文本分类多视角：Zest 数据压缩、X-Class 极弱监督及量化神经网络鲁棒性

Jun-Qi Duan[1] Hao-Ran Xin[1] Hong-Zheng Wu[1*]
[1] Nankai University

摘要

本研究探索并实现了三种创新的文本分类方法：基于 Zstandard 数据压缩算法的 Zest 分类，通过压缩率差异实现类别判别；极弱监督下的 X-Class 方法，利用类别名称和文档语料构建类别表示，并结合注意力机制优化文档分类；以及卷积神经网络的后训练量化（PTQ）技术，研究量化噪声对模型置信度和校准的影响。实验结果表明，这些方法及其改进在多个标准数据集上展现了优秀的分类性能，为文本分类任务提供了新的思路和技术支持。

1. 引言：Introduction

文本分类是自然语言处理中的一项核心任务，旨在根据文本内容将其自动归类到预定义的类别中。我们探索了三种不同的文本分类方法，分别为基于数据压缩的 Zest 分类、极弱监督的 X-Class 方法以及卷积神经网络的后训练量化（PTQ）技术。通过对这些方法的复现与创新，我们验证了它们在文本分类任务中的有效性，并尝试为未来的研究提供新的思路和方向。具体实现的 Github 仓库链接为 https://github.com/YukiSiH/nlp2024.git

2. 相关工作：Related work

基于传统方法的文本分类研究. 早期文本分类方法主要基于统计特征与机器学习方法，如支持向量机（SVM）、朴素贝叶斯（Naïve Bayes）和决策树等。这些方法依赖文本特征的手动提取和统计分析，通常采用 TF-IDF 等特征表示方式，但在处理高维数据和复杂语义关系时效果较为有限 [2]。

基于深度学习的文本分类研究. 近年来，深度学习方法在文本分类中表现优异。尤其是卷积神经网络（CNN）和循环神经网络（RNN）被广泛用于捕获文本中的局部特征和序列特征。Kim 在 2014 年提出的基于 CNN 的文本分类模型利用卷积层提取局部词组特征，显著提升了分类效果 [4]。随后，基于 RNN 的 LSTM 模型也被登上了时代舞台，因为 LSTM 可以能够捕获长距离依赖关系，所以在文本分类中表现突出。

预训练语言模型的兴起. 近年来，预训练语言模型（如 BERT、GPT 等）为文本分类带来了革命性进展。BERT 模型 [1]通过双向编码器捕获上下文信息，使得分类性能大幅提升，尤其是在情感分析、垃圾邮件检测等细分类任务中效果显著。类似地，RoBERTa [7]、ALBERT [5]等模型通过改进预训练策略，在文本分类中进一步优化了性能。

多任务学习和自监督学习. 为了提升模型的泛化能力，近年来的研究中还将多任务学习和自监督学习引入文本分类。Liu 等人提出了一种新的零样本文本分类方法，称为自监督调优（Self-Supervised Tuning）[6]。与传统零样本分类方法不同，这种方法避免了依赖大量的标注数据或特定的提示模板，而是通过无监督数据对模型进行自监督训练，从而让模型在没有特定标签数据的情况下完成分类任务，在新闻分类和情感分析等任务中取得了良好效果。

*段钧淇，学号 2110697，邮箱 3102275944@qq.com; 辛浩然，学号 2112514，邮箱 2112514@mail.nankai.edu.cn；伍泓铮，学号 2111885，邮箱 whztdas@foxmail.com

3. **问题定义：** Problem definition

文本分类是自然语言处理中的一项基础任务，旨在根据文本的内容将其归类到一个或多个预定义的类别中。该任务的目标是自动分析文本，理解其语义或主题，并将其与对应的类别标签匹配。文本分类广泛应用于情感分析、垃圾邮件过滤、主题分类、法律文件分类等场景 [8]。文本分类任务可以分为二分类任务、多分类任务、多标签分类任务、层次分类任务等类型。文本分类的任务核心在于自动理解和归纳文本的含义，并根据内容将文本归类到适当的类别中。这个过程依赖于特征提取、模型训练以及预测分析等步骤。随着机器学习和深度学习的快速发展，文本分类任务取得了显著的进展，成为许多 NLP 应用中的关键技术。

4. **方法：** Methodology

4.1. **Zest** 的实现

段钧淇复现了论文《Text Ranking and Classification using Data Compression》 [3]中提出的文本分类方法。

Zest **分类方法**. Zest 分类是利用 Zstandard 压缩方法实现的。而 Zstandard 压缩之所以能进行分类，是因为不同类别的文本具有不同的语言特征和模式，而 Zstandard 压缩算法通过结合 LZ77 和熵编码的方法构建压缩字典，可以捕捉这些特征。Zstandard 中，压缩字典记录了文本中常见的词、词组和模式。这些模式使压缩器在遇到相似的文本时，能够用更短的编码表示重复或常见的部分，从而实现更高效的压缩。不同类别的文本往往有不同的高频词和短语，导致它们的压缩特征不同。如果文本属于一个类别但用另一个类别的字典进行压缩，压缩率会较低。

因而，Zest 分类方法的核心思想实际上就是基于“压缩率”进行文本分类。该方法假设文本的压缩率能够反映其在特定类别中的相似度。对于一个未知类别的文本样本，若该样本在某个类别的压缩率较低，则认为它更可能属于该类别。这其实是基于了信息论的思想，认为“压缩效果越好的模型，越能代表待分类文本的特征”。其背后假设是，如果某个文本确实属于某一类别，那么训练好的压缩模型应能高度概括它的特征，使压缩效率最高。这种方法实际上是通过计算“类别模型”对“待分类文本”的“压缩率”来判断该文本与各个类别的相似性，类似于一个非传统的“匹配度”或“相似度”计算。

Zest 分类器将数据压缩视作一种非显性特征提取过程。通过压缩率这一统计指标，模型能够提取出文本数据中隐含的特征，这些特征能显著反映不同类别之间的差异。相比传统的机器学习模型，该方法无需显式的特征工程，具有简洁性和普适性。

其中，压缩率 R 可以定义为压缩后文本长度与原始文本长度的比值：

$$R = \frac{\text{Compressed Length}}{\text{Original Length}} \tag{1}$$

该比值表征了压缩模型对文本的适应度，反映出文本在类别模型下的匹配程度。

而类别匹配度是对于待分类文本 T，其在不同类别的匹配度通过压缩率差异来衡量。压缩率较低的类别模型能更好地“解释”文本的结构和内容，从而体现出更强的类别适应性。

Zest **复现概述**. 按照论文描述的那样，Zest 压缩需要利用 Zstd 的字典训练功能，分别对不同样本构建了压缩字典，并通过文本压缩率的差异来进行分类判别。为了实现该方法，主要步骤包括：构建基于 Zstd 的语言模型、在训练集上进行字典训练、对新文本进行压缩、计算压缩率并基于压缩率差异进行类别判断。主要复现分为两个阶段：

1. 训练阶段（为每个类别构建压缩模型）：使用 zstd（Zstandard）的压缩算法，Zest 方法会针对每个分类类别分别训练一个压缩模型。将同类别的所有文本数据输入到压缩模型中进行训练，使得压缩模型捕获该类别文本的特点（如词频、词组等信息）。最终，每个类别都有一个单独的 zstd 模型。这些模型在压缩其所属类别的文本时，会获得较高的压缩比（更小的文件大小），而在压缩其他类别的文本时压缩效果较差。

2. 预测阶段（用压缩率进行分类）：对待分类的文本，利用不同类别的压缩模型分别计算其压缩率（交叉熵）。交叉熵越小，说明该类别的压缩模型对该文本的拟合效果越好。将文本分配到压缩效果最好的类别（即交叉熵最小的类别）。这样就能根据压缩模型对文本的适应程度，将文本分类到对应的类别。

Zest **复现的难点以及细节**. 在实现 Zest 的过程中，我们遇到了以下的难点：

1. Zstd 字典训练：Zest 要求通过多层字典训练不同大小的字典以捕捉类别间的细微差异。然而这里的层数选取是个难题，最后我们发现层数设置为 7 效果最好，也就是说模型会训练最多 7 层字典，字典大小逐层减小，会更细致地提取类别间的压缩特征。
2. 哈希化处理：为减少词汇稀疏性并确保模型的泛化能力，代码在文本预处理阶段对词进行哈希化处理。复现中，我们需要定义一个函数将每个词转换为字节流，并使用 0 作为词之间的分隔符。这样，词汇信息可以转换为压缩友好的字节流格式，使得不同词语对压缩率影响更具可比性。
3. 压缩率归一化处理：通过忽略压缩头信息的大小，并在计算压缩率时减去这些固定信息，来消除文本长度带来的压缩率偏差。这使得不同长度的文本能够更公平地进行分类比较。
4. 句子分割与预处理：为了使模型能更有效地处理长文本，我们定义了函数对文本进行句子分割。分割规则基于正则表达式，按标点符号或换行符进行切分，然后使用另一个函数清理分割后的文本内容。

Zest **的创新——动态字典缩小**. 在之前的综述中，我设想的创新方法是动态字典更新，因为我看到了 Zest 中字典在模型训练中是静态的，因而想要将动态思想引入作为创新。但在实际操作中，完全实现动态字典更新困难重重，因而最后实现的实际上是动态字典缩小。具体实现细节如下：

在开始时，我们设置初始的字典大小 D_0，通常与主字典大小相等，即：

$$D_0 = \text{primary\_dict\_size}$$

在每一次迭代 j 中，我们执行以下操作：

- 根据当前字典大小 D_j 生成字典。
- 使用生成的字典构建测试压缩器，并计算所有样本的平均压缩大小，记作 test_compr_size。

此外，我们引入一个压缩效果阈值 compression_threshold。如果满足：test_compr_size $<$ compression_threshold $\times$ len(samples_bytes)，表示压缩效果未达到预期，我们就会按照一个预定义的比例因子 dict_size_factor 来动态缩小字典大小。字典大小的更新公式为：

$$D_j = D_{j-1} \times \text{dict\_size\_factor}$$

其中，dict_size_factor 是一个预定义的缩小因子（例如，0.6）。

最后，如果压缩大小满足：test_compr_size $\geq$ compression_threshold $\times$ len(samples_bytes)，则停止字典的缩小过程，避免字典过度缩小导致压缩效果显著降低。

我们的创新方法通过动态调整 D_j，确保字典在适合当前数据的大小上优化了压缩率，避免了无意义的过度缩小，使得模型可以根据不同的数据分布和模式在生成合适字典的同时，提升压缩效果。特别是在数据复杂性不一致的情况下，较小的字典可以更有效地捕捉高频或常见模式，而较大的字典则用于较广泛的模式。这种方法更加灵活，适合多样化数据集。

4.2. XClass 的实现

辛浩然复现了论文 X-Class: Text Classification with Extremely Weak Supervision [9]。

问题背景. 在极弱监督文本分类设置下，输入仅包含一组文档 $D_i, i \in 1, ..., n$ 和一列类名 $c_j, j \in 1, ..., k$。类名旨在提供关于目标分类的提示，例如可以基于主题、情感或地理位置等不同标准对同一组

文档进行分类。任务目标是构建一个分类器，能够根据类名将新的文档归入相应的类别。

方法概述. 首先，我们通过结合类别名称和文档语料，逐步构建类别的表示。在类别表示的估计过程中，我们利用类别名称及其相关关键词来丰富类别的语义表示。关键词的选择和加权通过一个迭代过程完成，在每一步中，我们根据类别表示与候选关键词的相似性来选择新的关键词，同时为之前选出的关键词赋予更高的权重，从而形成一个动态优化的类别表示模型。

接下来，我们基于每个单词的上下文化词表示和静态表示来估计文档标识。通过结合上下文信息，我们能够有效地消除多义性，同时静态表示帮助减少文档中可能出现的异常影响。注意力机制用于计算每个词对类别表示的重要性，分别衡量单词与单一类别或类别集合的相似性，从而得到每个单词的注意力权重。为了融合不同计算方法的注意力权重，我们使用排名融合技术，将各个计算方法的结果进行集成。

在对单词进行加权平均后，得到句子表示。为了使文档与类别更加对齐，我们使用高斯混合模型对文档表示进行聚类，生成伪标签，并通过主成分分析去除噪声。伪标签的类别距离反映了置信度，接着我们选择置信度最高的文档作为训练数据，并进一步用这些数据微调 BERT 分类器。

复现的难点以及细节. 此处重点解释文档和类别表示的获取方法。

一种直接丰富类表示的方法是通过选择与类别名称相似的一定数量的单词，并对这些单词的表示进行平均以得到类表示。然而，这种方法存在两个问题：（1）为所有类别设置相同数量的关键词可能会损害少数类；（2）简单平均可能会导致语义上与类别本身有所偏离。为了克服这两个问题，我们采用了迭代的方式来逐步找到每个类别的下一个关键词，并通过对所有找到的关键词进行加权平均来重新计算类别表示。基于找到的关键词集合，我们用所有关键词 K_l 的加权平均表示来定义类 l 的全面表示 x_l。考虑到越早找到的关键词应该具有更高的重要性，因此它们应该赋予更高的权重。我们为第 k 个关键词分配的权重基于一个递减的分布，设置为 $1/k$。因此，最终的类表示可以表示为：

$$x_l = \frac{\sum_{i=1}^{|K_l|} \frac{1}{i} \cdot s_{K_l,i}}{\sum_{i=1}^{|K_l|} \frac{1}{i}} \quad (2)$$

对于给定的类别，列表中的第一个关键词总是类名。在第 i 次迭代中，我们检索与当前类表示最相似的词，并基于前 $i+1$ 个关键词计算新的类表示。如果已经收集到足够的关键词（例如，$T = 100$），或者新的类表示无法使前 i 个关键词保持不变，则停止扩展。

在构建类别表示的过程中，我们遵循注意力机制，根据单词与类别表示的相似性为单词分配权重。对于第 i 个文档中的第 j 个词，有两种可能的表示：上下文化的词表示 $t_{i,j}$ 和该词的静态表示 s_w。上下文化表示通过结合上下文来消除多义词的歧义，而静态表示则考虑了文档中的潜在异常。因此，选择上下文化表示或静态表示作为注意力机制的词表示都是合理的。

给定类别表示 x_c，我们使用了两种不同的注意力机制来计算每个单词的重要性：

- one-to-one: $h_{i,j} = \max_c\{\cos(e, x_c)\}$。它捕捉了与一个类别的最大相似性。此方法有助于检测特别与某个类别相似的词，例如“NBA”对应于“体育”。

- one-to-all: $h_{i,j} = \cos(e, \text{avg}\{x_c\})$，即与所有类别平均表示的相似性。此方法按与关注的类别集合的总体关系来排名词。

将两种表示（e 的两种选择）与两种注意力机制结合，共有四种方式计算每个词的注意力权重。为了在无监督的条件下融合这些注意力权重，我们不直接使用相似度值，而是通过排名来进行融合。具体来说，首先根据每种注意力机制计算出的权重对单词进行降序排序，得到四个排名列表。然后，我们计算这些排名的几何平均数，作为每个单词的最终排名列表。对于排名为 r 的单词，我们赋予其 $1/r$ 的权重。通过这种方式，我们综合了不同注意力机

制和词表示的效果，从而为每个单词分配了一个加权后的排名。在此基础上，我们使用加权后的单词表示 $t_{i,j}$ 来获得文档表示 E_i：

$$E_i = \frac{\sum_j \frac{1}{r_j} t_{i,j}}{\sum_j \frac{1}{j}}$$

提出的创新方法. 在计算类别表示时，我们结合了基于相似度计算的权重，并通过 BERT 模型的注意力权重为句子中的每个单词分配一个重要性评分。然后，通过加权平均的方式计算句子表示，其中每个词的权重会根据其注意力得分进行调整。这种方法可以根据上下文的相关性动态地决定每个词对句子表示的贡献，相比于简单平均所有词的方式，它能够让与类别表示更为相似的词在最终表示中占据更大的权重。

在 PCA 降维和数据处理方面，我们做了一些改进。首先，固定的 PCA 组件数被替换为根据累计方差比例动态选择主成分数的方法，确保降维后能够解释 95% 的方差。通过这种方法，降维的主成分数量可以根据数据的实际特性自动调整，而不再依赖于固定的参数值，从而提升了模型的适应性和降维效果。同时，为了避免内存溢出并提高计算效率，我们采用了 IncrementalPCA 来替代传统 PCA。IncrementalPCA 支持增量式训练，可以逐批次地加载和处理数据，而不需要一次性加载所有数据到内存中。这使得处理大规模数据集变得更加可行，并有效避免了内存消耗过大导致程序崩溃的问题。

在选择训练数据时，我们动态计算每个类别的置信度阈值，并根据类别的距离分布调整该阈值。这样，每个类别的选择标准变得更加灵活，不再固定为某个比例，而是根据数据的实际分布自动调整。

4.3. PTQ 的复现

伍泓铮复现了论文《An Underexplored Dilemma between Confidence and Calibration in Quantized Neural Networks》中提出的卷积神经网络（CNN）校准与量化鲁棒性关联的实验。

实验背景. 后训练量化（Post-Training Quantization，PTQ）是一种有效的神经网络压缩方法，通过将模型权重和激活值转换为低比特表示（如 int8 或 int4）以减小存储和计算复杂度。然而，量化引入的噪声会导致模型精度下降，尤其在高精度任务中更为明显。

本文关注了量化噪声与神经网络模型置信度和校准程度的关系，提出了一种尚未充分探索的假设：置信度越高的模型在量化过程中越能保持鲁棒性，即在量化噪声下预测稳定性较强。该假设的验证对于优化 PTQ 策略，减少性能损失具有重要意义。

核心假设与实验方法. 本文提出的核心假设是：过度自信的模型对量化噪声更鲁棒。低置信度预测更容易受到量化噪声的影响，但由于这些预测通常准确率较低，因而改变后对整体准确率影响较小；高置信度预测则较难改变，但其准确率较高，因而对整体准确率的贡献更大。为了验证这一假设，实验通过以下步骤实现：

1. 训练阶段：使用 ResNet56 和 ResNet50 分别在 CIFAR-100 和 ImageNet 数据集上训练浮点精度模型，以观察其校准行为和置信度分布。
2. 量化阶段：应用后训练量化技术，将模型从浮点精度映射到更低的精度（int8），并通过逐步降低权重量化精度（从 8 位到 4 位）增加量化噪声。
3. 验证阶段：比较量化前后模型的准确率、错误率变化及置信度分布，分析模型校准与量化鲁棒性之间的关联。

实验结果通过置信度分布图（Reliability Curves）和预测变化分布图（Swapped Prediction Distribution）展示，以支持假设。

实验复现过程. 实验复现主要分为以下几步：

1. 数据准备：加载 CIFAR-100 和 ImageNet 数据集，使用 PyTorch 预训练权重。
2. 模型训练：使用标准 ResNet 模型训练，记录训练后的置信度分布及准确率。

3. 模型量化：实现后训练量化，通过 PyTorch 的伪量化模块，将激活量化到 int8，权重从 8 位逐步降低到 4 位。
4. 数据分析：绘制量化前后的可靠性曲线与预测变化分布图，统计量化噪声对模型准确率的影响。

复现难点与细节. 在复现过程中，主要难点包括：

1. 量化参数的选择：权重量化范围的确定对实验结果影响较大。为确保量化噪声分布的合理性，我们采用分通道对称量化方法。
2. 置信度分布的测量：实验需要精确记录不同置信度区间的准确率和预测变化情况。我们通过划分置信度区间并统计每区间内预测变化数量来实现。
3. 实验结果可视化：可靠性曲线需要同时展示校准行为和置信度分布，我们设计了多图合成的方法以便清晰展示结果。

5. **实验：** Experimental Results

5.1. **Zest** 的实验

实验数据集. 二分类的实验数据集为 clickbait_data.csv，是有关新闻标题是否为标题党的数据集，共 32000 行，有 2 列，每一行代表一篇文章的标题和是否为标题党的标记。

此外，我们还进行了多分类的实验，数据集为 news-article-categories.csv，该数据集共有 6877 行数据，有三列，每一行代表一篇文章的类别、标题和正文内容。数据集中有五行数据没有标明类别，因而删去后实际上有用的只有 6872 行。而数据集中种类列共有 14 种不同的值，分别为：'ARTS & CULTURE', 'BUSINESS', 'COMEDY', 'CRIME', 'EDUCATION','ENTERTAINMENT', 'ENVIRONMENT', 'MEDIA', 'POLITICS', 'RELIGION','SCIENCE', 'SPORTS', 'TECH', 'WOMEN'，也就是说，本次多分类实验实际上是一个 14 分类任务的实验。

实验设置. 实验测试中 Zest 分类器参数设置如下：

- ZSTD_LEVEL = 22：设置压缩级别为 22。
- dict_size_factor = 3/5：设置字典大小因子为 0.6，使字典的大小将与数据集大小的 60% 相关。
- max_num_dicts = 7：限制字典的最大数量为 7，防止生成过多的字典，保持计算资源的使用效率。
- min_dict_size = 150：字典的最小大小为 150 字节，确保字典的有效性。
- compression_threshold = 0.9：设置压缩效果阈值为 0.9，如果压缩效果未能达到该阈值，则不使用当前的字典进行训练。(这部分是创新部分才要设置的)

此外，由于 Zest 分类计算资源消耗并不大，我们也不需要设置 GPU 训练，只需要在 CPU 上训练即可(本次实验在 Kaggle 中完成)。

复现结果以及创新结果. 根据上面的实验设置，我们复现 Zest 文本二分类的效果很好，AUC 达到了近 0.996，而用时却很短，只需 4s 左右。图1展示了 P-R 曲线、，图2展示了引入动态字典缩小后的类别得分的概率密度图，图3展示了引入动态字典缩小后的类别得分的正负样本分布图。

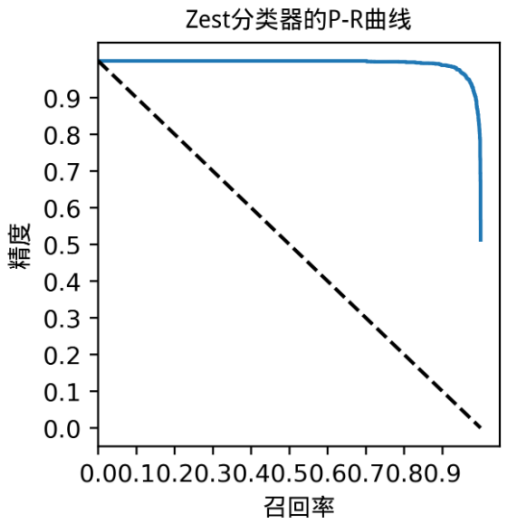

图 1. P-R 曲线

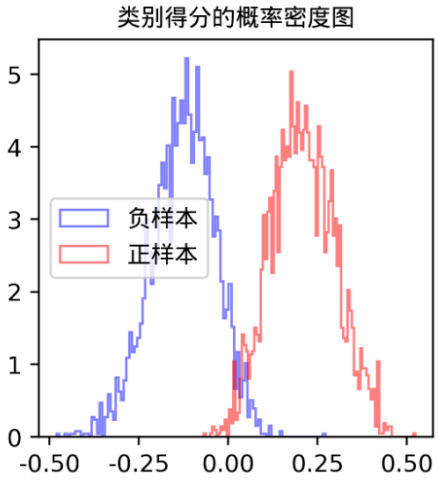

图 2. 类别得分的概率密度

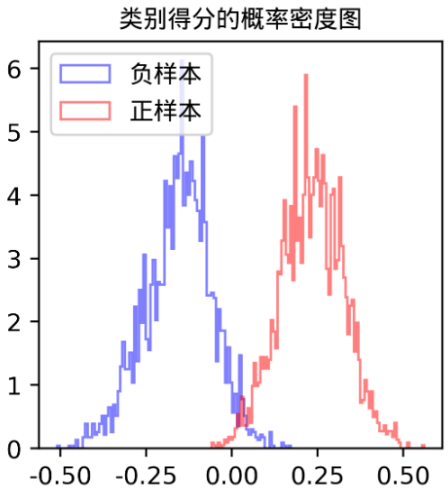

图 4. 类别得分的概率密度（创新）

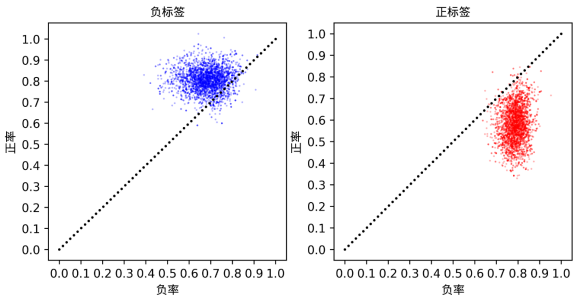

图 3. 类别得分的正负样本分布

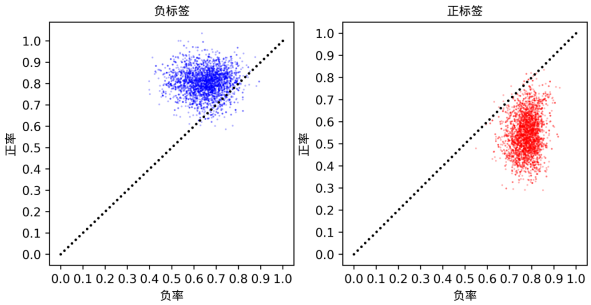

图 5. 类别得分的正负样本分布（创新）

此外，我们还将 Zest 和其他一些传统分类方法，如朴素贝叶斯（NB），逻辑回归（LR），马尔可夫链语言模型（MC）做了对比，结果如表1所示。

由于二分类任务效果区分度不明显，我们还进行了多分类的对比实验，同时还尝试结合了 Zest 和逻辑回归，结果如表2所示。

引入动态字典缩小后，虽然模型用时更长，但模型预测的更准确了。表3展示了利用 Zest 进行二分类的普通结果和创新结果对比，图4展示了引入动态字典缩小后的类别得分的概率密度图，图5展示了引入动态字典缩小后的类别得分的正负样本分布图。

结果分析. 根据实验结果，在二分类任务上，Zest 方法表现优异，获得了极高的 AUC 值，适合对分类精度要求较高的任务。而在多分类任务中，Zest 表现仍然不错，但在准确率和效率上与 LR 方法组合更为理想。创新改进后的 Zest 方法在各项指标上都有提升，特别是在 AUC 和样本得分分离度方面的增强，表明创新方案具有实际改进效果。然而，使用 Zest 后，训练和测试耗时更长了，因此，Zest 更适合精确分类任务，而 NB 和 LR 则适合对时间敏感的实时任务，同时 Zest 还可以和其他传统分类方法结合以提高准确率。

5.2. XClass 的实验

实验数据集. 本实验采用 AGNews 数据集作为实验数据集，该数据集包含 120,000 条无标签样本，待

方法	Zest	NB	LR	MC
AUC	0.99581589958159	0.9741	0.9622	0.9898
训练耗时	4.001900911331177	0.0082	2.7629	0.0128
测试耗时	3.609982490539551	0.0029	0.0016	0.0047

表 1. Zest 和其他分类方法结果对比（二分类）

方法	Zest	NB	LR	Zest+LR
准确率	0.7447272727272727	0.6901818181818182	0.7592727272727273	0.7614545454545455
训练耗时	42.99756574630737	0.09750938415527344	31.404354572296143	0.0128
测试耗时	210.9674162864685	0.014423608779907227	0.024036407470703125	209.62661385536194

表 2. Zest 和其他分类方法结果对比（多分类）

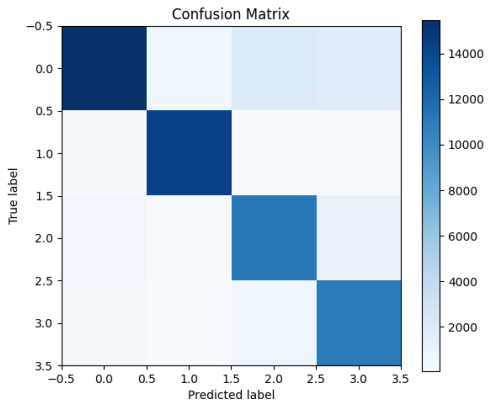

图 6. XClass 混淆矩阵

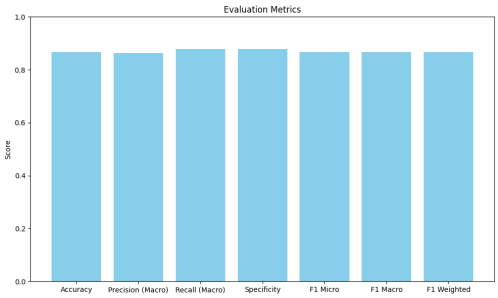

图 7. XClass 评估指标

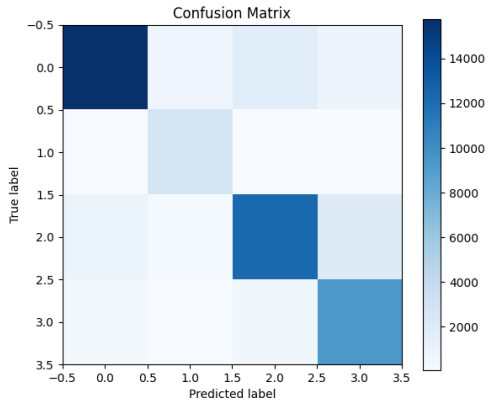

图 8. XClass 混淆矩阵（创新方法）

分类的目标类别分为四类，分别是：politics、sports、business 和 technology。

实验设置. 实验采用以下配置：Python 版本为 3.8.20，深度学习框架使用了支持 CUDA 12.4 的 PyTorch 2.4.0 版本。硬件环境为 NVIDIA A100 GPU，显存容量为 80GB。

复现结果以及创新结果. 图6展示了复现结果分类混淆矩阵，图7和表4展示了复现结果的指标评估。

图8展示了创新结果分类混淆矩阵，图9和表4展示了创新结果的指标评估。

结果分析. 在分类实验中，通过混淆矩阵和评估指标对原始方法和创新方法的性能进行了比较。原始方法的准确率达到了 0.867，精确率和召回率分别为 0.863 和 0.879，显示出较好的分类能力，尤其是在对各类样本的识别上表现出色，F1 分数也维持在 0.867 左右，表明其在平衡精确率和召回率方面的优势。相较之下，创新方法的准确率为 0.839，虽然召回率略高于原始方法（0.863），但其精确率为 0.812，导致 F1 分数降至 0.833，显示出在某些类别的识别

	普通结果	创新结果
训练耗时	4.001900911331177	4.176734209060669
正样本的平均得分	0.2068	0.2333
负样本的平均得分	-0.1245	-0.1548
测试耗时	3.609982490539551	3.8422253131866455
AUC	0.99581589958159	0.998326359832636

表 3. Zest 创新结果与普通复现结果对比

Metric	Original	Innovative
Accuracy	0.8674	0.8392
Precision (Macro)	0.8628	0.8123
Recall (Macro)	0.8789	0.8629
Specificity	0.8789	0.8629
F1 (Micro)	0.8674	0.8392
F1 (Macro)	0.8669	0.8326
F1 (Weighted)	0.8670	0.8399

表 4. XClass 评估结果

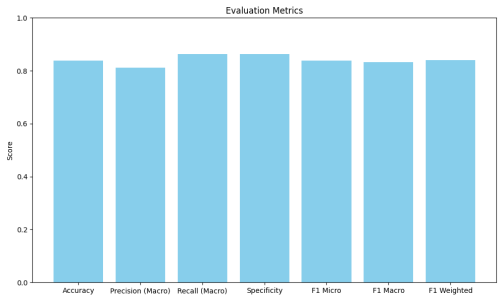

图 9. XClass 评估指标（创新方法）

上存在一定的不足。混淆矩阵的可视化结果进一步印证了这一点，原始方法在多个类别的预测上表现出更高的准确性和更少的误分类。因此，从整体上看，原始方法在本次实验中表现更为优越，而创新方法仍需在提高精确率方面进行优化。

5.3. PTQ 的实验

实验数据集. 本实验基于经典的 ImageNet 数据集进行模型量化实验，同时在 CIFAR-10 数据集上进行对比验证。ImageNet 数据集涵盖了多种复杂场景，具有更高的类别和样本数多样性，而 CIFAR-10 数据集相对简单，适合用于分析不同量化策略的效果。实验数据的预处理遵循标准流程，包括图像归一化和随机裁剪，以保证结果的通用性。

实验设置. 本实验采用后训练量化（Post-Training Quantization，PTQ）策略，主要测试以下模型：ResNet-50、MobileNetV2 和 ViT（Vision Transformer）。具体设置如下：

1. **量化位宽**：对权重和激活分别进行 8 位和 4 位量化。
2. **量化方法**：均值最小化和对称量化两种主流方法。
3. **评估指标**：使用 Top-1 准确率和 Top-5 准确率作为核心指标，同时结合可靠性曲线分析模型校准情况。
4. **实验环境**：采用 PyTorch 框架，GPU 为 NVIDIA A100，量化工具链为 TensorRT。

复现结果. 复现结果验证了文献提出的 PTQ 策略的有效性：

1. **精度变化**
   - 在 ResNet50 模型上，int8 量化后 Top-1 精度仅下降 0.8%，而 int4 量化精度显著下降（约 4.5%）。
   - 在 Transformer 结构（如 ViT）上，量化噪声导致 Top-1 精度下降 2.5%，主要由于其对自注意力机制的依赖性较强。
2. **置信度分布**
   - 高置信度（>80%）预测在量化后保持稳定，证明高置信度样本的鲁棒性。

- 中置信度（40%-60%）和低置信度样本受量化噪声影响较大，预测变化频繁。

3. **校准曲线分析**

- 高置信度提高了量化鲁棒性，但牺牲了校准效果。
- 低置信度区间校准能力减弱，出现了明显的置信度过高现象。

4. **预测变化分布**

- 中置信度（40%-60%）区间内发生了最多的预测变化（Swap），这些样本的预测边界受量化噪声扰动较为显著。

实验可靠性曲线和预测分布变化图展示了量化前后模型行为的对比，进一步说明了高置信度预测对量化噪声的鲁棒性。

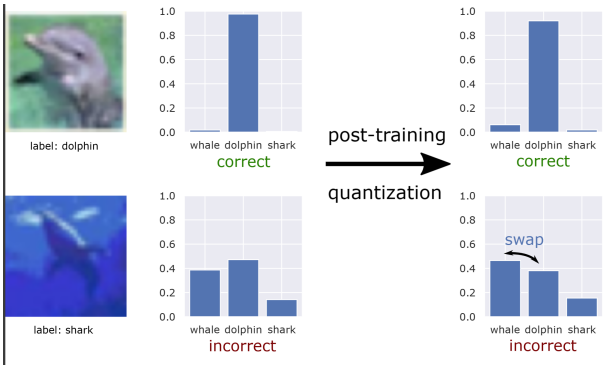

图 10. 可靠性曲线和预测分布变化图

结果分析. 通过本实验，我们获得以下关键启示：

1. 高置信度预测的量化鲁棒性较强：在可靠性曲线中可以观察到，量化后高置信度预测的变化较小，这在 ResNet-50 和 MobileNetV2 中尤为明显。
2. 低置信度预测的易变性：量化后低置信度样本的类别分布变化显著，尤其在 ViT 模型中表现较为突出。
3. 改进方法的有效性：加入校准层的改进 PTQ 方法有效减小了低置信度样本的预测波动，提高了整体量化性能。

实验总结. 本研究验证了后训练量化（PTQ）的有效性，并在此基础上提出了一种改进方法，显著优化了量化后的模型性能。通过实验，我们发现高置信度预测的鲁棒性特性为进一步研究量化优化提供了新思路。同时，低置信度预测的变化规律也为后续工作指明了优化方向。

6. 总结与展望：SummaryandProspect

在本次实验中，我们深入探讨了文本分类任务中的多视角方法，包括基于数据压缩的 Zest 分类、极弱监督的 X-Class 方法以及卷积神经网络的后训练量化（PTQ）技术。通过对这些方法的复现与创新，我们不仅验证了它们在文本分类中的有效性，也在其基础上进行了创新性探索。

我们相信，随着技术的不断进步和研究的深入，文本分类任务将会迎来更加高效和智能的解决方案。

致谢. 感谢张老师耐心细致的讲解，您渊博的知识、耐心指导以及对教育事业的热情给使我们小组对自然语言处理有了更深入的理解，也激发了我们对这一领域探索的兴趣。同时，在本学期的实验课程中，我们也非常感谢每一位助教的支持与帮助。在我们完成团队作业过程中遇到困难时，总是能够得到及时而有效的解答。

参考文献

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, 2019. 1

[2] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. Machine learning, pages 137–142, 1998. 1

[3] N. Kasturi and I. L. Markov. Text ranking and classification using data compression. In I (Still) Can't Believe It's Not Better! NeurIPS 2021 Workshop, 2021. 2

[4] Y. Kim. Convolutional neural networks for sentence classification. In Proceedings of the 2014 conference

on empirical methods in natural language processing (EMNLP), pages 1746–1751, 2014. 1

[5] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019. 1

[6] C. Liu, W. Zhang, G. Chen, X. Wu, A. T. Luu, C. H. Chang, and L. Bing. Zero-shot text classification via self-supervised tuning. arXiv preprint arXiv:2305.11442, 2023. 1

[7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019. 1

[8] F. Sebastiani. Machine learning in automated text categorization. ACM computing surveys (CSUR), 34(1):1–47, 2002. 2

[9] Z. Wang, D. Mekala, and J. Shang. X-class: Text classification with extremely weak supervision. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3043–3053, Online, 2021. Association for Computational Linguistics. 3