SMP-EWECT 技术报告

李煜东[1,2]，赵哲[1]，杨雪峰[1]，刘海哮[1]，赵欣[1,3]，

周鹏[1]，刘伟杰[1]，鞠奇[1]

Tencent Oteam[1]

中国地质大学（北京）[2]

中国人民大学[3]

摘要

微博情绪分类是一项经典的文本分类任务，我们基于 BERT 等预训练模型，随机选择和组合增量预训练、半监督学习、对抗训练、数据增强、模型融合等各种训练策略，在无需过多人工介入的情况下以自动化的方法训练出了大量具有低偏差-高方差的分类模型，对这些模型进行深层模型融合后分别在 usual 和 virus 测试集上获得了 0.7856 和 0.7078 的分数，排名第一名。

关键词：自然语言处理，情感分析，预训练模型，模型融合

1.引言

微博情绪分类任务旨在识别微博中蕴含的情绪，输入是一条微博，输出是该微博所蕴含的情绪类别。在本次评测中，微博按照其蕴含的情绪分为以下六个类别之一：积极、愤怒、悲伤、恐惧、惊奇和无情绪。微博情绪分类评测任务一共

包含两个测试集：第一个为通用微博数据集，其中的微博是随机收集的包含各种话题的数据；第二个为疫情微博数据集，其中的微博数据均与本次疫情相关。

每条微博被标注为以下六个类别之一：neural（无情绪）、happy（积极）、angry（愤怒）、sad（悲伤）、fear（恐惧）、surprise（惊奇）。

通用微博训练数据集包括 27,768 条微博，验证集包含 2,000 条微博，测试数据集包含 5,000 条微博。

疫情微博训练数据集包括 8,606 条微博，验证集包含 2,000 条微博，测试数据集包含 3,000 条微博。

评测以宏平均 F1 值作为评测指标，最终，会对通用微博测试集的测试结果和疫情微博的测试结果进行平均，作为最终的测试结果，即：

$$P_e = \frac{TP_e}{TP_e + FP_e}, R_e = \frac{TP_e}{TP_e + FN_e}$$

$$F_e = \frac{2 \times P_e \times R_e}{P_e + R_e}$$

$$Macro\_F = \frac{1}{n} \sum_{e \in Emotions} F_e$$

$$Macro\_F_{final} = \frac{(Macro\_F_{通用} + Macro\_F_{疫情})}{2}$$

其中$e \in Emotions, Emotions = \{happy, angry, sad, fear, surprise, neural\}$

2.数据分析及预处理

2.1 Early Data Analysis

首先分析文本的长度，如图 1、图 3 所示，大部分数据长度在 200 字符以内，95%的样本长度在 128 字符以内，99%的样本长度在 192 字符，其中有少量的样本长度达到 1000 以上，我们对多余部分使用截断处理。

数据分为 6 个类别，如图 2、图 4 所示，其中 natural、angry、happy 占数据的主要部分（分别占 usual 70.1%，virus 83.8%），由于评估指标使用的是 macro F1，准确区分数据量少的类将是提升得分的关键。

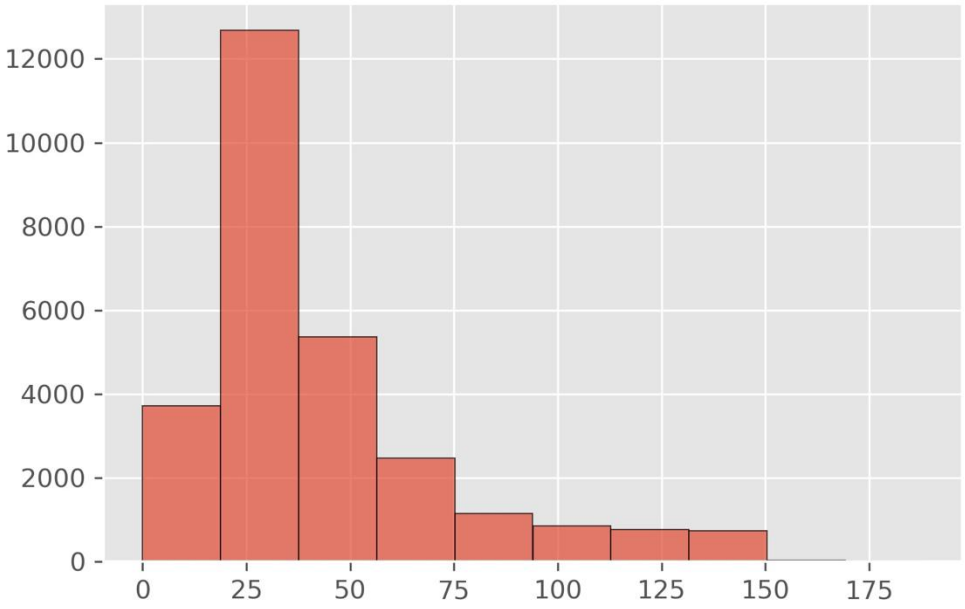

图 1 usual 长度分布

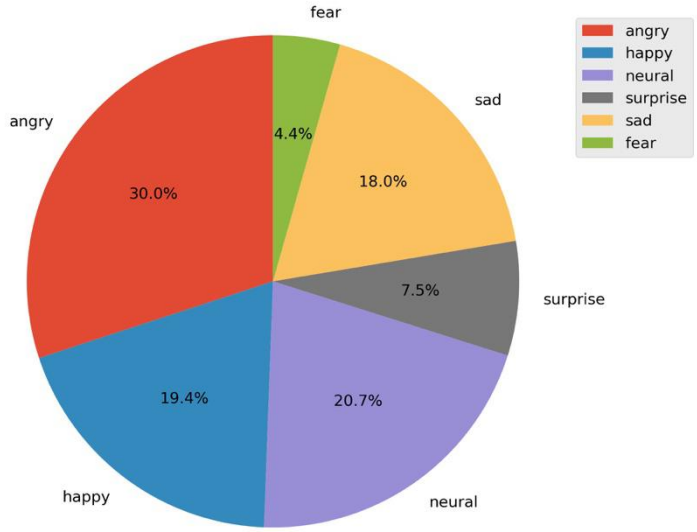

图 2 usual 标签分布

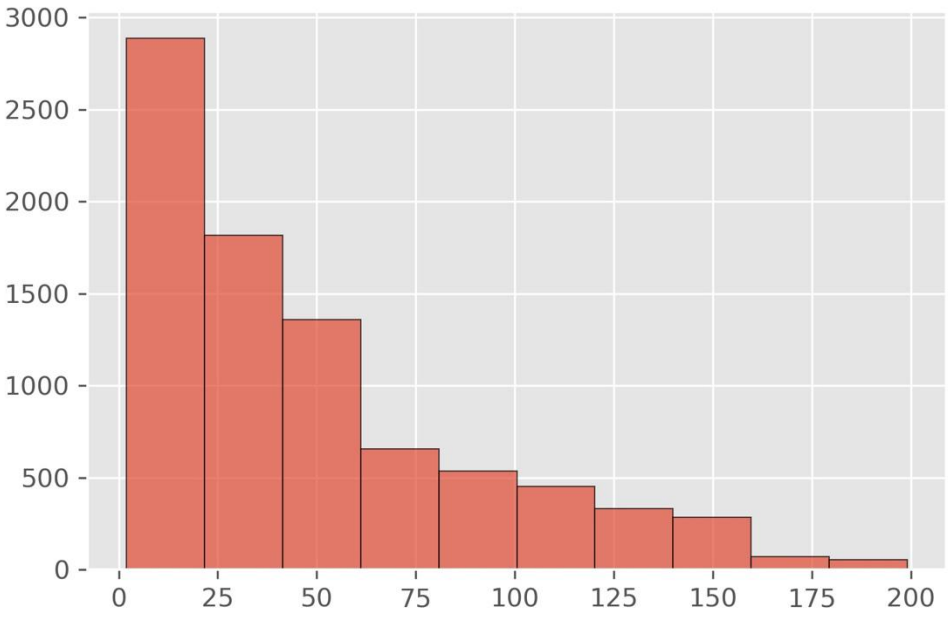

图 3 virus 长度分布

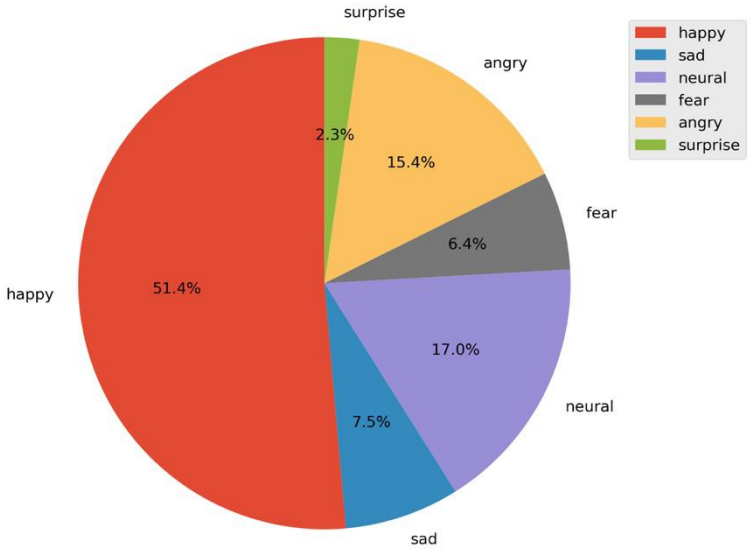

图 4 virus 标签分布

2.2 Data Preprocessing

原始数据中带有大量的标记，对分析文本的情感倾向没有帮助（例如用户 ID，转发标记、URL 等），我们利用正则表达式过滤了以上标记，并且将 emoji 表情翻译成中文释义。例如：

预处理前：

终于可以在❄纷飞的时候蹲在街边吃热乎乎的烤🍠了！

唉//@一只惨白:还有就是隔离。后来是发现有一点症状就隔离。不然可能传播得更广。 http://t.cn/A6vBFriq

预处理后：

终于可以在[雪花]纷飞的时候蹲在街边吃热乎乎的烤[红薯]了！

唉 还有就是隔离。后来是发现有一点症状就隔离。不然可能传播得更广。

在后续的训练中，我们同时使用了经过预处理和不经过预处理的数据作为模型输入。

3.相关工作

3.1 UER

UER[1-2]是我们的自研预训练框架，旨在快速复现各种预训练模型，UER 对 BERT 等一系列预训练模型进行了精准的复现，包括预处理、预训练、下游任务微调等步骤，并增加了多项预训练相关的功能和优化。 UER 提供与 Google BERT、Huggingface 权重的转换脚本，支持大部分开源预训练模型。

3.2 Pretrained Model Zoo

我们收集了各种开源预训练模型，根据规模和参数量重新分类，在规模命名上沿用 Google BERT 提供的标准[3]：

	H=128	H=256	H=512	H=768	H=1024
L=2	**2/128 (BERT-Tiny)**	2/256	2/512	2/768	2/1024
L=4	4/128	**4/256 (BERT-Mini)**	**4/512 (BERT-Small)**	4/768	4/1024
L=6	6/128	6/256	6/512	6/768	6/1024
L=8	8/128	8/256	**8/512 (BERT-Medium)**	8/768	8/1024
L=10	10/128	10/256	10/512	10/768	10/1024
L=12	12/128	12/256	12/512	**12/768 (BERT-Base)**	12/1024
L=24	24/128	24/256	24/512	24/768	**24/1024 (BERT-Large)**

我们收集到的模型有：

Organization	Model	Parameters	Vocab
Google	BERT_Base	101M	21128
Google	BERT-multi-cased_Base	177M	119547
Google	BERT-multi-uncased_Base	166M	105879
Baidu	ERNIE-1_Base	99M	18000
Baidu	ERNIE_Tiny	89M	50006
Zhuiyi	RoBERTa_Tiny	4M	21128
Zhuiyi	RoBERTa_Small	89M	21128
Zhuiyi	SimBERT_Base	96M	13685
HIT	rbtl3_Medium	38M	21128
HIT	rbt3_Small	60M	21128
HIT	BERT-wwm-ext_Base	101M	21128
HIT	BERT-wwm_Base	101M	21128
HIT	RoBERTa-wwm-ext_Base	101M	21128
HIT	RoBERTa-wwm-ext_Large	324M	21128
CLUE	RoBERTa_Mini	24M	8021
CLUE	RoBERTa-pair_Mini	24M	8021
CLUE	RoBERTa_Tiny	6M	8021
CLUE	RoBERTa_Small	28M	8021
CLUE	RoBERTa_Large	311M	8021
CLUE	RoBERTa-pair_Large	311M	8021
Tencent	UER_mixed_Base	101M	21128
Tencent	UER_mixed_Tiny	14M	21128
Tencent	UER_mixed_Small	30M	21128
Tencent	UER_mixed_Large	324M	21128
Tencent	AILab-BERT_Base	98M	16541

4.模型介绍

在比赛中，常规的思路是通过深入理解数据集以及大量的消融实验，以验证不同技术和技术组合的有效性，并对多个高质量的模型进行集成。然而由于技术以及技术的组合种类繁多，这种精细的方式能够探索的空间有限，最终得到的模型数量也有限。在这次比赛中，我们使用更加粗放的实验方式。通过随机选择和组合不同技术（如图 5 所示），得到大量的得到性质不同的分类模型，进行模型集成。这种方法能够以很少的人力成本，持续的堆积模型，增加系统的参数量。最终，我们训练了数百个分类模型，得到了超过 3000 亿（1TB）参数的系统。由于我们的技术主要围绕着 BERT[4-6]展开，因此我们把这个系统叫做 BERT forest。据我们所知，目前很少有工作会针对一个具体的任务，使用超过千亿级别参数量。我们的实验证明了在已有的工作上持续的增加参数量，仍然能取得可观的收益。

我们使用了大量中文预训练模型，通过不同的训练策略进行微调，从而得到大量的具有低偏差-高方差的基础模型特征，并在这些特征上进行 stacking。

我们从 model zoo 中选取模型进行微调和增量训练，此外还使用了 LSTM[7]、GPT[8]、Gated CNN[9]等模型，基于这些模型，随机组合了使用的数据集、文本长度、损失函数、对抗训练等训练策略，为 usual 和 virus 分别提取超过 300 个基模型特征用于 stacking 融合学习。

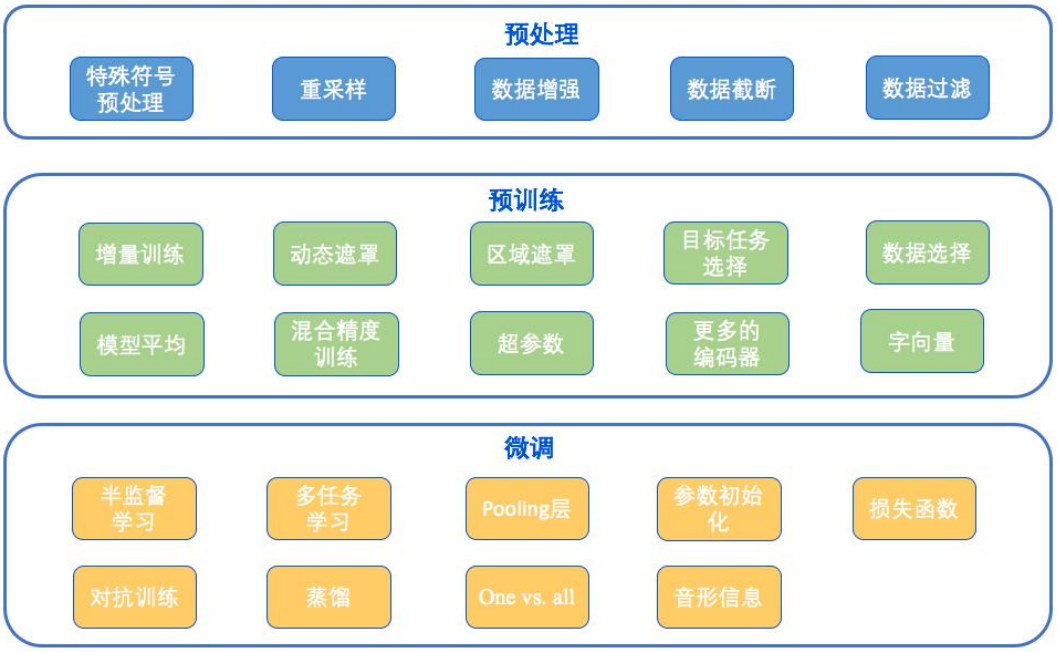

图 5 训练策略

4.1 预训练

UER 提供预训练模式，能够方便的对语料集进行预处理，以及选择预设的训练策略进行多机多卡增量训练。

我们使用的外部语料有：网络上收集的 4000 万条微博语料[10]、OpenKG 疫情相关知识图谱中提取的语料[11]、爬取的疫情相关微博以及从开放数据集中提取的情感相关语料[12]，结合这些语料和动态遮罩、区域遮罩、数据选择等训练策略得到增量训练模型[13]。

4.1.1 动态遮罩

在 BERT 的静态遮罩方案中，序列仅在预处理时被遮罩一次，这样每个序列都对应了固定的 mask。动态遮罩在预训练时随机遮罩，相较于静态方法能够获得稍好的效果[14]。

4.1.2 区域遮罩

区域遮罩从几何分布中采样遮罩的长度，随机遮罩连续一段字，能够提升预训练的难度[15]。

4.1.3 数据选择

我们使用数据选择，从大规模语料中分别选择和通用数据集/疫情数据集相关的语料，作为增量预训练的语料（如图 6 所示）。这里我们使用了 3 种数据选择策略：

1）基于分类器的数据选择。我们使用 BERT-tiny 作为分类器。正样本由下游任务构成、负样本由语料随机采样构成。分类器训练好之后，我们将大规模语料经过分类器，把高于指定分数的样本作为预训练语料。

2）基于统计信息的数据选择。我们使用 Spearman rank 衡量语料之间的相似度。我们假设相似的语料有着相似的词频排序。我们把语料划分成 block，一个 block 包含 10MB 语料。我们计算下游任务的词频排序和 block 中语料词频排序的 Spearman rank，把 Spearnman rank 高于指定阈值的语料作为预训练语料。

3）基于关键词的数据选择。我们在疫情数据集上使用 TF-IDF 得到权重较高的单词，通过这部分单词在大规模语料中进行检索，得到包含这些单词的句子，以组成预训练语料。

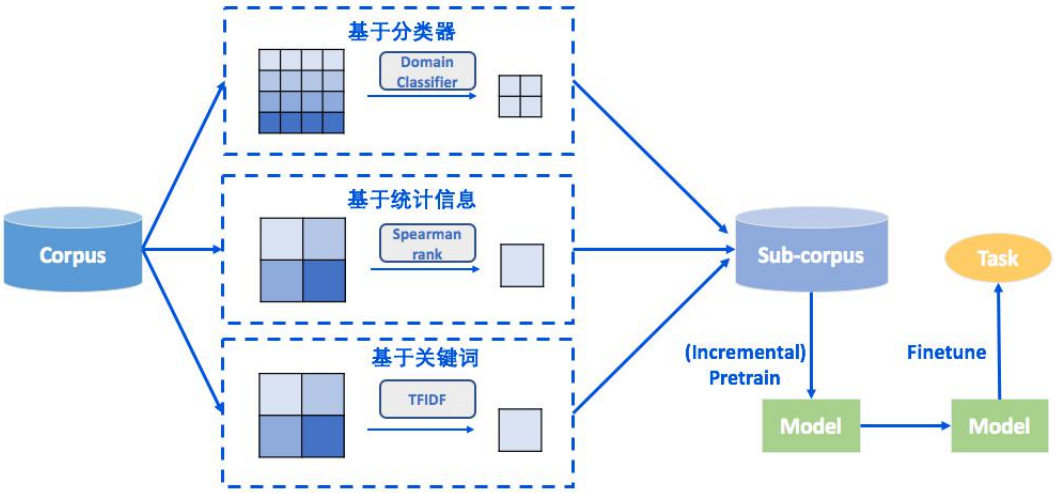

图 6 数据选择

4.2 数据增强

我们使用了数据增强方法来扩充数据集

4.2.1 简单数据增强

通过对样本进行随机删除、随机调换单词位置、同义词替换来扩充数据集[16]。

4.2.2 BERT cloze 数据增强

利用 BERT 模型的掩码语言模型特性，依次单独 mask 句子中的每个 token，再用 BERT 模型预测 token 可能的候选字，从原句子中选择 M 个位置随机替换成 N 个最有可能候选字中的一个，这样就生成了一个增强样本[17]，我们选取了多个 M 和 N 的组合来构建增强数据集。

例如：

原句子：晒个被子收回来的时候上面全是烟灰

增强样本：晒个被子，起来的时候上面全是烟熏
增强样本：晒着被子收回来的东候上面全的烟灰
增强样本：晒个被子收出来的时候上的全是烟头

4.2.3 回译数据增强

通过将中文翻译到外文再翻译回中文，能够在不改变句子原始含义的情况下改写句子，实现数据增强。我们利用机器翻译，将原始数据通过中-英-中、中-法-中和中-英-法-中分别构建了 3 份回译数据集。

4.2.4 重采样

由于数据集存在样本不均衡问题，一些样本数量较少的类别会出现学习不充分的情况，我们使用重采样的方法保证所有类别的样本数量一致。分为 2 种情况：

1. 下采样，构建训练数据时对样本数量较多的类别只选取其中一部分用于训练

2. 上采样，将样本数量少的类别中的样本重复多次放入数据集

4.3 Tricks

本节介绍我们在微调预训练模型时使用的技巧

4.3.1 对抗训练

NLP 中的对抗训练主要通过在 Embedding 层增加扰动，提升模型的泛化能力，对抗训练方法有 FGM[18]、PGD[19]、FreeAT[20]、YOPO[21]等，我们在 UER 中实现了其中的 FGM 方法和 PGD 方法 [22]。

4.3.2 半监督学习

使用的半监督训练[23-24]分为两种情况 1.使用任务语料在预训练模型上增量预训练，这样能够帮助预训练模型熟悉下游任务语料的分布 2. 将下游任务模型预测的验证集结果加入训练集中用于微调预训练模型。

4.3.3 多任务学习

同时微调多个下游任务,这些下游任务共享 Embedding 层和 Encoder 层参数。对于 usual 数据集，使用 virus 作为辅助任务；对于 virus 数据集，使用 usual 作为辅助任务[25]。

4.3.4 One Versus All

OVA 将之前的六分类问题转换成二分类问题来训练模型，对于每一个类别都训练一个分类器来判断是否属于该类别（例如是否为积极情绪），综合六个分类器的得分得出最终的判断结果。

4.3.5 损失函数

交叉熵是分类问题常用的损失函数，交叉熵对于所有类别的样本都有一样的权重，但是在一些情况下，我们希望 1. 模型更加关注样本数量少的类别 2. 模型更关注困难的样本。因此，除了交叉熵以外，我们还使用了 focal loss 和一个修正交叉熵损失函数[26]。

4.3.6 蒸馏

我们使用模型的集成去蒸馏单模型[27]。这里我们蒸馏的方式是优化 logits 的 MSE 损失函数。蒸馏的 transfer set 有多种选择。通常来说，我们希望蒸馏的 transfer set 和下游任务有着相似的数据分布。我们使用的 transfer set 包括：1）下游任务训练集、验证集、测试集；2）通过数据选择，在大规模语料中选择的和下游任务数据分布相似的数据。

5.模型融合

我们的框架使用 4 层模型，3 次融合的方案，在第一层使用大量预训练模型作为基模型特征，第二层使用 LightGBM 作为次模型特征，第三层使用 Linear Regression 之后平均作为最终结果。

其步骤为

1. 基模型：使用 Stacking 步骤对训练集进行 k 折训练，每轮训练后预测留出的验证集作为对训练集特征，并预测测试集平均后作为测试集特征。由于比赛采用 AB 测试集，需要保留训练的 checkpoint 用来预测 B 测试集，我们训练的大量模型对存储空间造成了一定压力，因此只保留 30%的模型 checkpoint 用于平均作为测试集结果，经过实验与 100%的 checkpoint 效果近似。

2. LightGBM 次模型：使用 LightGBM 进行第一次融合，将基模型特征通过有放回抽样分为 5 组，每组以 Stacking 步骤单独训练一个 LightGBM 模型作为第二层特征，LightGBM 通过贝叶斯优化确定超参数。

3. Linear Regression 次模型：利用线性回归模型对 LightGBM 模型进行第二次融合，使用十折每次取 90%的数据用于训练，留出 10%数据用于验证，最后提交的结果是对验证集／测试集预测的平均结果。

4. Averaging：对十折线性回归模型的结果平均作为最终结果。

5.1 贝叶斯优化

贝叶斯优化假设超参数与需要优化的损失函数存在一个函数关系，给定优化的目标函数，通过不断地添加样本点来更新目标函数的后验分布。我们使用开源

框架 BayesianOptimization 实现 LightGBM 超参数搜索[28]，用于确定模型中树最大深度、叶子最小记录数、学习率、正则化等参数。

6.实验结果及分析

单模型与 BERT Forest 使用不同的方法融合的对比结果：

	通用微博 F1	通用微博 Acc	疫情微博 F1	疫情微博 Acc
BERT 单模型	0.7402	0.7870	0.6476	0.7830
增量预训练 BERT 单模型	0.7534	0.7865	0.6634	0.7960
BERT Forest ＋ Average	0.7856	0.8085	0.6619	0.7930
BERT Forest ＋ LightGBM	0.7966	0.8145	0.6769	0.8065
BERT Forest ＋ LGB ＋ LR	0.8004	0.8200	0.6976	0.8110

7.参考

[1] Zhao Z, Chen H, Zhang J, et al. UER: An Open-Source Toolkit for Pre-training Models[J]. arXiv preprint arXiv:1909.05658, 2019.

[2] https://github.com/dbiir/UER-py

[3] https://github.com/google-research/bert

[4] https://towardsdatascience.com/how-to-make-the-most-out-of-bert-finetuning-d7c9f2ca806c

[5] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[6] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.

[7] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[8] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI Blog, 2019, 1(8): 9.

[9] Dauphin Y N, Fan A, Auli M, et al. Language modeling with gated convolutional networks[C]//International conference on machine learning. 2017: 933-941.

[10] https://bbs.hankcs.com/t/topic/42

[11] http://www.openkg.cn/organization/o1

[12] https://github.com/CLUEbenchmark/CLUEmotionAnalysis2020

[13] Sun C, Qiu X, Xu Y, et al. How to fine-tune bert for text classification?[C]//China National Conference on Chinese Computational Linguistics. Springer, Cham, 2019: 194-206.

[14] Liu Y, Ott M, Goyal N, et al. Roberta: A robustly optimized bert pretraining approach[J]. arXiv preprint arXiv:1907.11692, 2019.

[15] Joshi M, Chen D, Liu Y, et al. Spanbert: Improving pre-training by representing and predicting spans[J]. Transactions of the Association for Computational Linguistics, 2020, 8: 64-77.

[16] Wei J, Zou K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks[J]. arXiv preprint arXiv:1901.11196, 2019.

[17] Jiao X, Yin Y, Shang L, et al. Tinybert: Distilling bert for natural language understanding[J]. arXiv preprint arXiv:1909.10351, 2019.

[18] Miyato T, Dai A M, Goodfellow I. Adversarial training methods for semi-supervised text classification[J]. arXiv preprint arXiv:1605.07725, 2016.

[19] Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks[J]. arXiv preprint arXiv:1706.06083, 2017.

[20] Shafahi A, Najibi M, Ghiasi M A, et al. Adversarial training for free![C]//Advances in Neural Information Processing Systems. 2019: 3358-3369.

[21] Zhang D, Zhang T, Lu Y, et al. You only propagate once: Accelerating adversarial training via maximal principle[C]//Advances in Neural Information Processing Systems. 2019: 227-238.

[22] https://zhuanlan.zhihu.com/p/91269728

[23] Chen B, Huang F. Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data[C]//Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning. 2016: 314-323.

[24] Dai A M, Le Q V. Semi-supervised sequence learning[C]//Advances in neural information processing systems. 2015: 3079-3087.

[25] Hashimoto K, Xiong C, Tsuruoka Y, et al. A joint many-task model: Growing a neural network for multiple nlp tasks[J]. arXiv preprint arXiv:1611.01587, 2016.

[26] https://spaces.ac.cn/archives/4293

[27] Sanh V, Debut L, Chaumond J, et al. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter[J]. arXiv preprint arXiv:1910.01108, 2019.

[28] https://github.com/fmfn/BayesianOptimization