MainiwayAI at IJCNLP-2017 Task 2: Ensembles of Deep Architectures for Valence-Arousal Prediction

Yassine Benajiba **Jin Sun** **Yong Zhang** **Zhiliang Weng** **Or Biran**

Mainiway AI Lab
{yassine,jin.sun,yong.zhang,zhiliang.weng,or.biran}@mainiway.com

Abstract

This paper introduces Mainiway AI Labs submitted system for the IJCNLP 2017 shared task on Dimensional Sentiment Analysis of Chinese Phrases (DSAP), and related experiments. Our approach consists of deep neural networks with various architectures, and our best system is a voted ensemble of networks. We achieve a Mean Absolute Error of 0.64 in valence prediction and 0.68 in arousal prediction on the test set, both placing us as the 5th ranked team in the competition.

1 Introduction

While traditional sentiment analysis is concerned with discrete polarity classes, the dimensional approach has recently drawn considerable attention as a way of modeling sentiment more accurately. In this framework, affective states are represented as points in a multi-dimensional continuous space, such as the Valence-Arousal (VA) space described by Russell (1980), and shown in Figure 1.

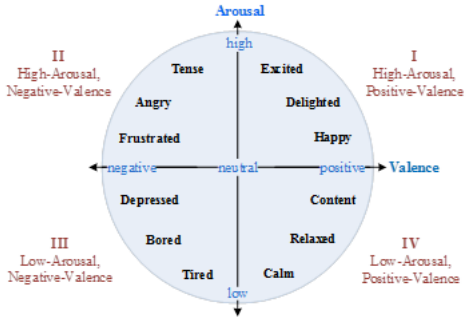

Figure 1: The two-dimensional Valence-Arousal space.

The DSAP shared task at IJCNLP 2017 is the second task related to Chinese sentiment analysis in the VA space, the first being the Dimensional Sentiment Analysis of Chinese Words at IALP 2016 (Yu et al., 2016a)(Yu et al., 2016b). The evaluation metrics in DSAP, as in the previous competition, are the Mean Absolute Error (MAE) and the Pearson Correlation Coefficient (PCC).

In contrast to the IALP task, where the test set consisted of single words only, the DSAP test set contains both single words and multi-word phrases. The variable length of the input adds a technical complexity to the typical modern approach to this task, which relies on an artificial neural network applied to the input in an embedding space, as phrase-level representations present many difficulties and are an ongoing area of research.

This paper presents our method of acquiring embedded representations, the various network architectures we utilize in the final system and our ensemble method, along with pre-competition experiments. In addition, we discuss the results of experiments with a nearest neighbor-based smoothing approach that was not included in the final system but provided interesting and valuable insight.

2 Models and Embeddings

This section describes several deep neural network architectures used in our experiments as well as the embedded word representation we use as input to the networks.

2.1 Embeddings

Word embeddings - continuous vector representations of textual units such as words or characters - are a standard approach for representing semantics. One popular option is *word2vec*, which generates embeddings for single words using either the CBOW or the skip-gram model (Mikolov et al., 2013). A more powerful approach is that

Proceedings of the 8th International Joint Conference on Natural Language Processing, Shared Tasks, pages 118–123,
Taipei, Taiwan, November 27 – December 1, 2017. ©2017 AFNLP

of *fastText*,[1] a tool for generating embeddings at the word or character level (Bojanowski et al., 2017). Like *word2vec*, *fastText* uses either the CBOW or the skip-gram model to learn embeddings from a text corpus. The difference between the two is illustrated in Figure 2; in this paper, we use character-level embeddings of 300 dimensions trained on Chinese Wikipedia with *fastText* using the skip-gram model.

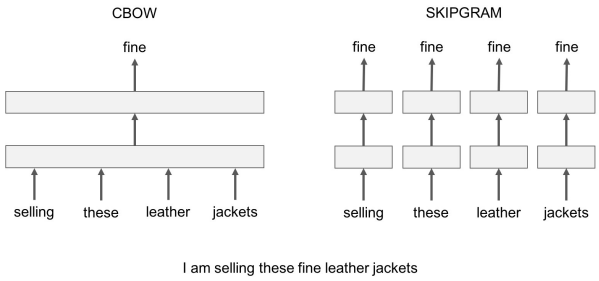

Figure 2: CBOW and skip-gram training architectures

The input to the neural network (described in the next section) is the embedding space representation of the text input. In the case of unseen words or phrases, the representation is constructed by *fastText* from its constituent character n-grams.

2.2 Network architecture

We treat the prediction of valence and arousal values as two regression tasks, and use a deep neural network to represent them. As the competition rules allow two separate runs, we chose two of these architectures as our final systems; in this section, we describe several architectures and ideas from which we made our choices. Section 3 describes the experimental results for these options, as well as the two systems that participated in the competition.

In all variations described below, the input layer is a 300-dimensional embedding space and the output layer is a single output. The hidden layers are all dense, with ReLU for the activation function and Adam for the optimization algorithm, with a batch size of 135 and no dropout. The loss function is the Mean Squared Error (MSE).

Using these base parameters, we first experiment with varying the number of hidden layers. Using a *rectangle* architecture, where all hidden layers consist of 300 fully connected neurons, we experimented with zero, three, and six hidden layers.

In addition to the rectangle architecture, we also tried an *hourglass* architecture with six hidden layers where the dimensionalities of the hidden layers (after the 300-dimensional input layer) are 225, 175, 100, 175, 225 and 300. The idea here is to compact the semantic information of the embeddings into a lower-dimensional sentiment information as it passes through the network, alleviating overfitting. This is conceptually similar to the intuition behind Auto Encoders.

Finally, we leverage both architectures using an ensemble method which averages the predictions of both models. To generalize from a two-model ensemble, we define a $2n$-model ensemble which averages between the predictions of $2n$ models, n rectangular (R) and n hourglass (H):

$$y = \frac{\lambda \sum_{i=1}^{n} R_i(x) + (1-\lambda) \sum_{i=1}^{n} H_i(x)}{n}$$

Where λ is a tunable weight hyperparameter which in this work we keep at 0.5. This adds a final layer to the model, illustrated for 20 models in Figure 3. The ensemble approach reduces the variance of the results and increase the accuracy of individual predictions.

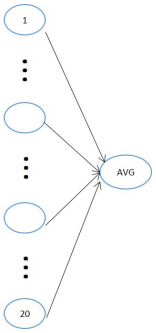

Figure 3: A final layer averaging 20 model predictions

3 Experimental Results and Task Results

This section describes the experiments we conducted with the variants described in the previous section as well as the final selected systems and their performance in the competition.

In addition to our architecture variants, we include two baselines to put these results in context: a linear classifier baseline, and a single-layer Boosted Neural Network (BNN), an architecture

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

Model	Layers (dense)
Rectangle, no hidden layers	[300; 1]
Rectangle, 3 hidden layers	[300; 300, 300, 300; 1]
Rectangle, 6 hidden layers	[300; 300, 300, 300, 300, 300, 300; 1]
Hourglass, 6 hidden layers	[300; 225, 175, 100, 175, 225, 300; 1]
Ensemble, $n = 1$	Ensemble of Rectangle(6) and Hourglass(6)
Ensemble, $n = 10$	Ensemble of Rectangle(6) $\times 10$ and Hourglass(6) $\times 10$

Table 1: Model architectures

Model	CVAW				CVAP			
	Valence		Arousal		Valence		Arousal	
	MAE	PCC	MAE	PCC	MAE	PCC	MAE	PCC
Linear	0.740	0.805	0.877	0.593	1.044	0.657	0.579	0.785
BNN	0.633	0.856	0.762	0.690	0.706	0.848	0.505	0.844
Rectangle, no hidden layers	0.697	0.829	0.838	0.638	0.659	0.864	0.521	0.840
Rectangle, 3 hidden layers	0.577	0.869	0.787	0.670	0.491	0.913	0.465	0.867
Rectangle, 6 hidden layers	0.585	0.854	0.816	0.634	**0.452**	**0.909**	0.488	0.850
Hourglass, 6 hidden layers	0.580	0.863	0.793	0.663	0.502	0.907	0.471	0.861
Ensemble, $n = 1$	0.557	0.878	0.771	0.679	0.487	0.912	0.459	0.870
Ensemble, $n = 10$	**0.531**	**0.887**	**0.740**	**0.707**	0.461	0.922	**0.448**	**0.876**

Table 2: Experimental results

that was popular at the IALP 2016 task (Du and Zhang, 2016). The architectures of the different variants are listed in detail in Table 1.

The results are shown in Table 2, highlighted by best MAE. The first thing to notice is that all deep architectures (three and six hidden layers) perform significantly better than the baseline and the BNN; that result validated our intuition that predicting a sentiment output from a semantic-space input requires multiple transformative layers.

Clearly, the $n = 10$ ensemble model has the best performance overall. The 6-hidden-layer rectangle model achieves better performance in Valence on CVAP, however. We therefore chose these two models as our two systems for the competition: the rectangular model with 6 hidden layers for Run 1, and the $n = 10$ ensemble model for Run 2. The architectures of these models are illustrated further in Figure 4 and Figure 5 for Run 1 and Run 2, respectively.

Note that while the same architecture was used in each run for both valence and arousal, the model was trained separately for each of the two tasks, using the training data provided for the competition.

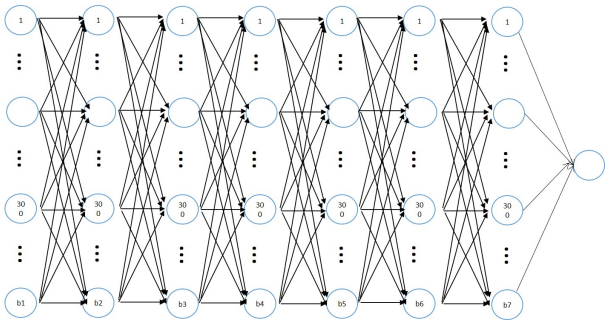

Figure 4: Run 1 architecture diagram

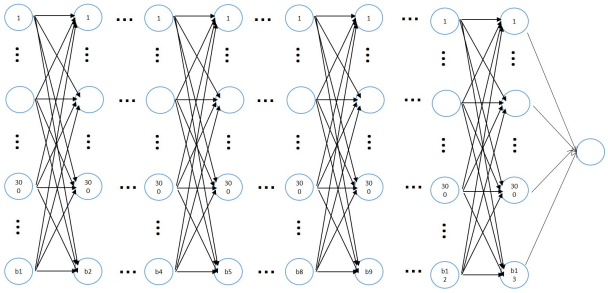

Figure 5: Run 2 architecture diagram

Submission	Valence				Arousal				Mean
	MAE	Rank	PCC	Rank	MAE	Rank	PCC	Rank	Rank
THU_NGN-Run2	0.427	1	0.9345	1	0.6245	1	0.7985	1	1
THU_NGN-Run1	0.4795	2	0.9085	2	0.6645	4	0.766	3	2.75
AL_I_NLP-Run2	0.5355	3	0.8965	3	0.661	3	0.766	2	2.75
AL_I_NLP-Run1	0.539	4	0.8955	4	0.659	2	0.761	4	3.5
CKIP-Run1	0.547	7	0.8895	6	0.6655	5	0.742	5	5.75
NCTU-NTUT-Run1	0.543	5	0.887	7	0.72	6	0.695	8	6.5
NCTU-NTUT-Run2	0.546	6	0.8865	8	0.7285	7	0.699	7	7
CKIP-Run2	0.5545	8	0.895	5	0.764	10	0.7365	6	7.25
MainiwayAI-Run2	**0.6415**	**9**	**0.837**	**10**	**0.7545**	**9**	**0.6825**	**9**	**9.25**
MainiwayAI-Run1	**0.6635**	**10**	**0.8285**	**11**	**0.793**	**13**	**0.651**	**11**	**11.25**
FZU-NLP-Run1	0.8165	14	0.7655	13	0.7425	8	0.6535	10	11.25
NTOU-Run2	0.757	13	0.7365	15	0.7775	12	0.61	12	13
CIAL-Run1	0.6835	11	0.844	9	0.9765	21	0.5895	14	13.75
NTOU-Run1	0.6925	12	0.805	12	0.7765	11	0.5225	22	14.25
CASIA-Run1	0.8665	16	0.7005	17	0.9425	19	0.5555	15	16.75
NLPSA-Run2	0.8445	15	0.7165	16	0.92	17	0.539	20	17
Baseline	**1.0175**	**20**	**0.6265**	**22**	**0.819**	**14**	**0.593**	**13**	**17.25**
NLPSA-Run1	0.9085	18	0.6895	18	0.9195	16	0.5415	18	17.5
NCYU-Run1	0.9785	19	0.685	19	0.945	20	0.549	16	18.5
SAM-Run1	1.029	21	0.654	21	0.8745	15	0.541	19	19
CIAL-Run2	0.898	17	0.7485	14	1.316	24	0.356	23	19.5
FZU-NLP-Run2	1.0675	22	0.5765	23	0.92	18	0.544	17	20
NCYU-Run2	1.205	23	0.6665	20	0.989	22	0.534	21	21.5
XMUT-Run1	1.3345	24	0.3825	24	1.0995	23	0.2675	24	23.75

Table 3: task results.

3.1 Task Results

The final competition results are shown in Table 3, with our system highlighted. The ensemble approach of Run 2 achieved higher results than the rectangular network of Run 1, which is consistent with our experimental results. Compared to other participants, we ranked fifth out of the 13 participating teams, with the Tsinghua and Alibaba teams coming first and second, respectively. The Run 2 system retained a very consistent rank of 9-10 across all task / dataset combinations.

4 Nearest Neighbor Experiments

The major drawback of embeddings when we are modeling a sentiment-related problem is that words of opposite polarities obtain very similar embeddings since they occur in similar contexts. The experiments we report in the previous sections show our research work to build a robust statistical model for valence and arousal despite this limitation of the embeddings. As we have shown, significantly good results can be obtained by using a deep network of dense layers.

We have experimented with a different idea that we would like to discuss separately in this section since we did not include it in the runs we submitted to the shared task. This idea consists of enriching the *fastText* embeddings with additional dimensions that indicate the valence/arousal values of the immediate semantic neighborhood in which the word we want to score is encountered. The steps that we take to infer these new dimensions can be described as follows:

1. $w \leftarrow$ word we want to score
2. $v \leftarrow$ fastText embedding of w
3. $nnList \leftarrow$ list of N nearest neighbors of v in the training set
4. $sortedNnList \leftarrow$ sort $nnList$ from lowest to highest value of valence/arousal score

As specified in the algorithm, the sorting is performed according to the score. Consequently, we obtain a different embedding for valence and arousal since both the order and the values of the new dimensions are different. Also, it is important to note that, the main reason behind sorting the scores of the nearest neighbors is to give a semantic consistency to the newly added features. By doing so, we make sure that the neural network that is going to perform the regression finds in these new dimensions an indication of the distribution of valence/arousal values in the immediate semantic neighborhood of the word we want to score. Sorting the nearest neighbors according to distance does not allow to achieve this semantic consistency of the dimensions.

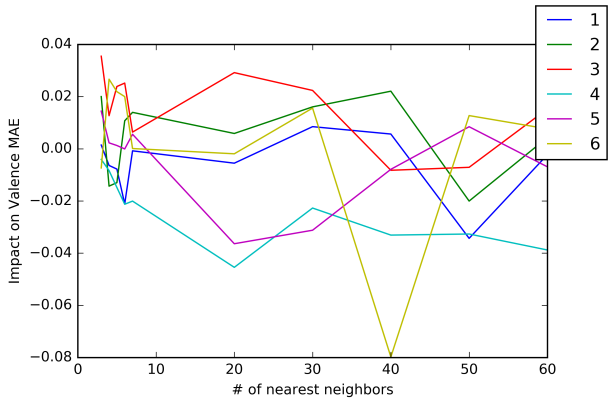

Figure 6: Impact on Valence MAE for neural networks of different depths after adding N nearest neighbor features

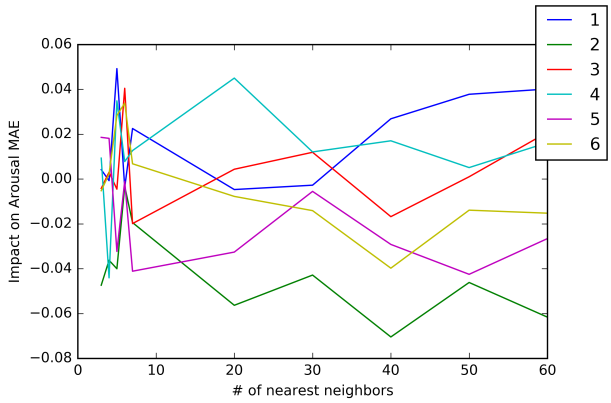

Figure 7: Impact on Arousal MAE for neural networks of different depths after adding N nearest neighbor features

Figures 6 and 7 show the obtained results.

The figures show the impact of adding the new features discussed in this section with difference values of N for neural networks with different layers. The general trend for both valence and arousal is that there an inflection point before and after which the new features either hurt or don't significantly help the results. We can also see however that for arousal the impact of the new features is significant even when N has a value between 3 and 7. Whereas for valence, we can only see a significant impact when the value of N is above 20.

This means that there is no clear correlation between the semantic neighborhood and the valence/arousal score, yet the situation is seemingly different between valence and arousal at the detailed level. For this reason, we wanted to go one step further in our analysis to shed more light on the relationship between the Euclidean distance (which we use to find the nearest neighbors) and the valence/arousal score. This is important specially because the learning machine needed information from less neighbors and ended up benefiting more from them for arousal scoring than valence.

In order to be able to visualize and try to interpret this relationship the difference between the score of a word from the mean score of its semantic neighborhood, i.e. we compute the histogram of the z-score of the words score with respect to the distribution of their semantic neighborhood. Figures 8 and 9 show these histograms for valence and arousal, respectively.

The two histograms clearly show that whereas the arousal distribution is similar to a Gaussian distribution centered around 0, the valence distribution looks more like a mixture of two Gaussians one centered around 5 and the other around -5. In other words, the arousal score of a word is more often than not the average arousal score of its immediate neighbors. This explains why in our experiments the nearest neighbor features were very helpful. For valence, however, the score of a word is very often at a standard deviation distance of -5 or 5 from the mean. We believe, this could be the main reason why it was harder for the neural network to use the semantic neighborhood information more efficiently in the case of valence.

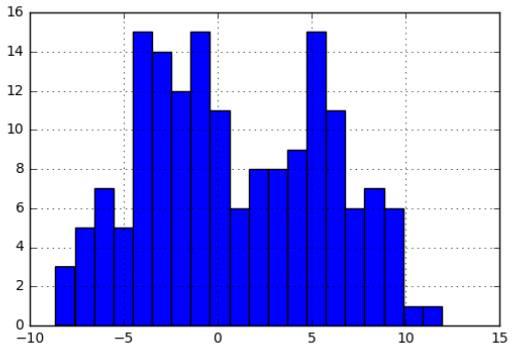

Figure 8: Histogram of Valence Z-score in the immediate Semantic Neighborhood

Finally, we would like to mention that the main reason we didn't include these new features in our final run is because the final MAE is better when we use deep networks with fastText features only.

5 Conclusion and Future Work

In this paper we describe our participation in DSAP 2017 shared task. We describe the various neural networks we explored. We also describe how we use ensemble methods to improve

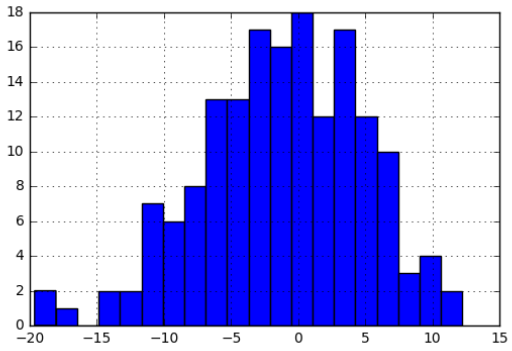

Figure 9: Histogram of Arousal Z-score in the immediate Semantic Neighborhood

the performance. In the latter, each classifier is trained independently with fine tuned layer parameters and we combine them by voting (averaging the scores).

In this work, we use character n-gram based word embedding with the help of *fastText*, hence the performance is limited by the embedding itself. We report our analysis of the relationship between the semantic neighborhood of a word and its valence/arousal score. In the future, we would extend our work to use sub-character components (radicals/strokes/components) learning in together with word embeddings to improve the performance further.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5:135–146.

Steven Du and Xi Zhang. 2016. Aicyber's system for IALP 2016 shared task: Character-enhanced word vectors and boosted neural networks. In *2016 International Conference on Asian Language Processing, IALP 2016, Tainan, Taiwan, November 21-23, 2016*. pages 161–163.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. 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.

J.A. Russell. 1980. A circumplex model of affect. *Journal of personality and social psychology* 39(6):1161–1178.

Liang-Chih Yu, Lung-Hao Lee, Shuai Hao, Jin Wang, Yunchao He, Jun Hu, K. Robert Lai, and Xuejie Zhang. 2016a. Building chinese affective resources in valence-arousal dimensions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 540–545.

Liang-Chih Yu, Lung-Hao Lee, and Kam-Fai Wong. 2016b. Overview of the IALP 2016 shared task on dimensional sentiment analysis for chinese words. In *2016 International Conference on Asian Language Processing, IALP*. Tainan, Taiwan, pages 156–160.