.6572.

[9] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards Deep Learning Models Resistant to Adversarial Attacks. ArXiv, abs/1706.06083.

[10] Miyato, T., Dai, A.M., & Goodfellow, I.J. (2017). Adversarial Training Methods for Semi-Supervised Text Classification. arXiv: Machine Learning.

[11] Miyato, T., Maeda, S., Koyama, M., & Ishii, S. (2019). Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41, 1979-1993.

[12] https://kexue.fm/archives/7234

[13] https://kexue.fm/archives/7466

[14]胡西范,陈世平.基于机器学习的小样本学习综述[J].智能计算机与应用,2021,11(07):191-195+201.

[15] 赵凯琳,靳小龙,王元卓.小样本学习研究综述[J].软件学报,2021,32(02):349-369.DOI:10.13328/j.cnki.jos.006138.

[16] 徐继伟,杨云.集成学习方法:研究综述[J].云南大学学报(自然科学版),2018,40(06):1082-1092.

[17] https://blog.csdn.net/qq_36330643/article/details/77621232

[18] https://github.com/flennerhag/mlens

[19] https://arxiv.org/pdf/1904.12848v2.pdf

[20] https://zhuanlan.zhihu.com/p/549605526

[21] https://www.jianshu.com/p/5d4e18b8de04

[22] https://zhuanlan.zhihu.com/p/186211797

[23] https://blog.csdn.net/xixiaoyaoww/article/details/104688002/

[24] https://github.com/Tongjilibo/bert4torch/tree/master/examples/training_trick

装 订 线

[25] https://zhuanlan.zhihu.com/p/387747120
[26] Liang, X., Wu, L., Li, J., Wang, Y., Meng, Q., Qin, T., Chen, W., Zhang, M., & Liu, T. (2021). R-Drop: Regularized Dropout for Neural Networks. ArXiv, abs/2106.14448.
[27] https://zhuanlan.zhihu.com/p/434779952
[28] https://github.com/fushengwuyu/R-Drop
[29] https://zhuanlan.zhihu.com/p/146777068
[30] https://blog.csdn.net/anshuai_aw1/article/details/89177406
[31] https://zhuanlan.zhihu.com/p/183852900
[32] Kang B, Xie S, Rohrbach M, et al. Decoupling representation and classifier for long-tailed recognition[J]. arXiv preprint arXiv:1910.09217, 2019.
[33] https://zhuanlan.zhihu.com/p/32940093
[34] https://zhuanlan.zhihu.com/p/274568639
[35] https://www.zhihu.com/question/59236897
[36] https://blog.csdn.net/weixin_40487385/article/details/88966487
[37] Huang, Yi, et al. "Balancing Methods for Multi-label Text Classification with Long-Tailed Class Distribution." arXiv preprint arXiv:2109.04712 (2021).
[38] https://zhuanlan.zhihu.com/p/510629949
[39] Zhang, Hongyi, Moustapha Cissé, Yann Dauphin and David Lopez-Paz. “mixup: Beyond Empirical Risk Minimization.” ArXiv abs/1710.09412 (2017): n. pag.
[40] Menon A K, Jayasumana S, Rawat A S, et al. Long-tail learning via logit adjustment[J]. arXiv preprint arXiv:2007.07314, 2020.
[41] https://blog.csdn.net/qq_39917365/article/details/108255756
[42] https://github.com/facebookresearch/mixup-cifar10
[43] 陈志,郭武. 不平衡训练数据下的基于深度学习的文本分类[J]. 小型微型计算机系统, 2020, 41(1): 1-5. CHEN Zhi,GUO Wu. Text Classification Based on Depth Learning on Unbalanced Data. Journal of Chinese Computer Systems, 2020, 41(1): 1-5.
[44] Kowsari K, Jafari Meimandi K, Heidarysafa M, et al. Text classification algorithms: A survey[J]. Information, 2019, 10(4): 150.
[45] Yang, Yuzhe and Zhi Xu. “Rethinking the Value of Labels for Improving Class-Imbalanced Learning.” ArXiv abs/2006.07529 (2020): n. pag.
[46] Hu, Han and Pengyuan Liu. “小样本关系分类研究综述(Few-Shot Relation Classification: A Survey).” (2020).
[47] Bayer, Markus et al. “A Survey on Data Augmentation for Text Classification.” ACM Computing Surveys (2021): n. pag.