A Multi-Label Sentiment Classification Approach using ERNIE 3.0 Pre-trained Model

Xinhui Lin

School of Computer Science and Engineering, Northeastern University

Introduction & Motivation

Sentiment analysis plays a vital role in natural language processing (NLP) tasks. Over recent years, the landscape of sentiment analysis tasks has evolved from basic multi-class problems, such as positive-negative-neutral classification, to more intricate multi-label problems. These advanced tasks require classifiers to identify and output specific emotions contained within the text. This shift poses challenges for language models.

The predominant approach in the NLP field involves employing pre-training and fine-tuning strategies. This method typically includes constructing a model with numerous parameters, undertaking self-supervised autoregression training on a large general corpus, and subsequently fine-tuning it on specific datasets tailored to particular tasks.

In this study, the ERNIE 3.0 base model is employed to conduct multi-label sentiment analysis on a given dataset.

Dataset Preprocess

Figure 1. train dataset demonstration

Validate the algorithm's viability and evaluate its performance on the provided dataset. The training dataset comprises 26,000 entries, each presented as [id, text, label]. Thirty percent of the data is randomly assigned for validation, with the remaining 70% designated as the training set. For each sample (a row in the CSV file), the text field undergoes processing, including trimming leading and trailing whitespaces, eliminating spaces from pre-tokenization, and subsequent input into a tokenizer to generate word vectors. The labels undergo one-hot encoding.

Following this preprocessing, the pre-trained model undergoes fine-tuning on the partitioned training set, and the optimal model is selected through validation. The saved optimal model is then utilized for inference on the testings set, consisting of 6,153 entries in the format [id, text]. The model outputs the one-hot encoded labels corresponding to the text field, subsequently binarized into readable labels based on a specified threshold.

Code Availability

The data, code, and models utilized in this project have been made available as open source.

The repository is accessible at https://github.com/hilinxinhui/kdd.

Should you have any inquiries or wish to provide feedback, kindly do so by submitting an issue on the repository.

Model & Algorithm

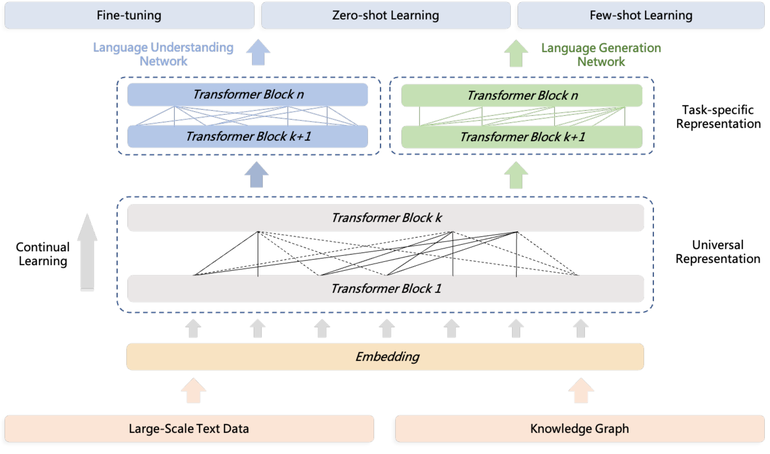

Figure 2. ERNIE 3.0 framework

Algorithm 1 Fine-tuning Stage

Require: Pre-trained model, labeled downstream task training set
Ensure: Fine-tuned model

1: Data cleaning: Remove invalid characters, pre-tokenization, generate word vectors for input text, and create one-hot encoding for labels.
2: Split the training set into training and validation sets.
3: Train the pre-trained model with input word vectors. Test the model on the validation set every 50 steps, measuring AUC, F1-Score, Precision, and Recall scores.
4: Save the model with the highest F1-Score on the validation set as the fine-tuned model.

Algorithm 2 Inference Stage

Require: Fine-tuned model, unlabeled downstream task test set
Ensure: Labels for the downstream task test set

1: Preserve all hyperparameters from data processing. Perform word embedding and one-hot encoding on the test set.
2: Use the fine-tuned model from Algorithm 1 to predict labels for the generated word vectors.
3: Apply the sigmoid activation function to the predicted labels and binarize according to a given threshold.
4: Generate original labels corresponding to the binarized results.
5: Save the labels for each sample in the test set.

Experiment

Hyperparameter	Value
Maximum Tokenizer Sequence Length	100
Test Set Batch Size	16
Validation Set Batch Size	16
Initial Learning Rate	2e-5
Learning Rate Decay Rate/Epochs	0.5
Optimizer	AdamW
Loss Function	BCEWithLogitsLoss
Fine-tuning Epochs	4
Inference Binarization Threshold	0.38

Table 1. table of hyperparameters

Metrics	Score
AUC	0.91337
F1-Score	0.67598
Precision	0.64911
Recall	0.70517

Table 2. performance on validation set

Future Works

1. **Optimizing Model Generalization:** Explore alternative models like TextCNN or RNNs on the limited dataset provided to achieve comparable test accuracies, addressing the challenge of model overfitting due to the mismatch between model parameters and dataset size.
2. **Adapting to Downstream Tasks Efficiently:** Future research can focus on fine-tuning specific layers of the base model to enhance adaptability to specific sentiment classification tasks, avoiding the introduction of unrelated tasks during pre-training.
3. **Enhancing Training and Inference Efficiency:** Efforts can be directed towards improving efficiency in both model training and inference. Consider using parallel hardware acceleration during training and implementing model quantization for performance optimization during inference.

References

[1] Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi.
Goemotions: A dataset of fine-grained emotions, 2020.

[2] Baidu Inc.
Official implementations for various pre-training models of ernie-family, covering topics of language understanding generation, multimodal understanding generation, and beyond.
`https://github.com/PaddlePaddle/ERNIE`, 2022.

[3] Baidu Inc.
PaddleNlp is a nlp library that is both easy to use and powerful.
`https://github.com/PaddlePaddle/PaddleNLP`, 2023.

[4] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang.
Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation, 2021.