Figure Eight at SemEval-2019 Task 3: Ensemble of Transfer Learning Methods for Contextual Emotion Detection

Joan Xiao
Figure Eight Inc.
San Francisco, CA 94103
joan.xiao@gmail.com

Abstract

This paper describes our transfer learning-based approach to contextual emotion detection as part of SemEval-2019 Task 3. We experiment with transfer learning using pre-trained language models (ULMFiT, OpenAI GPT, and BERT) and fine-tune them on this task. We also train a deep learning model from scratch using pre-trained word embeddings and BiLSTM architecture with attention mechanism. The ensembled model achieves competitive result, ranking ninth out of 165 teams. The result reveals that ULMFiT performs best due to its superior fine-tuning techniques. We propose improvements for future work.

1 Introduction

Traditionally sentiment analysis attempts to classify the polarity of a given text at the document, sentence, or feature/aspect level, i.e., whether the expressed opinion in the text is positive, negative, or neutral. More advanced sentiment classification looks at emotional states such as "Angry", "Sad", and "Happy".

Due to the increasing popularity of social media, over the past years sentiment analysis tasks in SemEval competitions have been mostly focused on twitter (Rosenthal et al., 2014) (Rosenthal et al., 2015; Nakov et al., 2016; Rosenthal et al., 2017). SemEval-2018 Task 1: Affect in Tweets (Mohammad et al., 2018) includes an array of subtasks on inferring the emotions (such as joy, fear, valence, and arousal) of a person from his/her tweet.

As we increasingly communicate using text messaging applications and digital agents, contextual emotion detection in text is gaining importance to provide emotionally aware responses to users. SemEval-2019 Task 3 (Chatterjee et al., 2019) introduces a task to detect contextual emotion in conversational text.

Deep-learning based approaches have recently dominated the state-of-the-art in sentiment analysis. However, a good performing model often requires large amounts of labeled data and takes many days to train. In computer vision, transfer learning has enabled deep learning practitioners to leverage models that have been pre-trained on ImageNet, MS-COCO, and other large datasets (Razavian et al., 2014; Shelhamer et al., 2017; He et al., 2016; Huang et al., 2017). Fine-tuning such pre-trained models in computer vision has been a far more common practice than training from scratch.

In Natural Language Processing (NLP), the most common and simple transfer learning technique is fine-tuning pre-trained word embeddings (Mikolov et al., 2013). These embeddings are used as the first layer of the model on the new dataset, and still require training from scratch with large amounts of labeled data to obtain good performance.

In 2018 several pre-trained language models (ULMFiT, OpenAI GPT and BERT) emerged. These models are trained on very large corpus, and enable robust transfer learning for fine-tuning NLP tasks with little labeled data.

In SemEval-2019 Task 3, we apply transfer learning approach using both pre-trained word embeddings and pre-trained language models. Our model achieves highly competitive result.

In this paper we describe our approach and experiments. The rest of the paper is laid out as follows: Section 2 provides an overview of the task, Section 3 describes the system architecture, and Section 4 reports results and performs an error analysis to obtain a better understanding of strengths and weaknesses of our approach and subsequently proposes improvements. Finally

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 220–224
Minneapolis, Minnesota, USA, June 6–7, 2019.

Label	Train	Dev	Test
Happy	4,243	142	284
Sad	5,463	125	250
Angry	5,506	150	298
Others	14,948	2,338	4,677
Total	30,160	2,755	5,509

Table 1: Train/Dev/Test set in SemEval2019 Task 3.

we conclude in Section 5 along with a discussion about future work.

2 Task Overview

2.1 Dataset

The organizers provide a training, development, and test set. Each row in the dataset is a 3-turn conversation between two people. The task is to classify the emotion of a conversation as "Happy", "Sad", "Angry", or "Others". Table 1 shows the distribution of the datasets across the labels. No other dataset is used in our experiments.

2.2 Evaluation Metric

Evaluation metric is micro-averaged F1 score for the three emotion classes i.e. Happy, Sad and Angry (excluding the class "Others"). This is referred as micro F1 score throughout the paper.

3 System Description

3.1 System Architecture

Figure 1 details the System Architecture. We now describe how all the different modules are tied together. The input raw text is pre-processed as described in Section 3.2. The processed text is passed through all the models described in Sections 3.3 to 3.8. Finally, the system returns the average of the predicted probabilities from all models as the output.

3.2 Pre-processing

The conversation text in the dataset is similar to tweets in that it may contain one or many emojis, and may have misspelled words. We use ekphrasis tool [1] to preprocess the data. The tool performs the following steps: tokenization, spell correction (i.e replace a misspelled word with the most probable candidate word), word normalization, and word segmentation. All words are lower-cased.

After the text in each turn is processed, we concatenate them with a separator "⟨eos⟩".

3.3 Fine-tuning ULMFiT

Universal Language Model Fine-tuning (ULMFiT) (Howard and Ruder, 2018) trains language models on Wikitext-103 (Merity et al., 2017b), which consists of 28,595 preprocessed Wikipedia articles and 103 million words. It's based on the language model AWD-LSTM (Merity et al., 2017a), a regular LSTM (with no attention, shortcut connections, or other sophisticated additions) with various tuned dropout hyperparameters. It provides two pre-trained models: a forward model trained from left to right, a backward model trained from right to left.

Furthermore, it introduced several novel techniques for transfer learning: discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing, to retain previous knowledge and avoid catastrophic forgetting during fine-tuning.

We first fine-tune the forward language model. We combine all data including training, dev and test set, and split into a training and validation set. We use fast.ai's lr_find [2] method to find the optimum learning rate, and use early stopping on validation loss to tune the dropout values from 0.7 to 2.5.

Then we fine-tune the classifier on the training set using 10-fold cross validation. We use early stopping on the evaluation metric of the task (micro F1 with "Others" class excluded). We experiment with dropout values from 0.7 to 0.85.

We repeat the same process for the backward language model.

3.4 Fine-tuning BERT

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) trains language models on BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia (2,500M words). It trains a deep bidirectional language models by masking some percentage of the input tokens at random, and then predicting only those masked token. This creates deep bidirectional representations by jointly conditioning on both left and right context in all layers.

In addition, it also trains a binarized next sentence prediction task which helps with understanding relation between two sentences, important for

[1] https://github.com/cbaziotis/ ekphrasis

[2] https://github.com/fastai/fastai

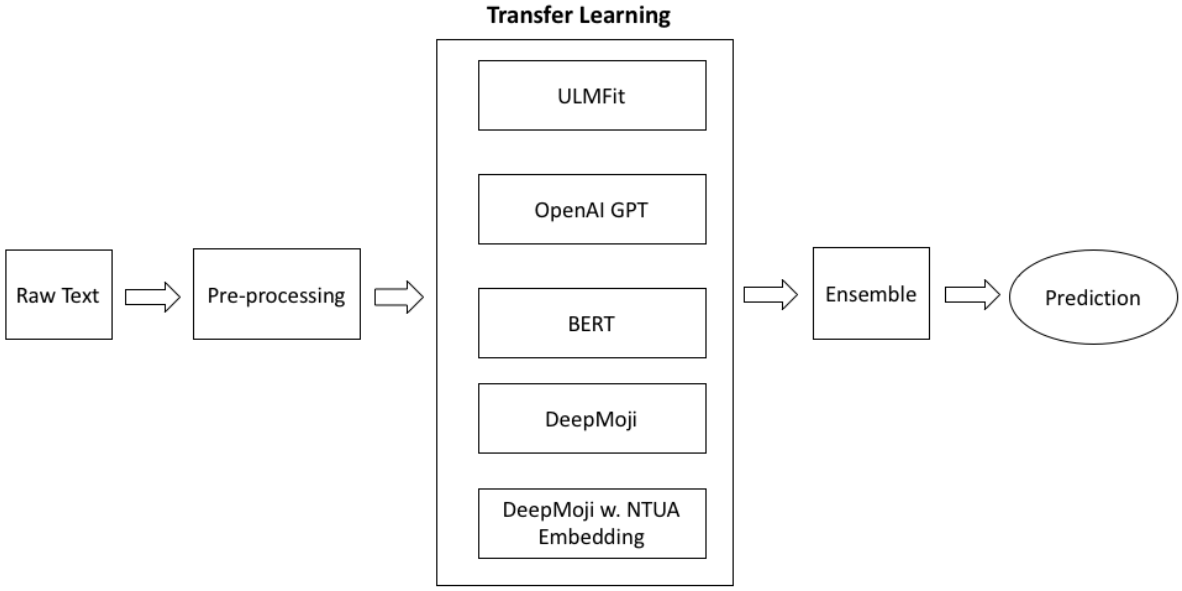

Figure 1: System Architecture.

Question Answering and Natural Language Inference tasks.

BERT provides pre-trained base and large models in multiple languages. In our experiments we use the large uncased English model. And we use the pytorch implementation by huggingface [3].

We experiment with fine-tuning the language model on a training and validation set split from a combined data set including training, dev and test set. We use early stopping on validation loss.

We then add a classifier layer on top of the output from the language model, and train it using the training set from the task with 10-fold cross validation. We use early stopping on the evaluation metric of the task (micro F1 with "Others" class excluded). We experiment with learning rate from 7e-6 to 3e-5.

3.5 Fine-tuning OpenAI GPT

OpenAI's Generative Pre-Training (GPT) (Alec et al., 2018) trains a language model using Transformer architecture on BooksCorpus. It obtains state-of-the-art result on many tasks including Natural Language Inference, Question answering and commonsense reasoning, Semantic Similarity, and Text Classification.

We tune the hyperparameters (clf_pdrop, embd_pdrop, resid_pdrop and attn_pdrop) in different combinations of values 0.1 and 0.2 (default value is 0.1) on the dev set. Due to that the dev score is less promising than the previous approaches, we do not use cross validation as it would take significant more time and compute resources. In fact this model was not included in the final submission.

[3] https://github.com/huggingface/pytorch-pretrained-BERT

3.6 Fine-tuning DeepMoji

DeepMoji (Felbo et al., 2017) performs distant supervision on a dataset of 1246 million tweets containing one of 64 common emojis. It obtained state-of-the-art performance on 8 benchmark datasets within sentiment, emotion and sarcasm detection using a single pretrained model.

We perform fine-tuning using the training set for training, and dev set as a validation set. We adopt the gradual unfreezing apporach (introduced by ULMFiT): first unfreeze the last layer and fine-tune all unfrozen layers for one epoch. We then unfreeze the next lower frozen layer and repeat, until we fine-tune all layers until convergence at the last iteration.

We do not use 10-fold cross validation due to that the highest micro F1 score on dev set does not seem promising.

3.7 Training a DeepMoji model with NTUA embedding

We also train a model from scratch using the DeepMoji's architecutre, but replace its embedding with a 310 dimensional embedding trained by NTUA-SLP team (Baziotis et al., 2018), which was trained on a dataset of 550M English twit-

Model	F1 (Dev)				F1 (Test)			
	Happy	Sad	Angry	Avg	Happy	Sad	Angry	Avg
ULMFiT Fwd	0.7138	0.8106	0.7593	0.7586	0.6901	0.7647	0.7535	0.7357
ULMFiT Bwd	0.7101	0.8077	0.752	0.7541	0.6993	0.7598	0.7387	0.7321
BERT	0.6585	0.7574	0.7403	0.7172	0.6289	0.7040	0.7345	0.6907
OpenAI GPT	0.6322	0.7481	0.7395	0.7050	0.6388	0.7279	0.7280	0.6976
DeepMoji	0.6195	0.7037	0.7435	0.6914	0.5933	0.6932	0.7190	0.6703
DeepMoji/NTUA	0.7066	0.7881	0.7011	0.7274	0.6997	0.7518	0.7048	0.7168
Combined (all)	0.7097	0.8077	0.7656	0.7585	0.7267	0.8023	0.7776	0.7680
Combined (no OpenAI)*	0.7285	0.8244	0.7761	**0.7742**	0.7153	0.7977	0.7713	0.7608
ULMFiT+BERT+OpenAI	0.7255	0.7658	0.8185	0.7619	0.7254	0.8031	0.7799	**0.7686**

Table 2: Micro Average F1 scores on dev set and test set. Bold indicates the highest F1 score on each dataset among the ensembled models. Asterisk indicates our final submission: ensemble of all models except OpenAI.

Turn 1	Turn 2	Turn 3	Label	ULMFiT	BERT
Lol i love it	Glad I made you laugh. LOL! That was awesome! 😄	cool	Happy	Others	Happy
Love you so much as always	Love you even more :)	Don't copy my stuff 😂	Happy	Happy	Others
I have got my friends	Dude, I almost had you..	😌	Sad	Sad	Others
I will not talk u	Why are you calling? That's rude. Text.	Bye😒	Sad	Angry	Angry
I don't smile for ur compliment	as you say so :)	😞	Sad	Sad	Happy
I don't need your idea	I just want to tell you all...	I didn't want to hear that	Angry	Others	Angry
Wow i didn't expect this frm u	no i didn't mean dere was sarcasm in it. ;)	I thought you meant it😡😡😡😡	Angry	Angry	Happy

Figure 2: Prediction examples by ULMFiT and BERT. Red indicates incorrect prediction.

ter messages. It was trained based on word2vec and has 310 dimensional embeddings, consisting of 300 dim word2vec embeddings and 10 dim affective dimensions.

We use the keras_lr_finder [4] method to find the optimum starting learning rate (with the fastest decrease in training loss), and train the model on the training set using 10-fold cross validation and early stopping on the evaluation metric of micro F1 score.

3.8 Ensembling

We combine the predictions of all models above by taking the unweighted average of the posterior probabilities for these models, and the final prediction is the class with the largest averaged probability.

4 Results and Analysis

Table 2 shows the results of various models on the dev set and test set. ULMFiT has the best performance on both dev and test sets, outperforming all other pre-trained models. The DeepMoji model trained from scratch with NTUA embedding ranks the second.

Figure 2 shows some examples where the ULMFiT or BERT makes incorrect predictions for the same conversations. We observe that BERT often makes incorrect predictions when emojis are present in the text, while ULMFiT is more robust to emojis. This suggests that the high performance of ULMFiT is due to not only the large corpus on which the language model is pre-trained on, but also the superior fine-tuning methods, such as discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing.

5 Conclusion

In this paper we describe our methods for contextual emotion detection. We achieved very competitive results in SemEval-2019 Task 3 using an ensemble of Transfer Learning models. We demonstrate that with sophisticated fine-tuning techniques in ULMFiT, transfer learning using pre-trained language models yields the highest performance, outperforming models trained from scratch. For future work we plan to explore these techniques with OpenAI GPT and BERT as well.

[4] https://github.com/surmenok/keras_lr_finder

References

Radford Alec, Narasimhan Karthik, Salimans Tim, and Ilya Sutskever Openai. 2018. Improving Language Understanding by Generative Pre-Training. Technical report.

Christos Baziotis, Nikos Athanasiou, Alexandra Chronopoulou, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, Shrikanth Narayanan, and Alexandros Potamianos. 2018. NTUA-SLP at SemEval-2018 Task 1: Predicting Affective Content in Tweets with Deep Attentive RNNs and Transfer Learning.

Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. 2019. Semeval-2019 task 3: Emocontext: Contextual emotion detection in text. In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval-2019)*, Minneapolis, Minnesota.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Technical report.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. Technical report.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339. Association for Computational Linguistics.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017a. Regularizing and Optimizing LSTM Language Models. Technical report.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017b. Pointer Sentinel Mixture Models.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Saif M Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 Task 1: Affect in Tweets. Technical report.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. SemEval-2016 Task 4: Sentiment Analysis in Twitter. Technical report.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 Task 4: Sentiment Analysis in Twitter. Technical report.

Sara Rosenthal, Saif M Mohammad, Preslav Nakov, Alan Ritter, Svetlana Kiritchenko, and Veselin Stoyanov Facebook. 2015. SemEval-2015 Task 10: Sentiment Analysis in Twitter. Technical report.

Sara Rosenthal, Preslav Nakov, Alan Ritter, and Veselin Stoyanov. 2014. SemEval-2014 Task 9: Sentiment Analysis in Twitter. Technical report.

Evan Shelhamer, Jonathan Long, and Trevor Darrell. 2017. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724.