2025《人工智能导论》大作业

任务名称：　谣言检测

小组人员：黄兰婷、刘畅、 龚玥琪、曹芊

完成时间：　2025.06.07

1．任务目标

基于谣言检测数据集，构建一个二分类检测模型，0 代表非谣言、1 代表谣言。实现对推文谣言属性的高效判定。模型需满足以下要求：尽量提高分类准确率；模型具有一定的泛化能力；有合理的运行时间。

2．具体内容

（1）实施方案

1）实验数据

数据集构成：

训练集、验证集、测试集数据使用 csv 格式存储，表头为 id,text,label,event，其中：id 表示推文 id，text 表示推文内容，label 中 0 代表非谣言，1 代表谣言 ，event 表示事件主题类别。原始训练集 train.csv 共 2870 条文本，验证集共 406 条文本，训练集与验证集之比约为 7:1，其中谣言与非谣言之比约为 1.2：1。为增加其他主题的推文内容，我们引入了部分非同源数据对原训练集进行扩容，得到的新数据集 train_new.csv 共 3900 条数据，与验证集之比约为 9.5：1，谣言与非谣言之比为 1.2：1。测试集选择和原测试集不同源的数据，共 126 条。

数据预处理：

首先对数据进行了清洗，包括去除无关字符（如 URL 链接、特殊符号等），以及对文本进行标准化处理（如统一小写）。

实验中替换用户名为@user，发现微弱提升效果。为了测试模型泛化性，使用 Bart-large-cnn 进行同义词替换和改写，生成 val_augmented 验证集。

2）模型构建

选择基于 Transformer 架构构建模型，它能高效处理长文本，利用自注意力机制并行计算单词间关系，捕捉文本特征。模型首先采用词嵌入层将离散文本符号转换为稠密向量表示，并引入位置编码捕获序列顺序信息；随后通过多层 Transformer 编码器进行特征提取，利用多头自注意力机制建模文本内部的复杂语义关联。

模型设计了可扩展的事件信息融合模块，支持选择是否加入 event 信息进行训练，可通过嵌入层将事件 ID 映射为特征向量后与 Transformer 输出的全局语义表示（[CLS]向量）进行拼接。最终通过带有 Dropout 正则化的双层全连接网络输出谣言概率预测值。

（2）核心代码分析

1）接口类实现

在初始化方法__init__中，首先根据 use_event 参数确定是否加载使用事件的训练模型。随后加载词汇表 vocab.json 获取词表映射关系，并读取 num_events.json 确定事件数量。为每个模型创建 TransformerRumorDetector 实例，加载预训练权重到指定设备，并设置为评估模式。

preprocess 方法实现文本预处理：先进行文本清洗和分词，添加特殊标记<CLS>和<SEP>，通过词表转换为 ID 序列，最后填充/截断至固定长度(MAX_LEN)并转为 PyTorch 张量。

classify 将输入文本预处理后，通过 classify 调用集成模型进行预测。

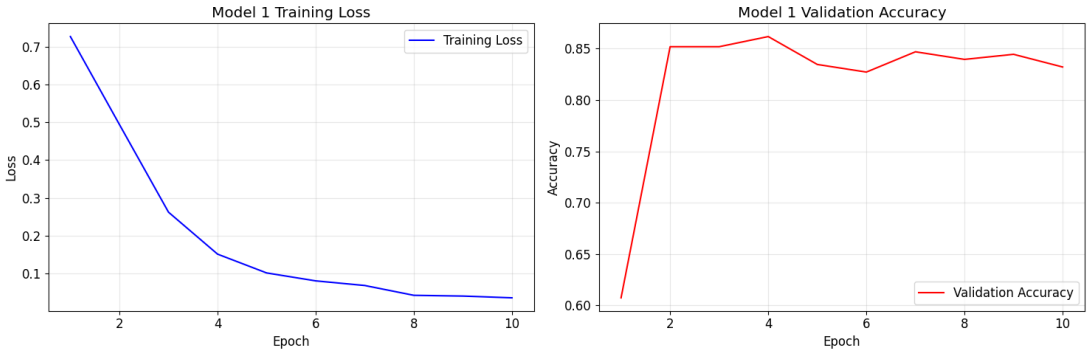
test_csv 方法提供批量测试功能。

2）模型训练

首先解析命令行参数确定是否使用 event 信息，加载数据。通过 build_vocab()构建词汇表并保存，同时统计事件数量。创建 RumorDataset 加载器后，启动多模型训练循环，为每个模型设置独立随机种子确保多样性，初始化 TransformerRumorDetector 实例，使用 AdamW 优化器进行训练。

定义 BCEWithLogitsLoss 损失函数，配合 ReduceLROnPlateau 学习率调度器和预热机制。在训练循环中执行标准流程：数据加载→前向传播→损失计算→梯度裁剪→反向传播→参数更新。每个 epoch 结束后在验证集评估性能，保存最佳模型并记录指标。

evaluate_single()计算单一模型准确率，evaluate_ensemble()实现集成模型评估，通过平均多个模型的 sigmoid 概率输出最终预测。训练完成后，加载所有最佳模型进行集成评估，对比单一模型与集成模型的性能差异。训练过程中，主要调节 lr 学习率、MAX_LEN 与 epoch。最后设置 lr=5e-4，MAX_LEN = 64。



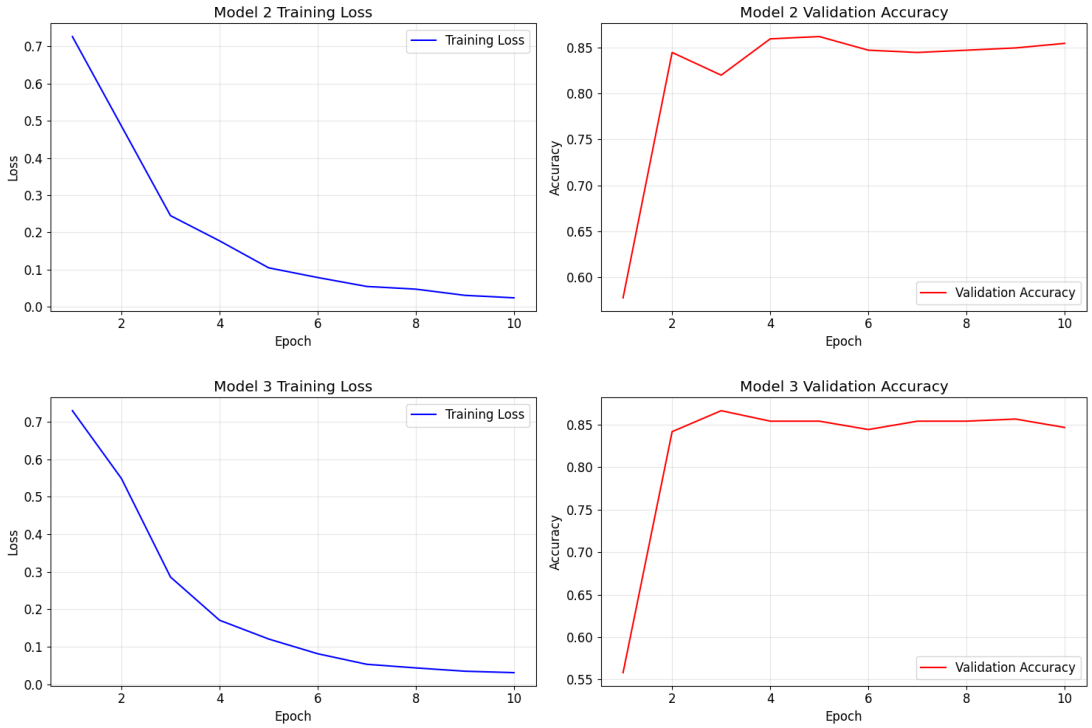


图 1 train_loss/ val_acc 随 epoch 增长的变化曲线

（3）测试结果分析

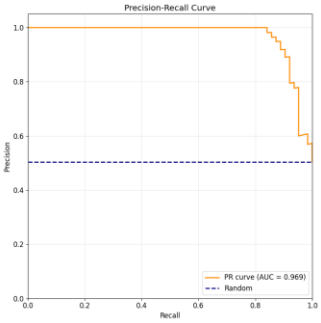
我们测试了加入 event 进行训练和不加入、单一模型与集成模型、加入新训练集与不加入新训练集三种情况，结果见下表：

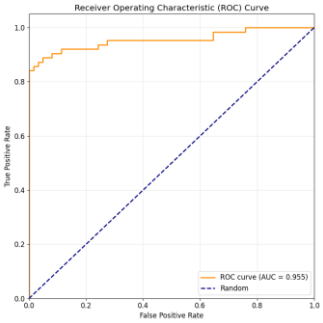
表 1 各种训练方法模型预测准确率

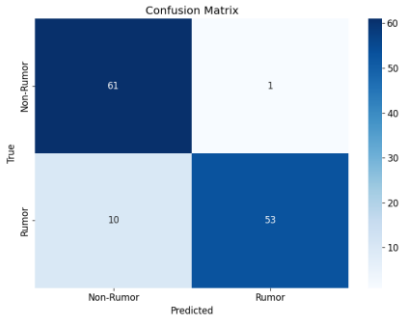
训练集	*训练方式*	*事件信息*	*val acc*	*test acc*	*备注*
原始训练集 train.csv	集成学习	加入	87.65% (集成) 86.17% (最佳单一)	64.00%	testing.csv 无 event 类型，测试时默认 event=0
		不加入	87.16% (集成) 86.67% (最佳单一)	66.40%	
	非集成学习	加入	86.17%	68.00%	
		不加入	85.19%	72.00%	
train_new.csv	集成学习	不加入	88.40% (集成) 86.67% (最佳单一)	91.20%	不能加入 event 进行训练，因为 train_new 中事件类型并不准确，是用默认值填充得到
	非集成学习	不加入	84.69%	87.20%	

下图为最终选择使用扩充后训练集、不加入 event 信息使用集成训练模型在

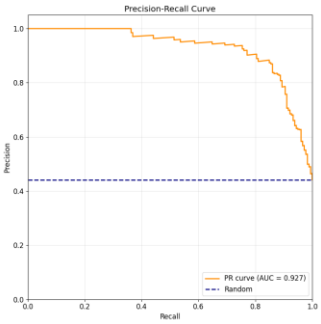
val.csv、val_augemented.csv、testing.csv 上的预测情况。

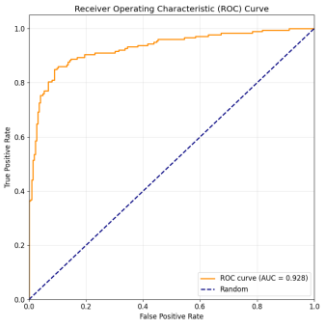


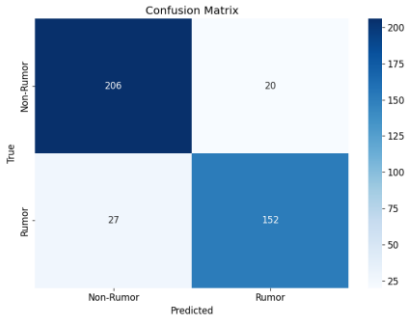




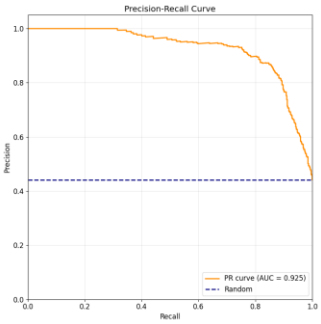
test Precision-Recall 曲线 ROC 曲线 Confusion Matrix

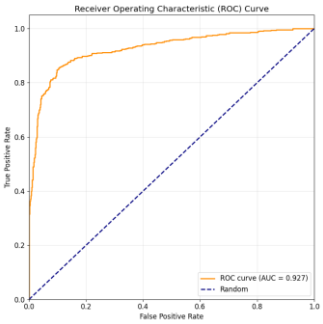


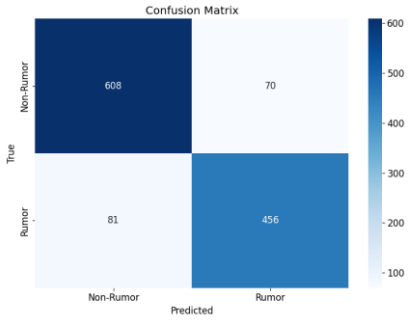




val Precision-Recall 曲线 ROC 曲线 Confusion Matrix







val_augmented Precision-Recall 曲线 ROC 曲线 Confusion Matrix

3. 工作总结

（1）收获、心得

通过动手构建谣言检测模型，我们对深度学习在自然语言处理领域的应用有了更直观的认识，加深对模型运作机制的理解。

此外，我们学习了文本数据清洗、预处理流程，积累模型训练经验，锻炼了自主学习和查阅资料的能力。

（2）遇到问题及解决思路

在训练中主要遇到的问题是原训练集数据不足。

最初使用提供的训练集进行训练，并尝试利用 ai 生成的测试集进行测试，发现模型表现不佳，准确率仅为 55%左右。考虑到所给数据集中的文本涉及主题较少，担心模型在处理与训练集中不同主题的文本内容时泛化能力不足，决定对训练集进行扩增。

https://github.com/OwenLeng/rumor-detection-include-twitter15-twitter16data-中使用的推文可信度训练、验证和测试集，该数据源提供的数据仅含有推文 id、文本内容 text 和标签（true、false、non-rumor、unverified）三项。使用 python 脚本读取该项目中数据集内容，选择其中标签为 true、false 的条目，设定 label 为 0、1，补全 event 项，与.csv 格式进行适配，最终截取 126 条数据作为测试集 testing.csv，余下部分全部添加至原训练集中，生成新的训练集 train_new.csv。使用改良后的训练集得到的模型在测试集中表现增强。

4．课程建议

增加实践案例，设置更多不同类型项目实战，强化理论与实践结合，提升解决实际问题能力。