Dynamic Facial Stress Recognition in Temporal Convolutional Network

Sidong Feng[0000−0001−7740−0377]

Research School of Computer Science, Australian National University, Canberra Australia
u6063820@anu.edu.au

Abstract. Stress is a major problem that infiltrates our society in countless ways. We cannot eliminate stress, but can recognize stress and manage it. Automatically recognizing stress through facial expressions has been extensively studied in the past decades. Recent research indicates that certain architectures can reach state-of-the-art accuracy in stress recognition. However, they recognise facial stress in view of static expressions, while only a few papers identify the fundamental limitations of static facial expression. This paper adapts ANUStressDB database in dynamic and develops a Temporal Convolutional Network to recognize continuous facial stress problem. We further apply Bimodal Distribution Removal to improve our result. The experimental results show that our system achieves 67.56% classification accuracy.

Keywords: Stress Recognition · Temporal Convolutional Networks · Bimodal Distribution Removal.

1 Introduction

Stress is defined as a state of mental or emotional strain. It is important to recognize stress so that it can be effectively managed. Automatically recognizing human's emotions through facial expressions (a.k.a. facial expression recognition or FER) has emerged as a key problem of human-computer interaction and psycho-physiology analysis [2]. We used ANUStressDB [10] to identify facial stress. In this problem, stress is identified based upon the signals acquired in real time from contact-less sensors such as RGB and thermal modalities. Given a time sequence signals, the goal is to simultaneously segment every emotion in time and classify each constituent segment as stress or not.

Deep learning practitioners commonly regard recurrent architecture as the default starting point for sequence modelling tasks [8]. In past decades Long Short-Term Memory (LSTM) [9] and Gated Recurrent Unit (GRU) [5] occupy time sequence problems. However, they take too long to process, because they read and interpret the time sequence one frame at a time, the neural network must wait to process the next frame until the current frame processing is completed. This means that RNNs cannot take advantage of massive parallel processing (MPP) [17] in the same way the CNNs can. Temporal Convolutional Net (TCN) [3] solve this problem.

By mid-2017, Bai et al. published a new architecture called TCN, which distills the best practices in convolutional (e.g. Causal Convolutions, Dilated Convolutions) network design into a simple architecture. It outperforms canonical recurrent networks such as LSTMs across a diverse range of sequence modeling tasks. Our task is thus to evaluate the performance on real time facial stress recognition problem using TCN.

Stress varies among individuals. Some people are naturally more sensitive and reactive to stress. Different kinds of stress have different symptoms and physiological signs [18]. It is a subjective topic, data could be fuzzy and vague, leading outliers in database. A number of methods for cleaning up noisy data has been proposed, such as Least Median Squares (LMS) by Rousseeuw [16] and Least Trimmed Squares (LTS) by Alfons [1]. These methods perform well on synthetic noisy data, but not well on real world data. Since our data is collected in real world, a more reliable method is required. Bimodal Distribution Removal (BDR) by Slade et al. [20] is a well-known outliers removal method, proved to perform well on both added artificial outliers and real noisy data [20].

Our contributions can be summarized below:

- We propose a convolutional based technique to automatically recognize temporal dynamic facial stress problem.
- We analyze the feasibility of BDR technique on the basis of ANUStressDB, which contains added artificial outlier and real world noisy data.
- We identify the fundamental limitations of static processing characteristics of stress recognition problem in previous work and propose to exploit continuity of stress to address these limitations.

2 The proposed system

The proposed system is shown in Fig. 1. It involves four procedures. 1) prepare ANUStressDB data; 2) apply techniques like data augmentation, dimensionality reduction and data scaling to help manage the data; 3) train the model; 4) apply BDR on pretrained model for further improvement. A halting condition is provided to decide termination.

2.1 Data Preparation and Preprocessing

In this paper, we use ANUStressDB as benchmark dataset to evaluate our model. The dataset involves 24 participants. Instructors played a film with a collection of negative and positive clips as stress stimulator. The clips are separated by displaying few seconds blank screen in between the clips to neutralize the participants' emotion before playing the next clip. Two cameras are working at 30 frames per second to capture thermal and RGB modalities. Then, facial features are extracted by using Linear Spectral Clustering (LSC) [11] and Local Binary Patterns (LBP) [14], respectively. As a result, we extract 36 features for each frame. In the ground truth data, we assign the patterns in the time series as stressed or not when the label of the clip is stressed or not.

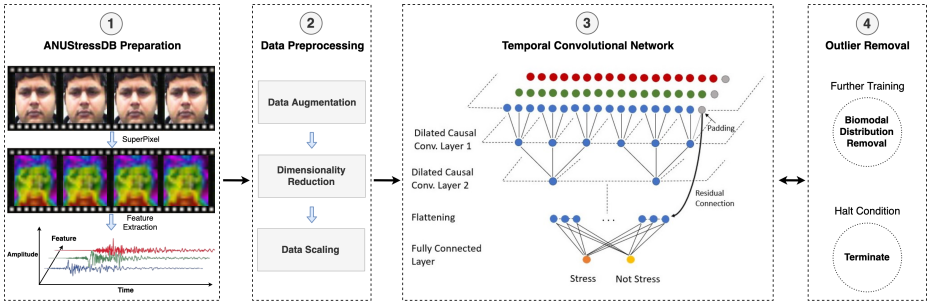

Fig. 1. Illustration of the proposed system.

Data Augmentation We have 24 participants and 12 clips, results in 288 time series, whereas too small to train a deep learning neural network. Hence, data augmentation is applied. We split time series into fixed length sub time series. As a trade off, sub time series may lose some information on the origin time series. To balance the quality and quantity, length is set to 10. We round length to encounter aliquant time series. Although we carefully select the time series length, we still suffer from noise. For example, a 17 second time series would be split into 0-10, 7-17 sub time series. If there is no emotional disclosure in the first 10 seconds, 0-10 will be an artificial noisy data. Whereas, BDR in Section 2.3 can solve this implicit problem.

Dimensionality Reduction As discussed in Section 2.1, a 10 seconds time series involves 300×36 features. Training model on high-dimensional data greatly increases the number of weights, making the training infeasible [21]. We reduce the data in two approaches. First, in time series dimension, we observe that the difference between each frame is small. Therefore, we take the average value of 30 frames (1 second) as one time sequence. Second, we reduce feature dimension by feature selection. We remove irrelevant features. By observation some features are slightly different. Therefore, we remove one of these features so we're left with only features with distinct values. Thus, each time series reduces to 10×16.

Data Scaling Standardization [6] (i.e., rescaling with 0 mean and unit variance) that changes the values of numeric columns in dataset to a common scale([-1,1]) is applied to improve neural network stability and training efficiency.

2.2 Temporal Convolutional Networks

This architecture is informed by convolutional architecture for sequential data (e.g., WaveNet [13]), but is deliberately kept simple. It combines the best practices of modern convolutional architectures, such as Dilated Causal Convolutions, Residual Connections. There are two major characteristics of TCN. 1) the convolutions in the architecture are causal, meaning that there is no information

leakage from future to past; 2) the architecture can take a sequence of any length and map it to an output sequence of the same length, just as with an RNN.

Dilated Causal Convolutions As mentioned, the TCN is based on two principles. One is the convolutions in the architecture are causal. To accomplish this point, the TCN uses causal convolutions. Causal simply means a filter at time step t can only see inputs that are no later than t. However, a major problem of this design is when the history is long. This is because, a causal convolution needs to look back at history with size linear to the time. For example, to predict output at time 1000, network needs to look back 1000 previous inputs. It requires an extremely deep network, which is inefficient and infeasible.

In the previous work WaveNet by Oord et al. [13], they employ dilated convolutions to allow the receptive field to increase exponentially [22]. Receptive field is the implicit area captured on the initial input by each input to the next layer. In TCN, it makes use of dilated convolution which is just a convolution applied to input with defined gaps. The kernel size k is to filter and the dilation factor d is to control the gaps. In common, dilation factor grows exponentially (i.e., d=2^i at depth i). This ensures that filter can hit each input within the effective history, while also allowing for an extremely large effective history using deep networks [3]. Takes the advantages of both techniques, the integration of causal and dilated is able to conquer long history problem.

Fully Connection To achieve another principle of TCN, we use a fully convolutional network (FCN) established by Long et al [12]. Fully connection layers is added after the output of the TCN to address binary classification problem. Determining a certain number of hidden layers and neurons is crucial and difficult in the research community. For our problem, we perform several trials on different numbers of hidden layers and neurons, finding that the best performance appears when there are no hidden layers after the TCN layers. Thus, we apply one fully connected layer at the end. An illustration is provided in Fig 1.

2.3 Bimodal Distribution Removal

The outliers in training set will have larger errors relative to the rest of the training set. First, we calculate the errors of each training pattern by using cross entropy. Then, calculate the mean of errors ($\overline{\Delta}_{ts}$). We define the error greater than $\overline{\Delta}_{ts}$ as high error peak and calculate the mean and standard deviation of errors of high error peak ($\overline{\Delta}_{ss}$) (σ_{ss}). Since $\overline{\Delta}_{ss}$ will be heavily influenced by outliers, it will be relatively high. It is possible to decide which patterns to permanently remove from the set. If the error follows the pattern:

$$error \geq \overline{\Delta}_{ss} + \alpha\sigma_{ss} \tag{1}$$

α is to control how many outliers need to be removed. Since our dataset is not large enough, we decide to set the removal factor α to 1, so the least outliers are

Table 1. Performance on different models

	Training (%)	Validation (%)	Testing (%)
Epoch 300	73.95	59.56	60.98
Epoch 500	81.21	55.73	57.92
Early Stop	69.06	64.30	67.56
Early Stop + BDR	84.61	54.71	53.45
Sharma (GA-SVM) [19]	-	-	86
Irani (SVM) [10]	-	-	89
Prasetio (CNN) [15]	-	-	95.9

Table 2. Performance on different models

	Testing (%)
Early Stop	67.56
Early Stop + BDR	53.45
Sharma (GA-SVM) [19]	86
Irani (SVM) [10]	89
Prasetio (CNN) [15]	95.9

removed.. To avoid removing all the data, a halting condition is set by variance υ_{ts} and the size of the remaining set. Low variance means the network is well trained and small size of training set means the network could easily overfit.

3 Experiments

In this section, we begin by discussing our hyperparameter settings. Then, we evaluate and analyze on the result of the system in detail. Finally, we discuss the comparison between our model and previous works. A synopsis of the result is shown in Table 2.

Hyperparameter Settings Table 3 lists the hyperparameters we used when applying the TCN. The most crucial factor for the TCN is k. They determine whether the receptive field is large enough to capture the sufficient context to predict. As previous work suggested, larger kernel size k helps network to converge faster. By several trials, k=7 performs best.

As discussed in Section 2.3, thresholds on variance υ_{ts} and size of training dataset are defined in Table 3. Early Stop [4] is applied to prevent overtraining. The stopping point depends on either validation accuracy or validation loss. We decide to use validation accuracy as the driving metric since it is the most vital factor in our problem. Since accuracy oscillates, we set patience value to 30 to determine whether it reaches the end or just floating. All threshold values are carefully selected through manually check.

Model analysis and discussion As we can see, the testing accuracy at early stop is 6.58% and 9.64% higher than training model at 300 epochs and 500

Table 3. Optimal hyperparameter settings of TCN and BDR

TCN		BDR	
Input features	16	**Further train epochs**	50
Sequence length	10	α	1
Kernel size k	7	**Variance** υ_{ts}	0.01
Hidden neurons	[10,10]	**Min train size**	1000
Learning rate	1e-4		

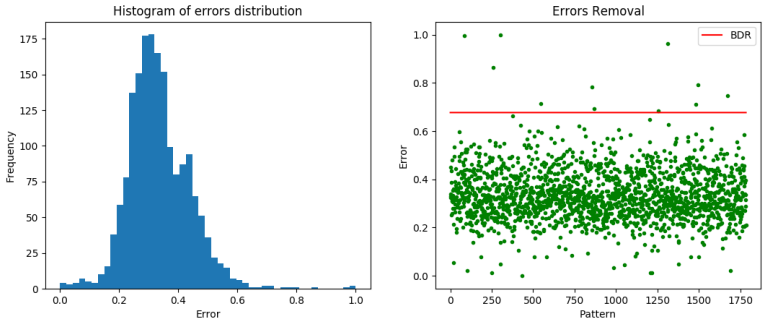

Fig. 2. (a) is the histogram of normalized error distribution at early stop checkpoint. (b) is the diagram of normalized error distribution of each patterns, green scatter point represents as each error. The line represents as the BDR line where the pattern above the line considered as outlier and will be removed.

epochs. Therefore, early stop is an effective technique to use. We assume the model using this technique as the pretrained network.

As mentioned in Section 2.3, BDR can clean up noisy data. Hence it may help improve our network. To test the usefulness of BDR, we implement BDR on pretrained network. As we observe in Table 2, BDR boosts 15.55% on training accuracy. Contrastly, validation accuracy and testing accuracy decreases 9.59%, 14.11% respectively. It is possibly overfit. There are two reasons for this problem. In Fig 2 (a), we can observe that the errors distribution after pretraining is not bimodal distribution. Thus, the algorithm to calculate high error peak is not accurate and precise any more. Another reason might be outliers can be legitimate data, representing an accurate observation of a rare case. Removal decreases generalization ability in neural network. Thus, BDR is not an effective approach for our model.

Comparison with previous works We further compare our best result 67.56% appears at Early Stop, with the previous works by Sharma et al. [19], Irani et al. [10] and Prasetio et al. [15]. From Table 2 we can see that previous works outperform our model. Next, we discuss possible reasons for the difference.

First, instead of using deep learning, Sharma and Irani use SVM as classifier. SVMs are originally designed for binary classification. On the basis of our prob-

lem, SVM has dominant position. Also, our dataset contains a small amount of training data. SVMs have advantages to predict in less training data. In such a case, SVM might be better than our model.

Apart from the benefits of using SVM, the model architecture is different. As proposed in Sharma work, they use GA [7] for feature selection. On the contrary, we use feature selection in statistical way (removing features if the values of this feature are slightly different). A dropped feature in statistical approach can drastically change the result as the slightly different values might transform and magnify to a major factor and drives the classification. Thus, manual observation might not be a scientific algorithm for feature selection. In contrast, GA is a proven advanced algorithm for feature selection which is more appropriate. As proposed in Irani work, they use fusion model which uses three separate SVMs, one for RGB, one for thermal and the last one learning from the combination. The complexity of RGB and thermal modalities might be different. Thus, applying modality in different model structures and hyperparameters might leads to better result. Hence, their approach is better than us. As proposed in Prasetio work, they take advantages of Sharma and Irani essence. Rather than using GA to reduce dimension as proposed in Sharma's work, they use feature extraction. Rather than fusing RGB and thermal modalities, they fuse Eye, Nose and Mouth, which is the intuition of Irani's work. In conclusion, the previous model architecture is more effective.

However, there is a fundamental limitation on training input in the previous works. Instead of time series data, they use frame data. Each frame considers as a pattern and labels as stress or not. Then randomly select some patterns into training set. However, as observation many patterns in sequence have minimal differences, especially in one film. It is highly possible that the patterns in the testing set are mostly the same in the training set. For example, assume two patterns A and B are in sequence. The difference between them is slight. After shuffling, A is divided into training set and B into testing set. This causes a crucial problem. As long as the network can classify A, it can classify B. In other word, network can simply memorize patterns, not learn, and still performs well.

4 Conclusion

In this paper, we proposed a Temporal Convolutional Network (TCN), whose core is a dilated and causal convolution method for facial expression recognition. Rather than using the canonical recurrent neural networks such as LSTMs and GRUs, we have presented convolutional neural network which can also be used in a way of solving sequence modeling tasks. The type of input data has a tremendous impact on the results. Our experiments on ANUStessDB confirm this claim, showing that the results by using static input outperform that by dynamic input. We intend to extend our work by applying Bimodal Distribution Removal (BDR) method to remove noise in artificial and real-world data. Contrastly, BDR worsens our neural network. The improvement on outlier removal suggests our

proposed system has the potential to improve the performance of other methods, which will be investigated in future work.

References

1. Alfons, A., Croux, C., Gelper, S.: Sparse least trimmed squares regression for analyzing high-dimensional large data sets. Ann. Appl. Stat. **7**(1), 226–248 (03 2013)
2. Andreassi, J.L.: Psychophysiology : human behavior and physiological response. Oxford University Press New York (1980)
3. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv:1803.01271 (2018)
4. Caruana, R., Lawrence, S., Giles, C.L.: Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In: Advances in neural information processing systems. pp. 402–408 (2001)
5. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
6. Cowan, R.: High technology and the economics of standardization (1992)
7. Gen, M., Lin, L.: Genetic algorithms. Wiley Encyclopedia of Computer Science and Engineering pp. 1–15 (2007)
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Irani, R., Nasrollahi, K., Dhall, A., Moeslund, T.B., Gedeon, T.: Thermal superpixels for bimodal stress recognition. In: IPTA. pp. 1–6. IEEE (2016)
11. Li, Z., Chen, J.: Superpixel segmentation using linear spectral clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1356–1363 (2015)
12. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
13. Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. arXiv:1609.03499 (2016)
14. Pietikäinen, M., Hadid, A., Zhao, G., Ahonen, T.: Local binary patterns for still images. In: Computer vision using local binary patterns, pp. 13–47. Springer (2011)
15. Prasetio, B.H., Tamura, H., Tanno, K.: The facial stress recognition based on multihistogram features and convolutional neural network. In: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 881–887. IEEE (2018)
16. Rousseeuw, P.J.: Least median of squares regression. Journal of the American statistical association **79**(388), 871–880 (1984)
17. Sankaradas, M., Jakkula, V., Cadambi, S., Chakradhar, S., Durdanovic, I., Cosatto, E., Graf, H.P.: A massively parallel coprocessor for convolutional neural networks. In: 2009 20th IEEE International Conference on Application-specific Systems, Architectures and Processors. pp. 53–60. IEEE (2009)
18. Schneiderman, N., Ironson, G., Siegel, S.D.: Stress and health: Psychological, behavioral, and biological determinants. Annual Review of Clinical Psychology **1**(1), 607–628 (2005)

19. Sharma, N., Dhall, A., Gedeon, T., Goecke, R.: Thermal spatio-temporal data for stress recognition. EURASIP Journal on Image and Video Processing **2014**(1), 28 (2014)
20. Slade, P., Gedeon, T.D.: Bimodal distribution removal. In: International Workshop on Artificial Neural Networks. pp. 249–254. Springer (1993)
21. Wójcik, P.I., Kurdziel, M.: Training neural networks on high-dimensional data using random projection. Pattern Analysis and Applications **22**(3), 1221–1231 (2019)
22. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122 (2015)