INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Minimization of Regression and Ranking Losses with Shallow Neural Networks on Automatic Sincerity Evaluation

Hung-Shin Lee[1,2], *Yu Tsao*[3], *Chi-Chun Lee*[4], *Hsin-Min Wang*[2]
Wei-Cheng Lin[4], *Wei-Chen Chen*[4], *Shan-Wen Hsiao*[4], *Shyh-Kang Jeng*[1]

[1]Department of Electrical Engineering, National Taiwan University, Taiwan
[2]Institute of Information Science, Academia Sinica, Taiwan
[3]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
[4]Department of Electrical Engineering, National Tsing Hua University, Taiwan
hslee@iis.sinica.edu.tw, yu.tsao@citi.sinica.edu.tw, cclee@ee.nthu.edu.tw

Abstract

To estimate the degree of sincerity conveyed by a speech utterance and received by listeners, we propose an instance-based learning framework with shallow neural networks. The framework plays as not only a regressor that intends to fit the predicted value to the actual value but also a ranker that preserves the relative target magnitude between each pair of utterances, in an attempt to derive a higher Spearman's rank correlation coefficient. In addition to describing how to simultaneously minimize regression and ranking losses, the issue of how utterance pairs work in the training and evaluation phases is also addressed by two kinds of realizations. The intuitive one is related to random sampling while the other seeks for representative utterances, named anchors, to form non-stochastic pairs. Our system outperforms the baseline by more than 25% relative improvement in the development set.

Index Terms: regression, ranking, degree of sincerity, shallow neural networks, computational paralinguistics

1. Introduction

Given a speech utterance, the goal of automatic sincerity evaluation is to tell us how sincere it could convey to human receivers by means of a learning machine. Without doubt, sincerity itself is a kind of very subjective affection with seemingly unmeasurable range due to individual differences. As a result, it is usually much easier to acquire the quantitative degree of sincerity that a listener feels than the mental state of sincerity that a speaker has while speaking. The compromise left a broad space for the progress of some applications, such as personal image consulting and performing arts. For example, with a well-trained machine, a politician can understand how sincerity the voters will feel after hearing the speech during a budget-insufficient campaign; an actor/actress can revise his/her speaking style according to the machinery feedback before an audition. In spite of a relatively limited amount of research and experimentation done in recent years regarding the latent factors of speaking sincerity [1, 2], we can still leverage machine learning strategies to obtain useful cues for sincerity evaluation if the coverage of linguistic content and prosodic types is satisfactory in the collected data and the respective labels are well annotated [3].

Evaluation metrics. In contrast to a similar task of sarcasm recognition that deals with binary classification [4, 5], automatic sincerity evaluation is treated as a problem of *regression* in this paper. That is, given a speech utterance, a predicted sincerity value has to be generated and expected to be as close to the actual value rated by annotators as possible. However, instead of the well-known mean squared error (MSE), the Spearman's rank correlation coefficient (ρ) [6], used in the research on human perception modeling and psycholinguistics [7, 8], has become a standard evaluation metric in paralinguistic computation in the Degree of Nativeness and Parkinsons Condition subchallenges in ComParE 2015 [9]. Spearman's ρ assesses how well the relationship between predicted and actual values can be described using a monotonic function by virtue of their own relative *ranks*. To our knowledge, there are two reasons why to adopt it. First, it is less sensitive to extreme values so that the evaluation of a system can be insusceptible to occasionally predicted outliers [10]. Second, a sincerity evaluation system might be good in a pragmatic sense, *not* because it can precisely tell us how annotators intend to rate, for these ratings are not as sensible as something like meters and grams in the metric system to ordinary people. Contrarily, a good system can give us a series of ordinal rank of a set of utterances to represent their relative degree of sincerity. Decision making by comparison is always much easier for we humans.

Possible problems. Many popular regression methods, including artificial neural networks (ANN) [11] and support vector regression (SVR) [12], are not designed ad hoc to achieve higher Spearman's ρ, but to minimize the residual sum of squares between the ground truth and predicted responses directly or through margin maximization [13]. Notwithstanding a model that gives perfect regression will also give perfect ranking, a model with near-perfect regression performance may yield poor ranking performance. For example, a regressor makes predictions of [1.1, 1.25, 1.24] with an MSE of only 0.002 but a ρ of 0.5 with respect to the true values of [1.1, 1.2, 1.3]. For a well-trained ranker, however, even if its predicted values of [4.1, 5.2, 6.3] result in a much worse MSE, it still achieves a perfect ρ. Therefore, even in less extreme cases, small regression errors can cause large ranking errors. Note that it does not indicate that we should abandon criteria aiming at minimizing regression errors and entirely pursue the maximization of Spearman's ρ, although ideally we have to. Unfortunately, there seems little research on this goal until now. Although some listwise approaches have been proposed in information retrieval for the purpose of optimizing other ranking measures of the test samples, such as the normalized discounted cumulative gain (NDCG) [14, 15, 16], they are not readily suitable for the evaluation scenario in this paper since each test sam-

http://dx.doi.org/10.21437/Interspeech.2016-756

ple has to be evaluated identically and independently.

Contributions. To solve the aforementioned problems, we propose a framework with shallow neural networks using an objective function for minimizing both regression and pairwise ranking losses. In contrast to a similar idea presented in [17], where the two losses are minimized *alternatively* for training a linear model by introducing a tradeoff coefficient that helps randomly fetch an input out of a single sample and a candidate pair, the proposed training algorithm minimizes the two losses *simultaneously*. In addition, this paper has two more contributions. First, two kinds of realizations are put forward to demonstrate how utterance pairs are generated and work in the whole learning and evaluating mechanism. Second, the use of artificial neural networks with a proper amount of regularization circumvents the small sample size problem in our task while sustaining the generalization ability [18, 19, 20].

The remainder of this paper is organized as follows. Section 2 gives our objectives in regression and ranking. Section 3 presents two kinds of architectures based on shallow neural networks and shows how to derive the optimal model by minimizing the regression and ranking losses simultaneously. Section 4 reports the experiment results on the Sincerity sub-challenge of ComParE 2016. Finally, Section 5 gives the conclusions.

2. Objectives

2.1. Regression and Ranking Losses

The goal of supervised regression is to learn a model $\mathcal{M}_{rg}$ that can predict a real-valued target $y \in \mathbb{R}$ for a feature vector $\mathbf{x}$ using a prediction function $f(\mathbf{x})$ with little loss with respect to a specified loss function. Given a set of labeled training data $\mathcal{D}$, the aggregate regression error based on the residual sum of squares between target y and predicted $f(\mathbf{x})$ is given by

$$L_{rg}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},y)\in\mathcal{D}} \left(f(\mathbf{x}) - y\right)^2 . \quad (1)$$

Therefore, a well-trained regression model $\mathcal{M}_{rg}$ that minimizes $L_{rg}(\mathcal{D})$ can be expected that the predicted degree of sincerity is desirably close to the human annotated one.

In the same vein of RankSVM [21], given the difference $\Delta\mathbf{x}_{ab}$ of two feature vectors $\mathbf{x}_a$ and $\mathbf{x}_b$, the goal of a supervised pairwise ranking method is to learn a model $\mathcal{M}_{rk}$ that can predict the difference Δy_{ab} of target values y_a and y_b by using a prediction function $f(\Delta\mathbf{x}_{ab})$, where $\Delta y_{ab} = y_a - y_b$ and $\Delta\mathbf{x}_{ab} = \mathbf{x}_a - \mathbf{x}_b$. Suppose a set of training pairs $\mathcal{P}$ is selected from $\mathcal{D}$, the incurred rank-based loss is given by

$$L_{rk}(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{((\mathbf{x}_a,y_a),(\mathbf{x}_b,y_b))\in\mathcal{P}} \left(f(\Delta\mathbf{x}_{ab}) - \Delta y_{ab}\right)^2 . \quad (2)$$

Therefore, by minimizing $L_{rk}(\mathcal{P})$, the prediction function, to some extent, attempts to guarantee that if utterance a sounds more sincere than b, i.e., $y_a > y_b$, then a will have a larger predicted value than b. Actually, (1) can be regarded as a special case of (2) if we set utterance b to a trivial comparing reference $(\mathbf{x}_b, y_b) = (\mathbf{0}, 0)$ for all utterances a.

2.2. The Combination of Regression and Ranking Losses

As mentioned in Section 1, although the final evaluation metric in our task is totally rank-based, we still need to minimize the regression loss to a degree in order to 1) reduce the number of uncontrollable occurrences of outliers and to 2) complement

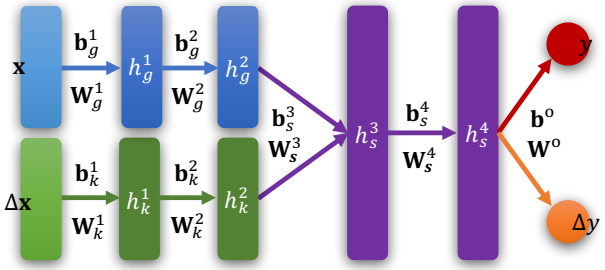

Figure 1: *The architecture of spR2NNs.*

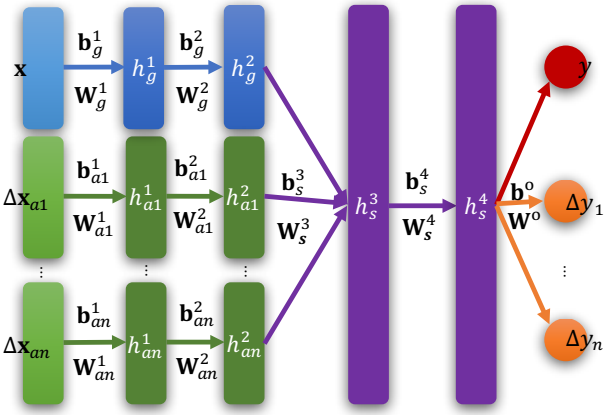

Figure 2: *The architecture of acR2NNs.*

the pairwise ranking loss, which is actually a makeshift suboptimal to the minimization of overall ranking error. Finally, by assuming that $\mathcal{M}_{rg}$ and $\mathcal{M}_{rk}$ share the same model $\mathcal{M}$ and combining (1) and (2), the new goal of the training process is to find an optimal $\mathcal{M}$ by minimizing

$$\mathcal{J} = \alpha L_{rg}(\mathcal{D}) + (1-\alpha) L_{rk}(\mathcal{P}) + \lambda \|\mathcal{M}\|_2, \quad (3)$$

where $\alpha \in [0, 1]$ is a weight to adjust the importance of the regression loss and the pairwise ranking loss, while λ is a regularization parameter that controls the complexity of the model.

3. Realizations

To realize the objective function (3) by means of artificial neural networks, we propose two structures, in light of multi-modal and multi-task neural networks [22, 23, 24, 25, 26, 27, 28]. They mainly differ in 1) the way that the utterance pairs are generated and used and 2) the way that a test sample is fed into the machine while predicting.

3.1. The Sampling-based Neural Network

The architecture of the proposed sampling-based regression and ranking neural networks (spR2NNs) is depicted in Figure 1. Each input stream, i.e., feature vectors in $\mathcal{D}$ or difference vectors in $\mathcal{P}$, is first independently modeled by separate neural networks, built up of hidden layers $\{h_g\}$ and $\{h_k\}$, respectively, and their corresponding parameters $\{\mathbf{W}, \mathbf{b}\}$. After a succession of encoder functions $h^1 = \varphi(\mathbf{W}^1\mathbf{x} + \mathbf{b}^1)$ and $h^i = \varphi(\mathbf{W}^i h^{i-1} + \mathbf{b}^i)$, where $i > 1$ and $\varphi(\cdot)$ is the activation function, the last separate layers are jointly connected to the strongly parameter-shared layers, denoted by $\{h_s\}$, which tend to mix the features learned from two kinds of data streams, namely, $\mathbf{x}$ and $\Delta\mathbf{x}$. The final output layer, resulted from the last layer of $\{h_s\}$ through the linear activation function, is constituted by only two nodes that represent the prediction results and relate to the ground truth and its difference, y and Δy.

Algorithm 1 The Training Procedure for spR2NNs

Input: The training data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, the development data $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_K]$, their respective label vectors $\mathbf{y}$ and $\mathbf{v}$, the initial model $\mathcal{M}^{(0)}$, the maximum number of epochs T, and the tolerance τ.

Output: The model estimate $\hat{\mathcal{M}}$.

1: **for** $i = 1$ to T **do**
2: Randomly permute $\mathbf{X}$ with respect of its columns and return a new $\mathbf{X}'$ and the corresponding $\mathbf{y}'$.
3: Calculate $\Delta\mathbf{X} = \mathbf{X} - \mathbf{X}'$ and $\Delta\mathbf{y} = \mathbf{y} - \mathbf{y}'$.
4: Update parameters of each layer with (4).
5: **if** the development set $\mathbf{U}$ does not exist **then**
6: Use Algorithm (2) to calculate ρ_i, the Spearman's ρ of $\mathbf{y}$ and the predicted vector from $\mathbf{X}$.
7: **if** $i \mod 100 = 0$ **then**
8: Calculate $\bar{\rho}_i$ by averaging $\rho_{i-100}, \ldots, \rho_i$.
9: **if** $|\bar{\rho}_i - \bar{\rho}_{i-100}|/\bar{\rho}_{i-100} < \tau$ **then**
10: Store the model $\mathcal{M}^{(i)}$ as $\hat{\mathcal{M}}$ and break the loop.
11: **end if**
12: **end if**
13: **else**
14: Use Algorithm (2) to calculate the Spearman's ρ from the development set $(\mathbf{U}, \mathbf{v})$ and store the model that results in the best ρ so far as $\hat{\mathcal{M}}$.
15: **end if**
16: **end for**

Algorithm 2 The Prediction Procedure for spR2NNs

Input: The feature matrix $\mathbf{X}$, the model $\hat{\mathcal{M}}$.

Output: The predicted vector $\mathbf{y}$.

1: Set $\Delta\mathbf{X}$ to be a zero matrix with the same shape of $\mathbf{X}$.
2: Feed $\mathbf{X}$ and $\Delta\mathbf{X}$ into the spR2NNs with $\hat{\mathcal{M}}$ and return $\mathbf{y}$ and $\Delta\mathbf{y}$. Note that $\Delta\mathbf{y}$ is abandoned.

To train the spR2NNs, the set of candidate pairs $\mathcal{P}$ has to be prepared in advance. Algorithm 1 shows the training procedure, where $\mathcal{P}$ is formed by randomly permutating the training set in each training epoch. Note that the tradeoff coefficient α in (3) does not function as a specific ratio for randomly picking up either of the input streams to individually optimize its corresponding loss function, as done in [17]. Instead, it signifies the relative importance between regression and ranking losses and involves in the derivation of models. By implementing a back propagation process from the top output layers down through the whole spR2NNs to adjust all parameters, each parameter in $\mathcal{M}$ during the t-th epoch is updated by gradient descent as:

$$\mathbf{W}^{(t)} \leftarrow \mathbf{W}^{(t-1)} - \eta \frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(t-1)}}, \tag{4a}$$

$$\mathbf{b}^{(t)} \leftarrow \mathbf{b}^{(t-1)} - \eta \frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(t-1)}}, \tag{4b}$$

where η is the learning rate, and $\mathbf{W} \in \{\mathbf{W}_s, \mathbf{W}_g, \mathbf{W}_k\}$ and $\mathbf{b} \in \{\mathbf{b}_s, \mathbf{b}_g, \mathbf{b}_k\}$ in Figure 1. Since the combined training data are not deterministic, $\mathcal{J}$ as well as the Spearman's ρ derived from the training set and the development set, respectively, are fluctuant, but tend to steadily move over training epochs in the long term. Therefore, steps 5 to 12 in Algorithm 1 show the stopping criterion, based on simple moving average of Spearman's ρ, to deal with the circumstance.

The simplest and reasonable way to feed test samples $\mathbf{X}$ into the trained machine is to set $\Delta\mathbf{X}$ to be a matrix that is filled with zeros, as shown in Algorithm 2.

Algorithm 3 The Training Procedure for acR2NNs

Input: The same as those in Algorithm 1.

Output: The model estimate $\hat{\mathcal{M}}$.

1: Run the K-means algorithm on $\mathbf{X}$ to yield n clusters and their respective centers $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n]$.
2: Pick out n samples, $[\mathbf{x}_{a1}, \ldots, \mathbf{x}_{an}]$, from $\mathbf{X}$ that are respectively the closest to the cluster centers.
3: Form input and truth streams in Figure 2 by calculating $\Delta\mathbf{X}_{ai} = \mathbf{X} - \mathbf{X}_{ai}$ and $\Delta\mathbf{y}_{ai} = \mathbf{y} - \mathbf{y}_{ai}$, $i = 1, \ldots, n$.
4: Run the same steps 4 to 16 in Algorithm 1, except that the prediction procedure has to be modified for acR2NNs.

3.2. The Anchor-based Neural Network

Without any sampling process implicated in the training phase, we propose another architecture called the anchor-based regression and ranking neural networks (acR2NNs), as depicted in Figure 2. The philosophy behind acR2NNs is to reinterpret the pairwise ranking problem as a reference-comparing problem. That is, it presumes that, given some fixed references, namely anchors, if we can accurately predict the relative distances between the labels of each sample and the anchors, then the predicted labels of samples will be ranked nearly the same as the ground truth. The training procedure of acR2NNs differs from spR2NNs only in the use of sample pairs. As shown in Algorithm 3, a clustering method, such as K-means, is first performed to derive the most representative samples in the training set, followed by calculating the feature differences to form other input and output streams. Note that, similar to other instance-based learning machines, these anchors have to be reserved with the model parameters during either training or test phases.

4. Experiments and Results

In this section, we analyze the performance of the proposed two instance-based methods on the speech material provided by the Sincerity sub-challenge in ComParE 2016 [3].

4.1. Features and Datasets

Each audio file contains 6,373 features that were extracted by the organizers with the OpenSMILE toolbox [29]. The training and test sets are comprised of 655 and 256 utterances recorded by 22 and 10 speakers, respectively. Since no separate development set was provided, prototyping tests are done with Leave-One-Speaker-Out cross-validation (LOSOCV) on the training set, where the predicted values are disjointly composed of those generated by 22 *different* models. In general, each learning process through each training/test combination in LOSOCV should be treated as an independent event so that the cross-validation estimate of metrics, such as accuracies and precisions, can be derived by average [30]. However, in the prototyping test, the final Spearman's ρ is contributed by 22 batches of predicted values which are highly correlated in the ranking sense. For instance, the ranked values of batch a might be highly affected by those of batches other than a. Moreover, we found that although LOSOCV can help adjust hyper-parameters in neural networks, such as the learning rate η, the regularization parameter λ, the tradeoff coefficient α and the structure of hidden layers, it is difficult to help select the suitable initial weights/bias and the optimal number of training epochs [31]. Therefore, we singled out 6 speakers from the training data as a development set, which satisfies 1) the gender ratio is the same as the test set, 2) the proportion of the sample size to the new training set is nearly consistent with that of the test set to the original training set,

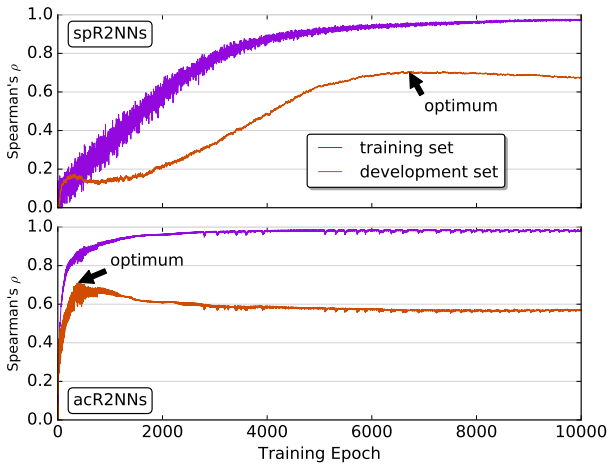

Figure 3: *Spearman's ρ on the training and development sets with respect to the training epoch in spR2NNs and acR2NNs.*

and 3) the experiment results of the development set are similar to the baseline results provided by the organizers [3].

4.2. Training on spR2NNs and acR2NNs

For the sake of convenience, the learning rate η, the regularization parameter λ, the tolerance τ, and the maximum number of training epochs are empirically fixed to 0.1, 1.0, 10^{-5}, and 10^4, respectively. Initial weights are uniformly sampled by the Glorot process that is fit for the rectified linear activation function (RELU) [32]. The adaptive gradient algorithm adopted to update model parameters is AdaGrad, which scales the learning rate by dividing with the square root of accumulated squared gradients [33]. To avoid overfitting, we recorded the evaluation results in each training epoch as shown in Figure 3, where the layer structure of spR2NNs is [0, 1]@128, which means that the number of separate layers is 0 while the number of parameter-shared layers is 1 with 128 hidden nodes, and the layer structure of acR2NNs is [0, 1]@256 with only 1 anchor sample. We can see that 1) the optimization of $\mathcal{J}$ on the training set indirectly implies the optimization of Spearman's ρ, 2) owing to the sampling process, the curve of spR2NNs is much more fluctuant than that of acR2NNs, and this might be the reason why the training process of spR2NNs takes a longer time to converge.

4.3. Results

We compare the results on the development set with various tradeoff coefficients in Figure 4. The layer structures of spR2NNs and acR2NNs are the same as those in Figure 3 except for the number of hidden nodes. We can see that 1) the lower α, which gives a higher weight to the pairwise ranking loss in (3), does not necessarily guarantee a higher Spearman's ρ, 2) acR2NNs outperforms spR2NNs, and all the results are better than the baseline expressed by the black dashed line, and 3) spR2NNs with 256 hidden nodes does not perform well, perhaps because the number of training epochs with higher model complexity is inadequate to reach convergence.

Figure 5 shows the results on the development set with various settings of the layer structure and number of anchor samples. Note that the numbers of hidden notes and the tradeoff coefficients used in spR2NNs and acR2NNs are based on the best results in Figure 4. The reason why the acR2NNs with larger numbers of anchor samples as well as the more complex layer structures did not reveal their theoretical learning power

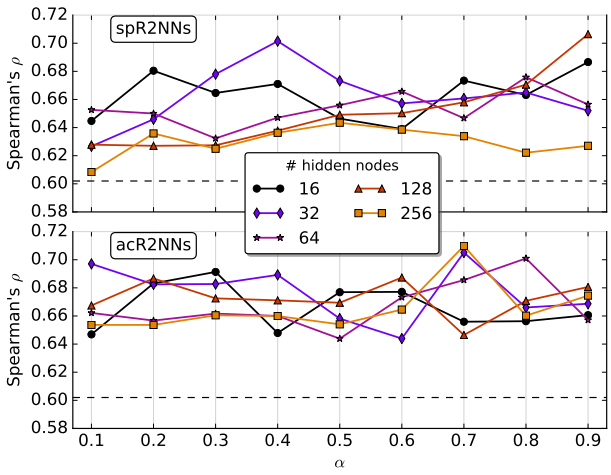

Figure 4: *Spearman's ρ derived on the development set with respect to various tradeoff coefficients α and numbers of hidden nodes in spR2NNs and acR2NNs.*

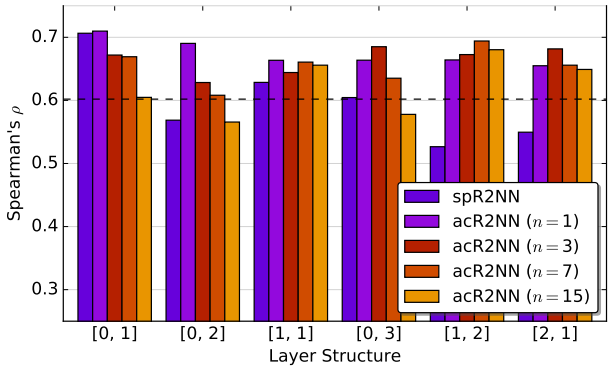

Figure 5: *Spearman's ρ derived on the development set with respect to various settings of the layer structure and numbers of anchor samples in spR2NNs and acR2NNs.*

might lie in the shortage of training samples.

Table 1 shows the final results on the test data provided by the organizers. The performance was obtained from a model trained on the original training set with the optimal parameters determined in the development phase. Note that the stopping criterion in Algrithm 1 (cf. steps 5-12) was adopted due to lack of the development data while training spR2NNs and acR2NNs. We can see that our proposed methods outperform the baseline by relative improvements of 6.8 % and 27.1 % in the test and development sets, respectively.

Table 1: *Spearman's ρ derived on the development and test sets with respect to the baseline and our proposed methods.*

Method	LOSOCV	Devel.	Test
Baseline (SVR, $C = 10^{-4}$)	.474	.602	.602
spR2NNs ([0, 1]@128)	.499	.706	**.629**
acR2NNs ([0, 1]@256)	.477	**.710**	.599

5. Conclusions

In this paper, we have proposed a framework based on simultaneous minimization of regression and ranking losses for the task of automatic sincerity evaluation. The framework has been realized by two kinds of neural network-based architectures. The experiment results demonstrated the potential of the framework.

6. References

[1] J. Eriksson, "Self-expression, expressiveness, and sincerity," *Acta Analytica*, vol. 25, no. 1, pp. 71–79, 2010.

[2] E. S. Hinchman, "Assertion, sincerity, and knowledge," *Noûs*, vol. 47, no. 4, pp. 613–646, 2013.

[3] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The INTERSPEECH 2016 computational paralinguistics challenge: deception, sincerity & native language," in *Proc. Interspeech Conf.*, 2016.

[4] H. S. Cheang and M. D. Pell, "Recognizing sarcasm without language: A cross-linguistic study of English and Cantonese," *Pragmatics & Cognition*, vol. 19, no. 2, pp. 203–223, 2011.

[5] J. Tepperman, D. R. Traum, and S. Narayanan, ""Yeah right": Sarcasm recognition for spoken dialogue systems," in *Proc. Interspeech Conf.*, 2006.

[6] R. V. Hogg, J. W. McKean, and A. T. Craig, *Introduction to Mathematical Statistics*, 7th ed. Pearson College Division, 2013.

[7] N. Itoh, G. Kurata, R. Tachibana, and M. Nishimura, "A metric for evaluating speech recognizer output based on human-perception model," in *Proc. Interspeech Conf.*, 2015, pp. 1285–1288.

[8] J. Gibson, N. Malandrakis, F. Romero, D. C. Atkins, and S. S. Narayanan, "Predicting therapist empathy in motivational interviews using language features inspired by psycholinguistic norms," in *Proc. Interspeech Conf.*, 2015, pp. 1947–1951.

[9] B. Schuller, S. Steidl, A. Batliner, S. Hantke, F. Hönig, J. R. Orozco-Arroyave, E. Nöth, Y. Zhang, and F. Weninger, "The INTERSPEECH 2015 computational paralinguistics challenge: nativeness, Parkinson's & eating condition," in *Proc. Interspeech Conf.*, 2015.

[10] M. M. Mukaka, "A guide to appropriate use of correlation coefficient in medical research," *Malawi Medical Journal*, vol. 24, no. 3, pp. 69–71, 2012.

[11] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. Wiley-Interscience, 2000.

[12] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[13] K. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[14] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," in *Proc. Int. Conf. Mach. Learning (ICML)*, 2007, pp. 129–136.

[15] H. Valizadegan, R. Jin, R. Zhang, and J. Mao, "Learning to rank by optimizing NDCG measure," in *Proc. Conf. Neural Inform. Process. Syst. (NIPS)*, 2009, pp. 1883–1891.

[16] H. Pareek and P. Ravikumar, "A representation theory for ranking functions," in *Proc. Conf. Neural Inform. Process. Syst. (NIPS)*, 2014, pp. 361–369.

[17] D. Sculley, "Combined regression and ranking," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining (SIGKDD)*, 2010, pp. 979–988.

[18] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network," *IEEE Trans. Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.

[19] Y. Hamamoto, S. Uchimura, T. Kanaoka, and S. Tomita, "Evaluation of artificial neural network classifiers in small sample size situations," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 1993, pp. 1731–1735.

[20] S. Ingrassia and I. Morlini, "Neural network modeling for small datasets," *Technometrics*, vol. 47, no. 3, pp. 297–311, 2005.

[21] T. Joachims, "Optimizing search engines using clickthrough data," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining (SIGKDD)*, 2002, pp. 133–142.

[22] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. Int. Conf. Mach. Learning (ICML)*, 2008, pp. 160–167.

[23] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. Int. Conf. Mach. Learning (ICML)*, 2011.

[24] D. Chen and B. Mak, "Multi-task learning of deep neural networks for low-resource speech recognition," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 23, no. 7, pp. 1172–1183, 2015.

[25] X. Lu, F. Wu, X. Li, Y. Zhang, W. Lu, D. Wang, and Y. Zhuang, "Learning multimodal neural network with ranking examples," in *Proc. ACM Int. Conf. Multimedia (ACMMM)*, 2014, pp. 985–988.

[26] X. Shu, G.-J. Qi, J. Tang, and J. Wang, "Weakly-shared deep transfer networks for heterogeneous-domain knowledge propagation," in *Proc. ACM Int. Conf. Multimedia (ACMMM)*, 2015, pp. 35–44.

[27] D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou, "Sentiment embeddings with applications to sentiment analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 496–509, 2015.

[28] R. Xia and Y. Liu, "A multi-task learning framework for emotion recognition using 2D continuous space," *IEEE Trans. Affective Comput.*, 2015.

[29] F. Eyben, M. Wöllmer, and B. W. Schuller, "OpenSMILE: the munich versatile and fast open-source audio feature extractor," in *Proc. ACM Int. Conf. Multimedia (ACMMM)*, 2010, pp. 1459–1462.

[30] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 1995, pp. 1137–1145.

[31] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*, 2012, pp. 437–478.

[32] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, 2010, pp. 249–256.

[33] J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learning Research*, vol. 12, pp. 2121–2159, 2011.