Facial Expression Analysis based on Transfer Learning and Feature Analysis

Cen Dai[1]

Research School of Comupter Science, Australian National University
u6266023@anu.edu.au

Abstract. Facial expression analysis based on static images is an active and interesting topic in deep learning area. Compared with images generated in labs, images caputured in wild environment are more natural but more difficult to be handled automatically. This research uses a static image dataset SFEW and transfer learning based on fine tuned ResNet18 network. The model has around 90% testing accuracy. A series of feature analysis is carried out with respect to their significance on the facial expression classification result. Magnitude analysis based on neural network's weight matrix is used to test feature's significance, and functional analysis based on the angle between multi-dimensional vectors is used to evaluate their distinctiveness. Analysis by pruning different features are completed for validation. No significant feature is observed and features are mutually independent. The result also suggests that the model performce well when only 10% features are used.

Keywords: Facial expression analysis · Feature analysis · Magnitude analysis · Functional analysis · Transfer learning.

1 Introduction

Facial expression is generated by the motions of facial muscles and it is the bridge between human's intrinsic emotion to extrinsic expression. Over the past few decades, facial expression analysis has drawn much attention among the community and has been widely applied into many areas, i.e., video-conferencing, virtual reality, customer/ user satisfication survey and lie detection, etc,. [6, 8, 9]

Facial expression analysis can be either video based or image based. It has been proved that video based analysis is more robust than image based method due to its dynamic nature. However, video based analysis is not alway applicable since sometimes the temporal sequence data is not available and it is computationally expensive because the volume of a video is larger than a static image. Hence, image based analysis is still worthy for study.

Some image databases are constructed for the purpose of image based facial expression analysis, e.g. the JAFFE database and the PIE database. Good classification accuracy is obtained over these databases using machine learning techniques [1]. Nevertheless, the images in these databases are collected in rigid lab environments which are quite different from the real-world situations. In this case, a database that consists facial expression images produced under real-world circumstances is essential to improve facial expression anlysis techniques.

Static Facial Expressions in the Wild (SFEW) [1] is a database formed by images extracted from films, which contains facial expressions that are close to real-world environments. This database allows us to develop techniques to handle real-world condition.

In the previous study [1], local phase quantisation (LPQ) and pyramid of histogram of oriented gradients (PHOG) descriptors are extracted from the images in SFEW database. These descriptors are used for a 7 expression classification task and the labels assigned to each image can be used as ground truth in training. Unfortunately, the average classification accuracy is around 19% which is not ideal [1].

To improve the classification accuracy, we use image data from SFEW rather than PCA descriptors. A fine-tuned pretrained ResNet18 is used to extract features from the images. Then, feature analysis techniques are applied to evaluate their impacts on the classification result and to prune the network. This paper is arranged as below,

- A 7 label facial expression classification task is carried out based on a pretrained ResNet18 model, in which the output layer is replaced by a fully connected classifier.
- Magnitude measures are applied to analyse the contribution of extracted features to ouputs.
- Features' similairty is evaluated by functional analysis using angle between multi-dimensional vectors.
- Based on results from above experiments, features are pruned and testing accuracy is analysed.

2 Method

2.1 Fully connected neural network

Firstly, a simple fully connected neural network is trained by back-propogation. The inputs are LPQ and PHOG components provided by previous study. They are normalised using z-score before being fed into the

neural network. The network's topology is 10-50-7, decided by 5-fold cross-validation. All the feature analysis techniques used in this paper are implemented, the results are summarised as reference for following experiments.

2.2 Transfer Learning with ResNet18 and data preprocessing

Transfer learning means using model pretrained by larger related dataset to extract features, then using a smaller target dataset to fine tune and test the model. In this paper, ResNet18 model pretrained on ImageNet is chosen as feature extractor [5]. After that, the last fully connected layer (feature-output) is replaced by a three layer (feature-hidden-output) fully connected classifier. Finally, the last two convolutional layer and the fully connected classifier are fine tuned by SFEW images.

As the raw images contain lots of irrelevant information (e.g. background scenes, highlights and shades), faces are detected and extracted using facenet-pytorch module [10]. For each image, only the face with highest detect probability is croped and resized to 224×224 based on ImageNet standard. If no face is detected, the central 224×224 area is cut out with the assumption that the main face occurs in the centre. The face images are normalised with ImageNet regularised mean and strandard deviation values.

Since we use transfer learning and a pretrained model, the main hyperparameters to tune are learning rate and the number of hidden neurons in fully conneted classifier. These hyperparameters are determined by 5-fold cross-validation.

For the purpose of repetitive experiment, 15 ResNet models with the seleted hyperparameters are trained independently. In this stage, 15% data is kept unseen for testing, the rest is split into training set and validating set with ratio 80:20. The model with highest testing accuracy during each training progress is selected. A second training process with smaller learning rate is applied to these models which are used for further analysis.

2.3 Magnitude measures

Wong, Gedeon and Taggart [11] brought up a magnitude measure, noted as P_{ij} to evaluate the contribution of an input i to a hidden neuron j in fully connected network.

$$P_{ij} = \frac{|W_{ij}|}{\Sigma_{p=1}^{n_i}|W_{pj}|} \tag{1}$$

Where W_{ij} is the weight, p is the index of input and n_i is the number of inputs.

This measure is extended by Gedeon [3] to describe the contribution of an input i to an output k, Q_{jk}, as

$$Q_{ik} = \Sigma_{r=1}^{n_h}(P_{ir} \times P_{jk}) \tag{2}$$

P_{jk} is the contribution of a hidden neuron j to an output k.

This approach fixes the cancellation caused by summing positive and negative weights so that the magnitude of the contribution can be retained.

In this study, measure described in equation 2 is applied to the features extracted by ResNet as they are inputs of fully connected classifier. The contribution of a feature to the classification result is represented by the sum of its contribution to each output. The magnitude analysis result is synthesised from multiple experiments.

2.4 Functional measures

In 1991, A distinctiveness analysis technique is introduced to examine the functional differences between hidden neurons using multi-dimensional vectors [2, 4]. It is then adapted to different models to calculate the functional difference between inputs [3]. As described in equation 3, weights of a feature to all hidden neurons form a multi-dimensional vector and the distinctiveness between two features is defined as the angle between two vectors.

$$\theta_{ij} = cos^{-1}\left(\frac{pattern_i(h) \cdot pattern_j(h)}{|pattern_i(h)| \cdot |pattern_j(h)|}\right) \tag{3}$$

Where $pattern_i(h)$ is the weight vector of feature i to all hidden neurons.

In this study, the average angle of a feature to others is used to represent its functionality. And the distinctiveness analysis results of multiple experiments are averaged to get the final result.

2.5 Feature pruning analysis

In this experiment, features are pruned based on magnitude measures and functional measures respectively. The models are also pruned randomly with the same ratio. The test accuracy of these three experiment are compared to validate previous analysis result.

3 Results and Discussion

3.1 Fully connected classifier

The simple fully connected classifier (10-50-7) has test accuracy 26.857% which is slightly higher than the accuracy achieved using SVM in the originial article [1]. The results of all the three sorts of input analysis is shown in Table 1 and Figure 1. We can see that no significant input is found as in all the results, each input has similar measure value or behavior. Therefore an arbitary pair of inputs can be eliminated to simplify the network without significant loss in accuracy.

Table 1. Average magnitude analysis result for each input

input	1	2	3	4	5	6	7	8	9	10
magnitude	0.6939	0.7644	0.7312	0.7271	0.6381	0.6700	0.6923	0.7060	0.6674	0.7096

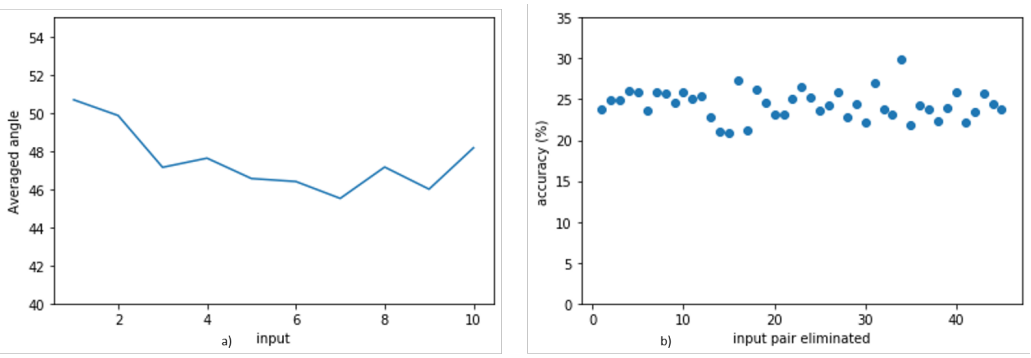

Fig. 1. a) Functional analysis result represented by average angles (°) for each input. b) Validating accuracy when inputs are eliminated in pairs.

3.2 Hyperparameter of the network

As discussed in section 2.2, the main hyperparameters that should be tuned are learning rate and the number of hidden neurons in fully connected layers.

Cross-validation is applied to networks with learning rate = [0.005,0.001,0.0005] and the number of hidden neurons = [100,200,300]. The number of epoch is set to 50. The averaged validation accuracy for all combinations of learning rate and number of hidden neurons are listed in table 2.

Table 2. Validating accuracy (%) with different learning rate and numbers of hidden neurons

		# of hidden neurons		
		100	200	300
lr	0.005	51.2593	52.1482	49.6296
	0.001	55.7037	52.4444	54.3704
	0.0005	54.5185	52.7407	55.1111

It is shown that validating accuracy doesn't vary greatly with hyperparameters ($mean = 53.1029, std = 3.8884$). Hence, learning rate is set to 0.005 as it searches a wider range for optimal solutions and number of hiden neurons is 100 to make the network simpler. Also, we can see that the accuracy using transfer learning is much higher than both the fully connected classifier (26.877%) and the SVM using PCA descriptors (19%), and is comparable to models based on deep learning techniques ($\approx 50\%$) [7].

As mentioned in section 2.2, 15 ResNet models are trained using the selected hyperparameters. Then they are trained again but with a much smaller learning rate, being 0.001 for 50 more epochs. After entire training process is finished, the testing accuracy of the 15 models is listed in table 3. We can see the accuracy is significantly increased by the second training.

Table 3. Testing accuracy (%) for each model ($mean = 90.2614, std = 2.5530$)

index	1	2	3	4	5	6	7	8
accuracy	87.2549	89.2157	85.2941	92.1569	92.1569	91.1765	91.1765	87.2549
index	9	10	11	12	13	14	15	
accuracy	95.0980	93.1373	91.1765	89.2157	91.1765	88.2353	90.1961	

3.3 Magnitude analysis result

Magnitude analysis is applied to weight matrices from all 15 ResNet models. The average magnitude for each feature is shown in fig 2, in which magnitude of 348 out of 512 features falls into the range ($mean \pm std$) . The values and metadata of the magnitude implies the feature significance's fluctuation is not very large. In this case, no significant features can be found in the network.

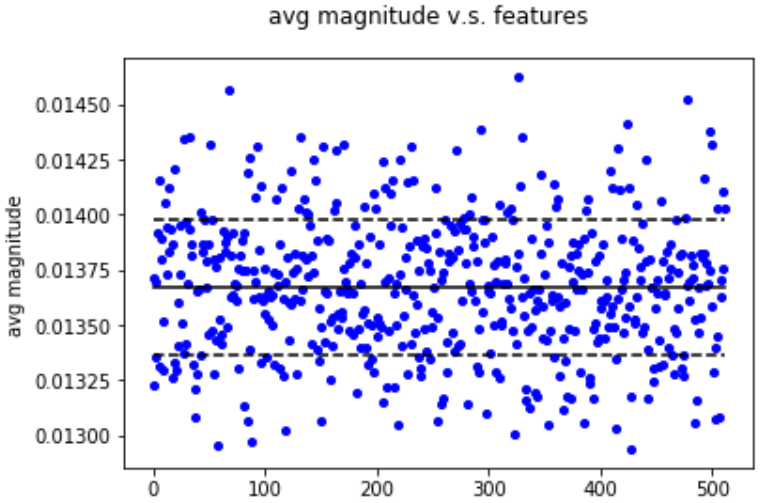

Fig. 2. Average magnitude for all features ($mean = 0.0137, std = 0.0003$). From top to bottom, the three horizontal lines are $mean + std, mean, mean - std$, respectively

3.4 Functional analysis result

Distinctiveness of one feature to other features are estimated using the angle between multi-dimensional vectors. Among all feature pairs, minimum angle is 40.7991° and maximum angle is 51.1359°. Average results of the 15 models are presented in fig 3 in which magnitude of 352 out of 512 features falls into the range ($mean \pm std$). All features are independent.

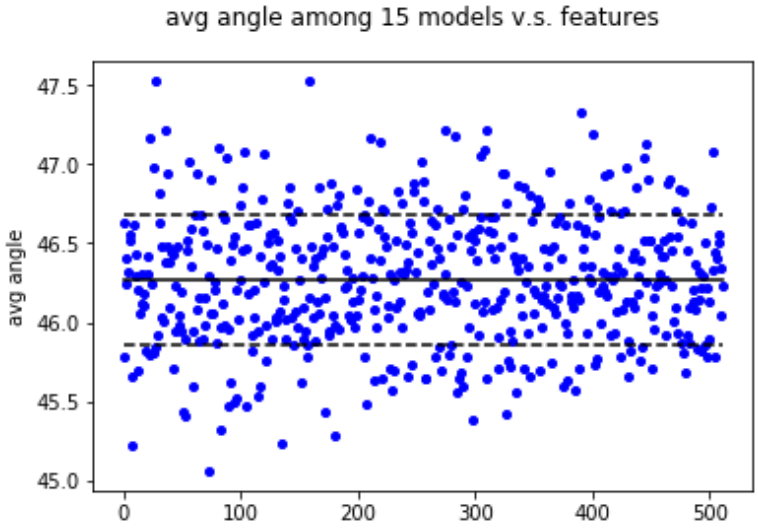

Fig. 3. Average angle for all features ($mean = 46.2731°, std = 0.4104°$). From top to bottom, the three horizontal lines are $mean + std, mean, mean - std$, respectively

.

3.5 Feature pruning analysis result

Three different baselines are used for pruning features based on results from above experiments. 1) Features with magnitude smaller than $mean - std = 0.0134$. In this experiment, 87 ($\approx 16.9\%$) features are pruned. 2) Features with angle smaller than $mean - std = 45.8627°$ are pruned. Therefore, 81 ($\approx 15.8\%$) features are pruned. 3) 77 (15%) random features are pruned. The original testing accuracy without any pruning is used for comparison purposes. The same as before, all the 15 models are pruned and average testing accuracy is collected for each pruning baseline. The result is shown in fig 4.

From the figure, it can be found the pruning result based on magnitude analysis and functional analysis is close to the result of random pruning, which backs the conclusion from previous experiments. Furthermore, all the pruning results are close to the original testing accuracy. It indicates that the accuracy loss caused by pruning 15% features can be mitigated by the rest.

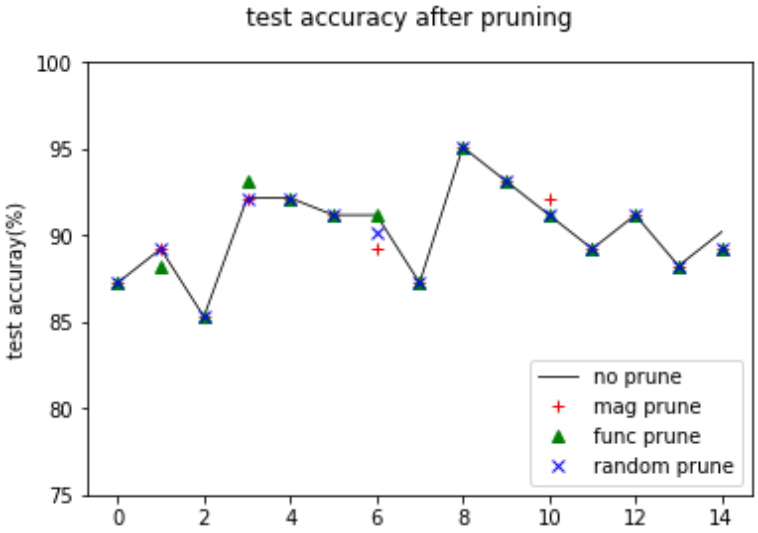

Fig. 4. Testing accuracy v.s. different pruning baseline

.

Finally, in order to determine the model's performance with respect to number of features, different amount of randomly chosen features are pruned, from 85% to 99% of the total number. The average testing accuracy for 15 models against pruning ratio is shown in fig 5.

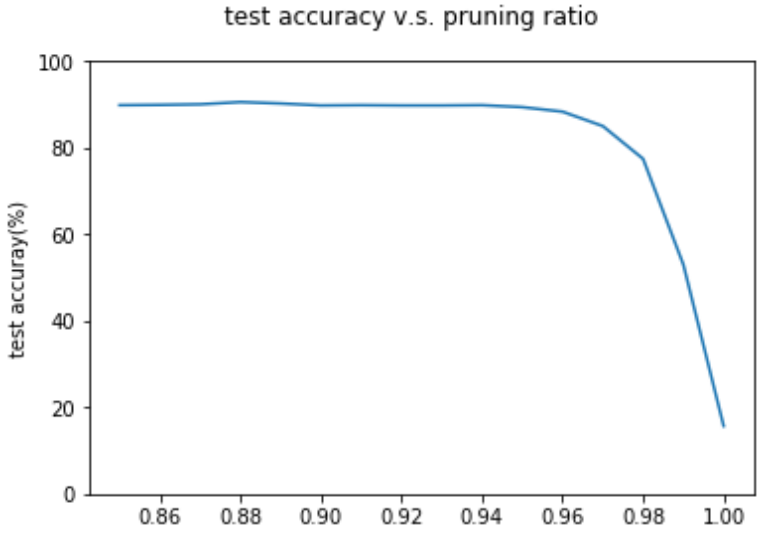

Fig. 5. Testing accuracy v.s. pruning ratio

.

Based on this result, testing accuracy drops greatly when more than 96% features are pruned. Hence, the fully connected classifier can work effectively with only 25 (5%) features.

4 Conclusion and Future Work

In this paper, a facial expression analysis model is trained based on pretrained ResNet18 model. Static facial expression image dataset (SFEW) is used to fine tune the model and test its accuracy. The model can achieve around 90% testing accuracy after training.

Based on this model, magnitude and functional feature analysis are applied to features extracted by ResNet. The result suggests no significant feature can be found. The model is pruned based on magnitude and functional analysis result and compared with random pruning and original testing accuracy to further validate the conclusion. At last, different number of features are pruned to estimate the model's performance with respect to number of features. It indicates that the model can work effectively with only 25 features.

In the future, more experiments can be practised on this model. Firstly, principal component analysis (PCA) can be used to explore the feature's distribution. Secondly, parameters of ealier layers could be pruned to make the network sparser and faster. Besides, grayscale SFEW images can be used in ResNet model to see if the performance can be better by eliminating the impact of colors.

References

1. Dhall, A., Goecke, R., Lucey, S., Gedeon, T.: Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). pp. 2106–2112 (Nov 2011). https://doi.org/10.1109/ICCVW.2011.6130508
2. Gedeon, T.D.: Indicators of hidden neuron functionality: the weight matrix versus neuron behaviour. In: Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems. pp. 26–29. IEEE (1995)
3. Gedeon, T.D.: Data mining of inputs: analysing magnitude and functional measures. International Journal of Neural Systems **8**(02), 209–218 (1997)
4. Gedeon, T., Harris, D.: Network reduction techniques. In: Proceedings International Conference on Neural Networks Methodologies and Applications. vol. 1, pp. 119–126 (1991)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

6. Kotsia, I., Pitas, I.: Facial expression recognition in image sequences using geometric deformation features and support vector machines. IEEE Transactions on Image Processing **16**(1), 172–187 (Jan 2007). https://doi.org/10.1109/TIP.2006.884954
7. Li, S., Deng, W.: Deep facial expression recognition: A survey. IEEE Transactions on Affective Computing (2020)
8. Shan, C., Gong, S., Mcowan, P.: Facial expression recognition based on local binary patterns: A comprehensive study. Image and Vision Computing **27**, 803–816 (05 2009). https://doi.org/10.1016/j.imavis.2008.08.005
9. Tian, Y.L., Kanade, T., Cohn, J.F.: Facial expression analysis. In: Handbook of face recognition, pp. 247–275. Springer (2005)
10. timesler: facenet-pytorch (Apr 2020), https://github.com/timesler/facenet-pytorch
11. Wong, P.M., Gedeon, T.D., Taggart, I.J.: An improved technique in porosity prediction: a neural network approach. IEEE Transactions on Geoscience and Remote Sensing **33**(4), 971–980 (1995)