Text-based Emotion Classification using different *n*-grams and their Combinations

Chih-Yi Lin **Yat Han Lai**

Abstract

Different *n*-grams and their combinations are proposed to consider word sequence to tackle the elusive nature of emotion expression, in contrast to the bag-of-words approach commonly used in text-based emotion classification. This approach is tested by comparing learning-based approaches, i.e. support vector machine (SVM) and convolutional neural network (CNN). Other than the generally superior performance of CNN, our experiments demonstrate that there is a consistent improvement in CNN in the case when higher-order *n*-grams (i.e. trigrams and 4-gram) and their combinations are used, suggesting that CNN filters can detect and activate unseen word sequences, hence leading to a better capacity for generalization, whereas SVM relies heavily on unigrams for prediction[1].

1 Introduction

As the use of higher *n*-grams that goes beyond unigrams and bigrams, in particular their combinations, are rarely tested in previous studies, we propose to compare support vector machine (SVM) and convolutional neural network (CNN) with features built upon different *n*-grams to investigate the best solution to utilize word sequences in text-based emotion classification.

2 Background

Text-based emotion classification aims to detect and recognize human emotions expressed in texts. Supervised machine learning algorithms have been widely implemented in text-based emotion classification, and deep learning approaches are recently being adopted because they are more robust and their deep layers are able to extract hidden details in text (cf. Acheampong et al., 2020). Traditional machine learning, features of which commonly built upon linguistic features, i.e. bag-of-words and TF-IDF, is often used as a baseline to compare with deep learning algorithms, and one most commonly used classifier is support vector machine (SVM) (Kratzwald et al., 2018). For example, Kratzwald et al. (2018) employed SVM as a baseline, features of which were built upon TF-IDF (that measures the relative importance of terms to a document), and reported an F-measure of 0.54. Consistent with their research, the experiments in this paper recruit SVM together with TF-IDF features as the baseline.

The argument for the superior results yielded by deep learning is that neural networks can iterate over a sequence of individual words of an arbitrary length, therefore the need of feature extraction can be circumvented, unlike traditional machine learning methods (Kratzwald et al., 2018). Johnson and Zhang (2014) experimented the Convolutional Neural Network (CNN) on high-dimensional text data to learn the embeddings, comparing with SVM up to 1+3-grams on sentiment classification task, and demonstrated that CNN can learn some language patterns which help better predict unseen data.

One major challenge of text-based emotion classification is that emotion expression is often elusive in nature, for example, given the prevalence of implicit expression of emotions and metaphors (Seyeditabari et al., 2018). To this the *n*-gram method is one potential solution because by maintaining word order, it is able to cover syntactic patterns and include critical information (Nandwani and Verma, 2021). Traditional machine learning methods in existing studies normally represent each document entirely with one *n*-gram and rarely go beyond bag-of-words representation (Seyeditabari et al., 2018), Tan et al. (2002) postulated that two-word phrase (i.e. bigrams) results in raising quality of feature sets in text classification. However, existing literature which explored the combination of *n*-grams is scarce, with some exceptions such as

[1] We are Group 12 of SS22 Team Lab. Please refer to `https://github.com/chihyi-lin/Emotion-Classification` for our Github repository.

Johnson and Zhang (2014), who reported a set of experiments, in which CNN reached higher accuracy by combining two parallel convolution layers. This paper aims to extend previous research like Johnson and Zhang (2014) by investigating how higher-order n-grams and their combinations can facilitate the prediction of emotion expression.

3 Methodology

This section presents the feature extraction methods and classification models used in the experiments.

3.1 Feature extraction

3.1.1 *n*-grams

n is an integer that represents the number of words in a sequence. Adding features of higher n-grams might facilitate detecting multi-word expression in text (Shaaban et al., 2021).

3.1.2 TF-IDF

TF-IDF is formulated by term frequency × inverse document frequency, to create a composite weight for each term in each of the documents (cf. Rahman et al., 2020):

- Term Frequency (TF): the measurement of a term continually occurs in one document
- TF(t) = (the number of times of one term appearing in one document)/(total number of terms in the same document)
- Inverse Document Frequency (IDF) :measurement of how important a term is
- IDF(t) = log_e(total number of documents/number of documents with term t in it)

$$\text{TF-IDF} = \text{TF(t)} \times \text{IDF(t)} \tag{1}$$

3.1.3 Pre-trained GloVe Embeddings

GloVe stands for global vectors for word representation (Pennington et al., 2014). Its word vectors is developed by Stanford by training on aggregating global word-word co-occurrence from a corpus and covering 300 dimensional vectors for a vocabulary. Word embeddings are learnt in such a way that words with similar semantics will be positioned nearby in the vector space and therefore can better express similarity between words. The embeddings applied in the experiments are Glove-6B embeddings (Wikipedia 2014 + Gigaword 5) with 100 dimensions, which covers 400K vocabulary. Words not present in the pre-trained embeddings are mapped to <OOV> (out-of-vocabulary) token, represented by a zero vector.

3.2 Model

3.2.1 SVM

Despite the high dimensionality of text data, support vector machines (SVMs) can learn regardless of the dimensionality of the feature space (Deng et al., 2019). SVMs can also model non-linear relationships because they are able to take all features of text data into account and work well with sparse document vectors, a characteristics of text classification (Kratzwald et al., 2018; Purgstaller, 2019). The SVM classifier of the experiments is performed by a linear kernel function, following previous studies comparing SVM and CNN (e.g. Johnson and Zhang, 2014; Wang and Qu, 2017), and the regularization parameter is set to 10 using Python library *scikit-learn* (Pedregosa et al., 2011).

3.2.2 CNN

Two architectures for convolutional neural network (CNN) are employed: single channel and multiple channels. The former is for the same order of n-gram, while the latter is for a combination of different orders of n-grams. To experiment various orders of n-grams in CNN, we specify filter sizes equal to those n we would like to test on. The input sequences are processed by three modules: a word embedding module, a convolutional module, and a fully connected sigmoid module. For the multiple channels architecture, there are multiple parallel word embedding modules and convolutional modules, and the produced feature maps will further be concatenated and passed to the fully connected module (Kim, 2014).

The first module contains an embedding layer, receiving input sequences padded with the same length, 180, which is the maximum length of sentence in the dataset. All word vectors obtained from GloVe are kept static throughout training, and only other parameters are learned. The convolutional module has a convolution layer with 128 filters to obtain features, and is activated with a ReLU function. Then the most important features are chosen in the global max pooling layer, and are passed to the fully connected module, which contains a dense layer with 128 neurons, activated with ReLU function, followed by a dropout layer with 0.5 dropout for regularization, and a final dense layer is activated with sigmoid function.

Training is done with categorical cross-entropy as loss function and Adam optimizer, with mini-batch size of 50. Early stopping is applied once the validation loss starts to increase to avoid overfitting. On average, the models are trained between 11-12 epochs in our experiments.

4 Experiments

4.1 Experiment settings

4.1.1 Dataset

All experiments use the International Survey on Emotion Antecedents and Reactions (ISEAR) dataset, in which seven primary emotions, namely joy, fear, anger, sadness, disgust, shame and guilt were reported. After removing invalid data, there are 5245 instances in the training set, 1127 instances in the validation set, and 1126 instances in the test set. The training set is only used for training models, the validation set is used for hyperparameters tuning and the test set is used for testing the trained models and generating the results of the experiments.

4.1.2 Preprocessing

All texts are converted into lowercase and tokenized without stemming[2].

4.1.3 Feature extraction

In both learning-based approaches, features are extracted using different phrases (i.e. *n*-grams), consisting of 1-gram (unigrams), 3-gram(trigrams), 1+2-gram (joining unigrams and bigrams), 3+4-gram (joining trigrams and 4-gram), 1+2+3-gram (joining unigrams, bigrams and trigrams). In terms of word representation, the features of SVM are built upon TF-IDF whereas CNN's features are built upon pre-trained GloVe embeddings.

4.2 Results & Discussion

The resulting performance of our experiments is listed in Table 1, using the macro-averaged F1-scores. The F1-score is formulated by the harmonic mean of precision and recall, i.e.:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2}$$

Preprocess	Features	SVM	CNN
Tokenization	1-gram	0.54	0.56
	3-gram	0.42	0.58
	1+2-gram	0.55	0.58
	3+4-gram	0.41	0.58
	1+2+3-gram	**0.56**	**0.61**

Table 1: Experiment results are evaluated with the macro-averaged F1-scores. The highest F1-scores for both models are in bold.

Comparison of different *n*-grams and their combinations between SVM and CNN

Table 1 shows that results of SVM obtained from using 1+2-gram and 1+2+3 gram are compatible with 1-gram, while 1+2+3-gram shows the best performance with a F-measure of 0.56 among the three. The results suggest a tendency for performance to slightly improve when more *n*-grams are combined with 1-gram; nonetheless, this improvement is not significant when the accuracy of the three are compared (1-gram: 54.4%; 1+2-gram: 55.14%; 1+2+3-gram: 55.57%)(cf. the comparison in Bekkerman and Allan (2004)). The performance using 3-gram, however, shows a tendency that combining more *n*-grams might not improve the performance, i.e. a 0.01 decrease in F-measure comparing 3-gram and 3+4-gram. Going beyond Tan et al. (2002)'s argument that two-word phrase (i.e. bigrams) would raise the quality of feature sets in using traditional machine learning methods, our results show that this may not be applicable to trigrams using SVM.

The results of CNN reveal that 1+2+3-gram perform the best in comparison with other *n*-grams and their combinations. In terms of window size, larger size is not necessary to perform better. 3-gram performs 0.02 better than 1-gram, yet 3+4-gram reach the same score as 1+2-gram. This could be due to small window sizes (i.e., 1 or 2) holding detailed information which may contribute to the prediction (Soni et al., 2022). Another noticeable result is that more combinations of *n*-grams perform better than less combinations. An increase of 0.02 in F1-score is observed comparing 1-gram with 1+2-gram, while there is a 0.03 increase from 1+2-gram to 1+2+3-gram.

Comparing the classification performance of SVM and CNN, CNN performs uniformly superior to SVM across all settings of using different *n*-grams and their combinations. The results of

[2] We tested in addition the performance of the two algorithms by removing punctuation and stopwords, whose results are, however, consistently worse across all conditions of feature extractions for SVM and CNN. We decided to report in this paper only the results obtained from performing only tokenization.

SVM are probably in line with what Johnson and Zhang (2014) suggest: SVM still heavily counts on unigram and cannot benefit from higher-order n-grams, which means only the n-grams that appeared in the training data can be used in prediction. Conversely, the results of CNN show a more consistent picture that higher-order n-grams could contribute to accurate prediction even if they did not appear in the training data, as long as some of their constituent words did. This will be discussed in further detail in the next section.

Comparison between 1-gram and 3+4-gram

A further comparison between the two learning-based models is performed by comparing their results using 1-gram and 3+4-gram, because they show an opposing trend: while CNN performs the worst using only 1-gram, it is the approximate best performance of SVM; while SVM performs the worst combining 3+4-gram, CNN has the approximate best performance.

Comparing the confusion matrices, *joy* and *fear* are the most accurately predicted emotions for SVM using 1-gram and CNN using both 1-gram and 3+4-gram. *Digust* is, however, the most accurately predicted emotion for SVM using 3+4-gram. *Anger* and *shame* are the most misclassfied for both classifier models: using 1-gram, *shame* is the most misclassified emotion for SVM with an error rate of 56% whereas it is *anger* for CNN with an error rate of 57%; in the case of 3+4-gram, both classifiers show a higher error rate, with *anger* as the most classified emotion for SVM (72%) and *shame* for CNN (66%).

A further analysis is performed by investigating the misclassified instances by using one of the n-gram features but correctly classified by another:

- *When we were in high school, a few guys sometimes provoked a friend of mine...Once one of the girls...said something nasty and this made me terribly angry.* (Emotion label: *anger*)
- *When my boyfriend was leaving...one night, I had a very deep sense of uneasiness and... that I wouldn't see him again.* (Emotion label: *fear*)

The listed examples further illustrate the opposite trend of the two models: using 1-gram, they are correctly classified by SVM but misclassified by CNN; however when 3+4-gram are combined, they are in turn misclassified by SVM but correctly predicted by CNN.

By investigating instances like the ones listed above, it seems that the unigram representation suffices for SVM to obtain correct predictions; however, these correct predictions are lost when the model's features are built upon higher-order n-grams (i.e. 3+4-gram). This might be an additional proof that SVM relies more on unigrams. CNN, conversely, using 3+4-gram manages to predict the above examples correctly, which are misclassified using 1-gram, even though the triggering word phrases underlined did not appear in the training set. This means that CNN using 3+4-gram can benefit from these constituent words (even if they were only partially seen in the training set) by taking their embedded features which have large values in the heavily-weight predictive component (Johnson and Zhang, 2014).

Regarding the elusive nature of emotion expressions, CNN shows to be more promising in using information provided by word orders, especially the long phrases (i.e. 3+4-gram) by showing superior performance to SVM.

5 Conclusion and future work

Our experiments show that CNN generally shows superior performance to SVM in text-based emotion classification. While the two models show compatible performance in using unigrams, CNN presents more effective use of higher-order n-grams, i.e. trigrams and their combination with 4-gram, than SVM, suggesting that the parallel CNN framework can learn and combing several types of embeddings, whereas SVM relies more heavily on unigrams for prediction.

We propose exploring CNN models with a deep architecture that go beyond the current parallel architecture, i.e. multiple convolutional layers whose features are built upon n-grams of even higher order than trigrams and 4-gram. Other possible future work might involve the combination of traditional machine learning and deep learning as it seems to be a promising approach that shows even superior performance to using, for example, SVM and CNN individually (cf. Wang and Qu, 2017; Wu et al., 2018), and it is tested in other languages such as Chinese. As pointed out by Eilertsen et al. (2019), a majority of existing research in emotion classification solely uses English as their target language, little is known whether existing methods are applicable to other languages. Recruiting more languages should prove fruitful to test the applicability of upcoming newer approaches in text-based emotion classification.

References

Francisca A. Acheampong, Wenyu Chen, and Henry Nunoo-Mensah. 2020. Text-based emotion detection: Advances, challenges, and opportunities. *Engineering Reports*, 2(e12189).

Ron Bekkerman and James Allan. 2004. Using bigrams in text categorization. *Technical Report IR-408*, Center of Intelligent Information Retrieval(UMass Amherst).

Xuelian Deng, Yuqing Li, Jian Weng, and Jilian Zhang. 2019. Feature selection for text classification: A review. *Multimedia Tools and Applications*, 78:3797–3816.

Alexander C. Eilertsen, Dennis Højbjerg Rose, Peter Langballe Erichsen, Rasmus Engesgaard Christensen, and Rudra Pratap Deb Nath. 2019. Languages' impact on emotional classification methods. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 277–286. IEEE.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Bernhard Kratzwald, Suzana Ilić, Mathias Kraus, Stefan Feuerriegel, and Helmut Prendinger. 2018. Deep learning for affective computing: Text-based emotion recognition in decision support. *Decision Support Systems*, 115:24–35.

Pansy Nandwani and Rupali Verma. 2021. A review on sentiment analysis and emotion detection from text. *Social Network Analysis and Mining*, 11(81).

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Roman Purgstaller. 2019. Dynamic *N*-gram based feature selection for text classification. Master's thesis, Graz University of Technology.

Sheikh Shah Mohammad Motiur Rahman, Khalid Been Md. Badruzzaman Biplob, Md. Habibur Rahman, Kaushik Sarker, and Takia Islam. 2020. An investigation and evaluation of n-gram, tf-idf and ensemble methods in sentiment classification. In *International Conference on Cyber Security and Computer Science*, pages 391–402.

Armin Seyeditabari, Narges Tabari, and Wlodek Zadrozny. 2018. Emotion detection in text: a review. arXiv preprint arXiv:1806.00674.

Yasmin Shaaban, Hoda Korashy, and Walaa Medhat. 2021. Emotion detection using deep learning. In *16th International Conference on Computer Engineering and Systems (ICCES)*, pages 1–10. IEEE.

Sanskar Soni, Satyendra Singh Chouhan, and Santosh Singh Rathore. 2022. Textconvonet: A convolutional neural network based architecture for text classification. arXiv preprint arXiv:2203.05173.

Chade-Meng Tan, Yuan-Fang Wang, and Chan-Do Lee. 2002. The use of bigrams to enhance text categorization. *Information processing management*, 38(4):529–546.

Zhiquan Wang and Zhiyi Qu. 2017. Research on web text classification algorithm based on improved cnn and svm. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pages 1958–1961. IEEE.

Huaiguang Wu, Daiyi Li, and Ming Cheng. 2018. Chinese text classification based on character-level cnn and svm. In Hu Peng, Changshou Deng, Zhijian Wu, and Yong Liu, editors, *Computational Intelligence and Intelligent Systems*, pages 227–238. Springer, Singapore.

Contributions

Report Writing

Abstract & (1) Introduction: Yat Han Lai
(2) Background: Yat Han Lai
(3) Methodology: Chih-Yi Lin (CNN part), Yat Han Lai (SVM part)
(4.1) Experiments: Chih-Yi Lin (CNN part), Yat Han Lai (SVM part)
(4.2) Experiment results and discussion: Chih-Yi Lin, Yat Han Lai
(5) Conclusion & Future work: Chih-Yi Lin, Yat Han Lai
Development of research question: Chih-Yi Lin
Experimental design: Chih-Yi Lin, Yat Han Lai
Proof-reading: Yat Han Lai

Code Implementation

SVM: Yat Han Lai

CNN: Chih-Yi Lin

Perceptron baseline: Chih-Yi Lin

Perceptron baseline - evaluation: Chih-Yi Lin, Yat Han Lai

Readme file writing and project management on Github: Chih-Yi Lin