KDE-AFFECT at SemEval-2018 Task 1: Estimation of Affects in Tweet by Using Convolutional Neural Network for n-gram

Shinnosuke Himeno and Masaki Aono
Department of Computer Science and Engineering
Toyohashi University of Technology
himeno@kde.cs.tut.ac.jp, aono@tut.jp

Abstract

This paper describes our approach to SemEval-2018 Task1: Estimation of Affects in Tweet for 1a and 2a. Our team KDE-AFFECT employs several methods including one-dimensional Convolutional Neural Network for n-grams, together with word embedding and other preprocessing such as vocabulary unification and Emoji conversions into four emotional words.

1 Introduction

With the rapid spread of SNS services (e.g. Twitter, Facebook, Instagram), massive user opinions have accumulated on the Internet. Among such opinions, it has been observed that not a few SNS contents naturally entail the affects (including joy, anger, sadness, fear) within themselves. Hence, the need to accurately detect the affects is increasing year by year.

In SemEval-2018 Task 1: Estimation of Affects in Tweet, we have attempted to extend our horizon from positive, neutral, and negative polarity estimations in former SemEval sentiment analysis in tweet having been held till 2017, to multiple emotions (joy, anger, sadness, and fear) in terms of regression (Task-1 1a) and classification (Task-1 2a). In doing so, we have adopted a standard one-dimensional Convolutional Neural Network (CNN), which is believed to be effective for text polarity estimation, where the kernel window size for 1D convolution is analogous to the concept of word n-gram. In addition, as most people have noticed, a tweet has potentially many Emojis to express emotions. In the following, we first briefly survey related work on tweet sentiment analysis including emotion estimation. Then, we describe our system, followed by showing the results returned from the organizer, and finally concluding our paper.

2 Related Work

Sentiment analysis of tweets has been studied by many researchers from the standpoint of classifying a tweet into either positive or negative polarity, and classifying it into multiple emotions (Giachanou and Crestani, 2016; Silva et al., 2016). A supervised approach to polarity classification of a tweet was proposed by Go et al. (2009). They employed Naive Bayes, Maximum Entropy, Support Vector Machine, and several other machine learning methods for their supervised learning. Bravo-Marque et al. (2013) presented an approach using multiple emotion dictionaries, while Saif et al. (2016) employed co-occurrence information of words. Severyn et al. (2015) introduced a deep learning approach. Lu et al. (2013) proposed a deep learning method suited for short texts. In SemEval, since 2014, sentiment analysis tasks using Twitter have been officially conducted, where a variety of methods have been tested (Hagen et al., 2015; Giorgis et al., 2016; Deriu et al., 2016; Rouvier and Favre, 2016; Xu et al., 2016). In SemEval2017 Rosenthal et al. (2017), Cliche et al. (2017) and Hamdan et al. (2017) presented methods for combining multiple Convolutional Neural Networks (CNNs) and multiple Long Short-Term Memories (LSTMs). Mohammad (2017) published an open dictionary of emotion scores for each word. Mohammad et al. published a dataset for estimating emotion intensities (Mohammad and Bravo-Marquez, 2017).

3 Methodology

In this section, we focus on our methods and ideas employed in this task. The fundamental idea of our method is based on the observation that "n-grams" seem to have vital effects to represent the emotion of a tweet, where "n-gram" denotes n consecutive words (instead of n consecutive char-

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 156–161
New Orleans, Louisiana, June 5–6, 2018.

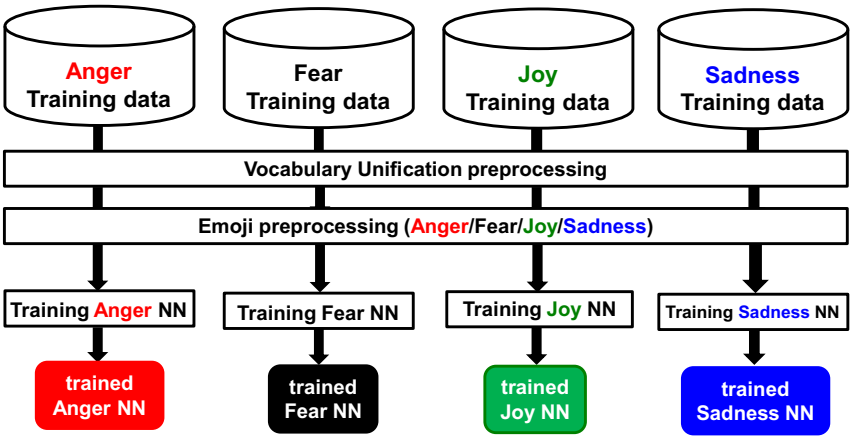

Figure 1: Overall Training Flow of KDE-AFFECT

acters). For instance, if the tweet sentence is "At last I made it.", 2-gram includes (At, last) and (made, it). Similarly, 3-gram includes (At, last, I) and (I, made, it).

We have adopted the method based on the n-gram convolution proposed by Kim (2014). Here, we prepare a matrix corresponding to a sentence representing an n-gram convolution in which this filtering process is carried out by the unit of an n-gram.

The overview of our system is as follows: First we apply preprocessing with "vocabulary unification" including lower case conversion, URL unification, two or more consecutive character squeezing, and hashtag elimination. Second, we apply Emoji conversion into four emotional words, which will be elaborated later. From Emoji conversion, we train the model independently for each emotion. Finally, we predict the emotion score for an unknown tweet by using the trained model. The overall system flow is shown in Figure 1.

3.1 Preprocessing with Vocabulary Unification

This step is applied to all emotions. It consists of the following processing:

- lower case conversion
- conversion of every instance of a URL string in a tweet to "<URL>"
- collapse of two or more consecutive letters into two
- elimination of hash sign (#)

It should be noted that by a url string we mean a regular expression starting with either "http", "https", "ftp", or "www". Any url string is converted to <URL>. For example, "I want to be happy on http://t.co/S6moxr1U" is converted to "I want to be happy on <URL>".

3.2 Preprocessing for Emoji

From our observation of real tweets, approximately more than 20% of them have some kind of Emojis. Emotions are naturally represented by many different Emojis. Hence, we introduce the conversion of possible emotions represented by an Emoji into each emotional word. Please note that Emoji preprocessing is applied to all Emoji data, regardless of emotions. For instance, each anger Emoji might appear not only in an Anger dataset, but in Fear, Joy, and Sadness datasets as well. This is why we have decided to apply the Emoji conversion despite the differences of emotions. In the following, we present Emoji for each emotion, where Emoji has been taken from a Full Emoji Web site[1]. The selection of Emoji has been made by using the labels (such as "face-positive") annotated to the above Web sites.

3.2.1 Anger Emoji

The Anger Emojis we selected are shown in Figure 2. All of them are replaced by "anger".

Figure 2: Anger Emoji

3.2.2 Fear Emoji

The Fear Emojis we selected are shown in Fig. 3. All of them are replaced by "fear".

[1]https://unicode.org/emoji/charts/full-emoji-list.html

Figure 3: Fear Emoji

3.2.3 Joy Emoji

The Joy Emojis we selected are shown in Fig. 4. All of them are replaced by "joy".

Figure 4: Joy Emoji

3.2.4 Sadness Emoji

The Sadness Emojis we selected are shown in Fig. 5. All of them are replaced by "sadness".

Figure 5: Sadness Emoji

3.3 Convolutional Neural Network for n-gram

Once preprocessing is done, we have a kind of rectified tweet, represented by a matrix. Figure 6 illustrates a word-by-word matrix representation of a rectified tweet. Here we take a matrix of 80 by 300, where 80 is the maximum number of words per tweet, and 300 corresponds to our embedding vector size. If a tweet has less than 80 words, zero padding is performed to fill the input matrix.

3.3.1 Embedding

In Embedding, each tweet is converted to a matrix. Specifically, we first divide a tweet into words using a whitespace, thereby treating a special character (one of ".h, ",h, "!h, and "?h) as a separate word. Second, we transform each word into its distributed representation of 300 dimensions using Word2Vec (Mikolov et al., 2013a,b). The training of Word2Vec itself is done by using approximately 470 million tweets after the processing described in 3.1. We finally obtain the embedding by padding zero values to a fixed size of a 80 by 300 dimensional matrix.

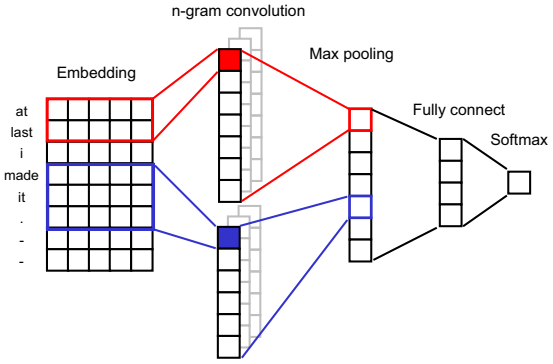

Figure 6: Our n-gram Convolution-based Approach

3.3.2 n-gram Convolution Layer

In an n-gram convolutional layer, we perform convolution, and generate a length $m - n + 1$ vector, where m denotes the maximum word length (here 80). This is straightforward, since both ends are trimmed during the n-gram convolution stage shown in Figure 6. For instance, if 3-gram is concerned, the length in our implementation will be 80-3+1 = 78. Note that we have multiple n-gram convolutional layers for each emotion. "Joy" neural network architecture, for example, has 2-gram, 3-gram, 4-gram, and 5-gram convolutional layers, which will be discussed later in Table 3.

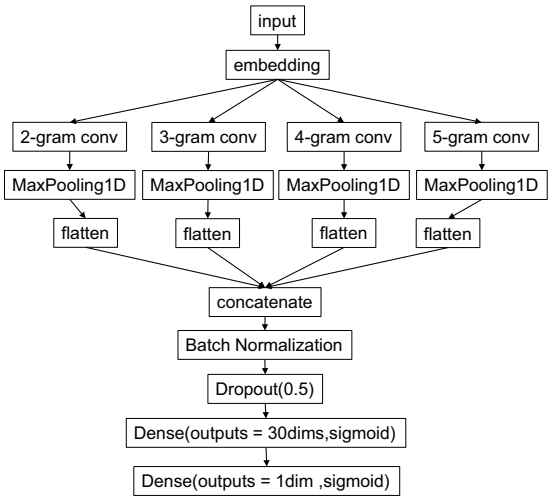

Figure 7: KDE-AFFECT system's DNN architecture

3.3.3 Max Pooling Layer

In a Max Pooling layer, from each n-gram convolutional layer, the maximum value is computed,

Intensity range	Intensity amount
$[0.0, 0.35)$	0 (no E)
$[0.35, 0.5)$	1 (low amount of E)
$[0.5, 0.65)$	2 (moderate amount of E)
$[0.65, 1.0]$	3 (high amount of E)

Table 1: Inferred Intensity Level

	Anger	Fear	Joy	Sadness
Training	1701	2252	1616	1532
Dev.	388	389	290	397
Test (1a)	17940	17924	18043	17913
Test (2a)	1002	986	1105	975

Table 2: SemEval 2018: Task 1 dataset (1a and 2a)

and a vector of the length equal to the number of filters is generated. In our system, the output of four multiple n-gram convolutional layers are flattened and concatenated in the subsequent layers. The output dimension is the number of filters multiplied by the number of n-gram convolutional layers.

3.3.4 Fully-connected Layers

In our system, we have two fully-connected layers, where the first hidden fully-connected layer accepts the input from the concatenation layer conneted from multiple max pooling layers. Empirically, we set 30 outputs for the first fully-connected layer. The second layer outputs either the estimated intensity value of an emotion (Task 1a) or the estimated intensity level (Task 2a). The way to estimate the intensity level (Task 2a) is elaborated in the next section.

3.4 Estimating Intensity Level (Task 2a)

For Task 2a, we need to estimate the intensity level. Specifically, participants are required to classify the emotional intensities into four levels; high amount, moderate amount, low amount, and nothing. Our strategy for the amount of emotional intensity amount level is simple, which is based on the inferred intensity range as shown in Table 1. In Table 1, the left column denotes the range of emotional amount that we have defined for this task. For example, $[0.0, 0.35)$ means that the left boundary 0.0 is inclusive, while the right boundary 0.35 is exclusive in the range of the amount of emotion.

4 Experiments

Here we describe the experimental environment and our evaluation results.

4.1 dataset

All participants are given SemEval 2018: Task 1 Affect in Tweets (AIT) (Mohammad et al., 2018) dataset. The details are shown in Table 2.

4.2 Evaluation Measure

Here, the evaluation measure for a model is correlation coefficient r. Given variables x and y, where x corresponds to a predicted emotion value and y to a true emotion value, and their associated sample variances S_x, S_y, and the covariance S_{xy} are represented by the following equation:

$$r = \frac{S_{xy}}{S_x S_y}$$

4.3 Experiment Environment

For our deep learning program for the task, we used the following list of hyper-parameters:

loss function: Root Mean Square Error (RMSE)
filter number: $200 \times n$
epochs: 30
dropout rate: 0.5
optimizer: Adam
batch size: 64

The framework we use is Keras with backend Tensorflow. In our Ubuntu server, it took approximately 1 second for each epoch.

4.4 Preliminary Experiments for n-gram Convolutions

For each emotion, our system attempts to find an empirical optimal combination of n-gram convolutions. Table 3 summarizes the results of preliminary experiments for this purpose. Here, $r(A)$ denotes the correlation coefficient for Anger. Similarly, $r(F)$ for Fear, $r(J)$ for Joy, and $r(S)$ for Sadness. From the table, we decided as follows: For Anger, we chose [1,2,3,4,5,6] (meaning we took the combination of 1-gram, 2-gram, 3-gram, 4-gram, 5-gram, and 6-gram convolutions). For Fear, we chose [2,3,4,5,6]. For Joy and Sadness, we chose [2,3,4,5].

4.5 Experimental Result (Task 1a)

According to the Official Leaderboard for Task 1a, our team KDE-AFFECT turned out to be 30-th. If

n	$r(A)$	$r(F)$	$r(J)$	$r(S)$
$[1, 2, 3, 4, 5, 6]$	**0.5529**	0.5919	0.5771	0.6349
$[2, 3, 4, 5, 6]$	0.5518	**0.5994**	0.5464	0.6173
$[1, 2, 3, 4, 5]$	0.5381	0.5948	0.5730	0.6078
$[2, 3, 4, 5]$	0.5309	0.5736	**0.5906**	**0.6360**

Table 3: Preliminary experiments for n-gram convolustions

Team	avg-r	$r(A)$	$r(F)$	$r(J)$	$r(S)$
KDE-AFFECT	0.620	0.630	0.621	0.598	0.630
SeerNet^{1st}	0.799	0.827	0.799	0.792	0.798
NTUA-SLP^{2nd}	0.776	0.782	0.758	0.771	0.792
PlusEmo2Vec^{3rd}	0.766	0.811	0.728	0.773	0.753
CrystalFeel^{14th}	0.717	0.740	0.700	0.708	0.720
EliRF-UPV^{15th}	0.696	0.705	0.686	0.693	0.700
iit_delhi^{29th}	0.621	0.633	0.645	0.618	0.588
DeepMiner^{31th}	0.575	0.581	0.570	0.575	0.573
Baseline^{37th}	0.520	0.526	0.525	0.575	0.453

Table 4: Our result with selected other teams for Task 1a

Team	avg-r	$r(A)$	$r(F)$	$r(J)$	$r(S)$
KDE-AFFECT	0.530	0.530	0.470	0.552	0.567
SeerNet^{1st}	0.695	0.706	0.637	0.720	0.717
PlusEmo2Vec^{2nd}	0.659	0.704	0.528	0.720	0.683
psyML^{3rd}	0.653	0.670	0.588	0.686	0.667
UNCC^{9th}	0.599	0.604	0.544	0.638	0.610
ECNU^{16th}	0.531	0.565	0.441	0.581	0.536
CrystalFeel^{18th}	0.530	0.576	0.466	0.540	0.538
Baseline^{26th}	0.394	0.382	0.355	0.496	0.370

Table 5: Our result with selected other teams for Task 2a

we use similar notations as in Table 3, and pick up the top-3 ranked teams, as well as randomly chosen teams CrystalFeel (14-th place), ELipRF-UPV (15-th place), iit_delhi (29-th), DeepMiner (31-th), and the baseline (37-th), the result looks like Table 4.

4.6 Experimental Result (Task 2a)

According to the Official Leaderboard for Task 2a, our team KDE-AFFECT turned out to be 17-th. If we use similar notations as in Table 3, and pick up top-3 ranked teams, as well as randomly chosen teams UNCC (9-th place), ECNU (16-th place), CrystalFeel (14-th place), and the baseline (26-th), the result looks like Table 5.

5 Conclusion

This paper describes the approach we took for SemEval-2018 Task 1: Affect in Tweets (subtasks 1a and 2a). We have chosen a combination of different n-gram convolutions with preprocessing including vocabulary unification and Emoji conversion.

Acknowledgments

Part of this research is supported by MEXT KAKENHI, Grant-in-Aid for Scientific Research (B), Grant Number 17H01746.

References

Felipe Bravo-Marquez, Marcelo Mendoza, and Barbara Poblete. 2013. Combining strengths, emotions and polarities for boosting twitter sentiment analysis. In *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, WISDOM '13, pages 2:1–2:9, Chicago, Illinois.

Mathieu Cliche. 2017. Bb_twtr at semeval-2017 task 4: Twitter sentiment analysis with cnns and lstms. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 573–580, Vancouver, Canada. Association for Computational Linguistics.

Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. 2016. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1124–1128, San Diego, California. Association for Computational Linguistics.

Anastasia Giachanou and Fabio Crestani. 2016. Like it or not: A survey of twitter sentiment analysis methods. *ACM Computing Surveys*, 49(2):28:1–28:41.

Stavros Giorgis, Apostolos Rousas, John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2016. aueb.twitter.sentiment at semeval-2016 task 4: A weighted ensemble of svms for twitter sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 96–99, San Diego, California. Association for Computational Linguistics.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6.

Matthias Hagen, Martin Potthast, Michel Büchner, and Benno Stein. 2015. Webis: An ensemble for twitter sentiment detection. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 582–589, Denver, Colorado. Association for Computational Linguistics.

Hussam Hamdan. 2017. Senti17 at semeval-2017 task 4: Ten convolutional neural network voters for tweet polarity classification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 700–703, Vancouver, Canada. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems 26*, pages 1367–1375. Curran Associates, Inc.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Saif Mohammad and Felipe Bravo-Marquez. 2017. Emotion intensities in tweets. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 65–77. Association for Computational Linguistics.

Saif M. Mohammad. 2017. Word affect intensities. *CoRR*, abs/1704.08798.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518. Association for Computational Linguistics.

Mickael Rouvier and Benoit Favre. 2016. Sensei-lif at semeval-2016 task 4: Polarity embedding fusion for robust sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 202–208, San Diego, California. Association for Computational Linguistics.

Hassan Saif, Yulan He, Miriam Fernandez, and Harith Alani. 2016. Contextual semantics for sentiment analysis of twitter. *Information Processing and Management*, 52(1):5–19.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 959–962, Santiago, Chile. ACM.

Nadia Felix F. Da Silva, Luiz F. S. Coletta, and Eduardo R. Hruschka. 2016. A survey and comparative study of tweet sentiment analysis via semi-supervised learning. *ACM Comput. Surv.*, 49(1):15:1–15:26.

Steven Xu, HuiZhi Liang, and Timothy Baldwin. 2016. Unimelb at semeval-2016 tasks 4a and 4b: An ensemble of neural networks and a word2vec based model for sentiment classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 183–189, San Diego, California. Association for Computational Linguistics.