DMCB at SemEval-2018 Task 1: Transfer Learning of Sentiment Classification Using Group LSTM for Emotion Intensity prediction

Youngmin Kim, Hyunju Lee
Gwangju Institute of Science and Technology,
Data mining and Computational Biology Lab, Gwangju, Korea
{minok00, hyunjulee}@gist.ac.kr

Abstract

This paper describes a system attended in the SemEval-2018 Task 1 "Affect in tweets" that predicts emotional intensities. We use Group LSTM with an attention model and transfer learning with sentiment classification data as a source data (SemEval 2017 Task 4a). A transfer model structure consists of a source domain and a target domain. Additionally, we try a new dropout that is applied to LSTMs in the Group LSTM. Our system ranked 8th at the subtask 1a (emotion intensity regression). We also show various results with different architectures in the source, target and transfer models.

1 Introduction

Sentiment analysis is one of the most famous Natural Language Process (NLP) task. In this study, we perform a task that predicts emotional intensities of anger, joy, fear and sadness with tweet messages, where intensity values range from 0 to 1. This task is competed at SemEval-2018 Task 1 (Mohammad et al., 2018). In previous studies, neural networks with word embedding and affective lexicons were widely used (Goel et al., 2017; He et al., 2017). Also, many studies employed support vector regression (Duppada and Hiray, 2017; Akhtar et al., 2017).

Transfer learning was recently proposed as an effecive approach to have higher performance, when data is not abundant. Using a pre-trained deep-learning model with an abundant data set has been popular and shows good results in various tasks (Donahue et al., 2014; Conneau et al., 2017). Especially in a medical image task, it is very efficient because of lacks of medical data (Tajbakhsh et al., 2016). Just as humans can learn new things better with their past knowledge, neural networks can also be trained on target domains by transferring knowledge from the source domain.

We make a transfer model that can be divided into a source model and a target model. The source model is constructed based on the paper (Baziotis et al., 2017). The model of this paper uses LSTM with attention. However, we introduce Group LSTM (GLSTM) (Kuchaiev and Ginsburg, 2017) with a new dropout. After then, we make the target model with LSTM.

In the result section, we provide comparison of LSTM and GLSTM in the source model, and results of various pre-trained word embeddings with target model. Finally, we discuss about the result of the transfer model that is a combined model with the source and target models.

2 System Description

2.1 Data and Label

For transfer learning, we use a source data provided by SemEval 2017 Task4 (a) (Rosenthal et al., 2017). The task of the source domain is to classify sentences to positive, negative and neutral sentences. Training data is 44,613 sentences (10% are used as a development set), and test data is 12,284 sentences for the source model evaluation. For transfer learning in this study, all training and test data are used as training data.

For the target domain, training data is about 2,000 sentences for each emotion. Although the main task is regression prediction, we change it as distribution prediction (Tai et al., 2015). In this way, we deal it as a classification problem. Intensity scores y are changed to labels $\mathbf{t}$ satisfying:

$$t_i = \begin{cases} y' - \lfloor y' \rfloor & \text{if } i = \lfloor y' \rfloor + 1 \\ \lfloor y' \rfloor - y' + 1 & \text{if } i = \lfloor y' \rfloor \\ 0 & \text{otherwise} \end{cases}$$

$$\text{where } i = [1, 2, 3, 4, 5] \text{ and } y' = 4y$$

Size of the final output is 5. For example, if an intensity score y is 0.7, label $\mathbf{t}$ is [0, 0, 0.2, 0.8, 0].

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 300–304
New Orleans, Louisiana, June 5–6, 2018.

With given **r** = [0, 0.25, 0.5, 0.75, 1], label y can be obtained again by dot product with **t** and **r** (0.7 = 0.2*0.5 + 0.8*0.75).

2.2 Text preprocessing

To normalize words and remove noise in sentences, we use ekphrasis library (Baziotis et al., 2017). It helps to apply social tokenizer, spell correction, word segmentation and various preprocessing. We normalize time and number, and omit URL, email and user tag. Annotations are added on hashtags, emphasized and repeated words. We annotate them as a group because hashtags are gathered in many cases (see Table 1). Lastly, emoticons are changed to words that represent emoticons.

#letsdance #dancinginthemoonlight #singing
$\Rightarrow \langle hashtag \rangle$ lets dance dancing in the moonlight singing $\langle /hashtag \rangle$

Table 1: Example of preprocessing hashtag

2.3 Word embedding

We try five pre-trained word embeddings to choose the best one for the target model. Two are trained with GloVe (Pennington et al., 2014) using different data sets: one[1] is trained with very large data in Common crawl, and the other[2] is made with tweets (Baziotis et al., 2017). Other word embedding methods are fastText[3] (Bojanowski et al., 2016), word2vec[4] (Mikolov et al., 2013) and LexVec[5] (Salle et al., 2016). LexVec is the mixed version of GloVe and word2vec. Dimensions of them are all 300. Among them, GloVe with tweet is used for the source and transfer models.

Emoji can be good features but most of emoji ideograms are not contained in embedding vocabulary. Hence, we change a emoji to a phrase with python 'emoji' library. For example, 😀 is decoded to "Smiling Face with Open Mouth and Smiling Eyes". Because it is quite long, embedding vectors of emoji are changed to mean of vectors of each decoded words. In this way, we reduce Out-Of-Vocabulary and prevent the sentence from lengthening.

[1]https://nlp.stanford.edu/projects/glove/

[2]https://github.com/cbaziotis/datastories-semeval2017-task4

[3]https://github.com/facebookresearch/fastText

[4]https://code.google.com/archive/p/word2vec/

[5]https://github.com/alexandres/lexvec

2.4 LSTM and GLSTM

Recurrent Neural Network (RNN) works well in a sequence model like language by addressing its arbitrary length (Tai et al., 2015). However, RNN is difficult to be optimized because of a gradient vanishing problem. To solve it, LSTM suggested a cell state and gates as bridges to control the flow of error (Hochreiter and Schmidhuber, 1997).

GLSTM is just a group of several LSTMs, where outputs of LSTMs are concatenated. The idea is that LSTM can be divided into several sub-LSTMs (Kuchaiev and Ginsburg, 2017). This model has some advantages compared to the original LSTM. The number of parameters is reduced with a preserving feature size. Also, it can be parallelized and computation times are reduced because the computation of each sub-LSTM is independent.

2.5 Dropout

To avoid overfitting and achieve generality, we use three types of dropout. One is normal dropout between layers (Srivastava et al., 2014). If a shape of the layer is sequential, dropout mask is shared on sequential axis. Another dropout is inside cells of LSTM. In the each LSTM cell, the same dropout mask is applied on hidden values that come from the previous cell (Zaremba et al., 2014). Applying different dropout masks for each cell can mislead memory and information. With the same dropout mask, however, LSTM cell can dropout nodes consistently so that the model can forget or memorize information stably. The last one is dropout between sub-LSTMs. To get more generality, we dropped several LSTMs in GLSTM. For example, if GLSTM consist of five sub-LSTMs, we dropped two LSTMs and only use the rest three LSTMs.

3 Model structure

3.1 Source model

For the source model, Glove with tweets is used as input vectors of the embedding layer. After embedding layer, two GLSTM layers are stacked. GLSTM is made of 5 LSTMs with 40 feature size. Additionally, we concatenate forward and backward GLSTM to be bidirectional. So hidden size of each recurrent layer is 400 ($= 5 \times 40 \times 2$).

Next is an attention layer, which calculates importance of each time step. Attention mechanism shows good performance on sequential tasks like

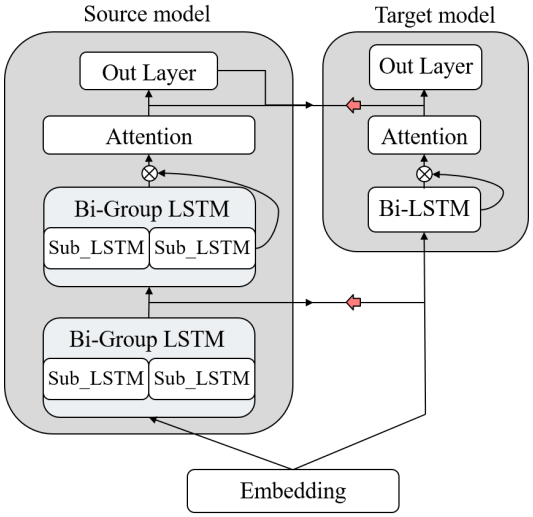

Figure 1: Structure of models. For the transfer model, connections between source and target models are used. Large arrows are paths of reduced gradient flow during backpropagation.

machine translation (Bahdanau et al., 2014) and sentiment analysis (Baziotis et al., 2017). It helps to concentrate position related to emotion. Attention values are calculated:

$$e_t = W_h h_t + b_t$$

$$a_t = \frac{exp(e_t)}{\sum_i^l exp(e_i)}, \quad \sum a_t = 1$$

Calculated attention values are multiplied by each current hidden state and they are all added up.

Passing through the attention layer, the output becomes non-sequential representation vectors. It enters a fully connected softmax layer as a final classification layer, where the size of the layer is 3.

3.2 Target model

Unlike the source model, a normal bi-LSTM is used with 100 feature size. After then, attention and output layers are stacked. The size of output layer is 5.

For transfer learning, outputs of several layers on the source model are used as additional features. The LSTM layer on the target model takes as input the concatenation of the embedding layer and the first LSTM layer output of the source model. After the attention layer, in a similar way, outputs of the attention and the final layers on the source model are concatenated and entered into the final layer as input.

3.3 Regularization

At the embedding layer, Gaussian noise is applied with sigma = 0.2. It helps models to be robust by avoiding overfitting on specific features of words. Dropouts are used everywhere between layers with probability p = 0.3 except before the final layer. Before the final layer, p = 0.5 dropout is applied. Additionally, LSTM dropout was applied on every LSTM layers with p = 0.3. The probability of dropout at GLSTM on the source model is 0.3. Also, we use L2 normalization. It prevents weights to be large values by adding weight penalty to loss. We set up it with 0.001 for the source model and 0.0001 for the target model.

3.4 Training

For the source and target models, categorical cross-entropy is used as a loss function. For updating weights, we apply the Adam (Kingma and Ba, 2014) optimizer with a learning rate of 0.001. During training the transfer model, since we want to preserve target model weight parameters with a little updating, we decrease gradient flow of backpropagation from the source model to the target model by 0.05 times (see large arrows on Figure 1). Because there are many parameters on the final model, we take that constraint to prevent overfitting.

4 Result and discussion

4.1 GLSTM

Figure 2 shows the result of GLSTM and normal LSTM on the source model for Sentiment Classification (SemEval 2017 Task 1a). We tried various feature sizes. The number of sub-LSTM in GLSTM is fixed to 5 and the feature size of each sub-LSTM is changed. As the sizes of features increase, the performances of GLSTM increase. On the other hand, although the performances of LSTM gradually improve with larger feature sizes, it starts to decrease rapidly after 100. Thus, we infer that GLSTM with dropout is more effective on overfitting than LSTM with larger feature size. Based on this result, we use GLSTM for the source model.

4.2 Various Embedding

We tested five different word embedding vectors using the target model to choose the best embedding. To compare the performances of embeddings, the embedding layers was not trained

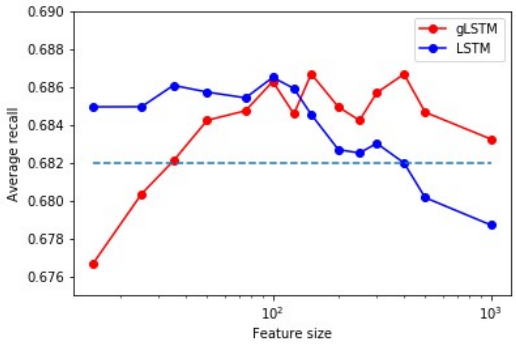

Figure 2: Performance comparison between GLSTM and LSTM on the source model for sentiment classification. A dotted line is the result of (Baziotis et al., 2017).

Embedding	Avg	Anger	Fear	Joy	Sadness
Tweet GloVe	**.690**	**.730**	**.670**	**.675**	.684
Common GloVe	.667	.690	.624	.656	**.698**
Fast Text	.639	.667	.586	.638	.665
Word2vec	.654	.704	.599	.631	.681
Lex vec	.648	.656	.630	.646	.659

Table 2: Pearsons correlation of Dev set on the target model for SemEval-2018 Task1(a).

(static). Note that we did not use transfer learning in this experiment. Table 2 shows Pearson correlation between the given emotion intensities and predicted intensities by the models on the development set. Tweet GloVe had the best score and Common GloVe showed the second best score. Hence, we decided to do transfer learning with Tweet GloVe and Common GloVe.

4.3 Transfer

Our main task results are described in Table 3. There are four models. Tweet Glove and Common GloVe were picked from the conclusion of 4.2, and we performed two approaches: training the embedding layer or not (non-static or static) (Kim, 2014). Tweet GloVe with static showed the best performance as a single model and it is almost same to non-static. However, the non-static method had a higher score than the static for Common GloVe embedding. In addition, the ensemble model by averaging all single models showed better performance than the single models. We also found that compared to the scores without transfer learning on dev set (Table 2), there were significant performance improvements when transfer learning used in Table 3.

5 Conclusion

This paper described the system submitted to SemEval-2018 Task 1: Affect in tweets and analysis of various models. Various embedding vectors were tried and we chose Tweet GloVe with static. The main method is LSTM with attention and transfer learning that uses sentiment classification as source domain. In future work, we will perform transfer learning with labeled data sets such as SNLI or SST data sets. Also, training tagging or tree parsing can be used for transfer learning.

Acknowledgments

This research was supported by the Bio-Synergy Research Project (NRF-2016M3A9C4939665) of the Ministry of Science, ICT and Future Planning through the National Research Foundation.

References

Md Shad Akhtar, Palaash Sawant, Asif Ekbal, Jyoti Pawar, and Pushpak Bhattacharyya. 2017. Iitp at emoint-2017: Measuring intensity of emotions using sentence embeddings and optimized features. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 212–218.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Embeding	Avg		Anger		Fear		Joy		Sadness	
	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev
Tweet GloVe static	**.731**	.732	.755	.774	**.731**	**.698**	.713	.728	.726	.727
Tweet GloVe Non -static	.730	**.733**	**.760**	**.777**	.708	.686	**.720**	**.736**	**.732**	**.734**
Common GloVe static	.689	.695	.707	.702	.704	.677	.680	.688	.665	.712
Common GloVe Non -static	.700	.721	.718	.738	.700	.684	.681	.725	.700	.735
Ensemble	.753	.755	.773	.786	.753	.720	.729	.744	.758	.768

Table 3: Experiment results of the transfer model on SemEval-2018 Task 1(a) Emotional Intensity regression. The submitted system to the task is Tweet GloVe with static.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655.

Venkatesh Duppada and Sushant Hiray. 2017. Seernet at emoint-2017: Tweet emotion intensity estimator. *arXiv preprint arXiv:1708.06185*.

Pranav Goel, Devang Kulshreshtha, Prayas Jain, and Kaushal Kumar Shukla. 2017. Prayas at emoint 2017: An ensemble of deep neural architectures for emotion intensity prediction in tweets. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 58–65.

Yuanye He, Liang-Chih Yu, K Robert Lai, and Weiyi Liu. 2017. Yzu-nlp at emoint-2017: Determining emotion intensity using a bi-directional lstm-cnn model. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 238–242.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Oleksii Kuchaiev and Boris Ginsburg. 2017. Factorization tricks for lstm networks. *arXiv preprint arXiv:1703.10722*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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.

Jeffrey Pennington, Richard Socher, and Christopher 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.

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.

Alexandre Salle, Marco Idiart, and Aline Villavicencio. 2016. Matrix factorization using window sampling and negative sampling for improved word representations. *arXiv preprint arXiv:1606.00819*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. 2016. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.