Comparison of CasCor and VGG on Facial Expression Classification

Chunhao Li

Research School of Computer Science,
Australian National University
u6527752@anu.edu.au

Abstract. The Static Facial Expressions in the Wild (SFEW) is a collection of facial emotions extracted from movies, which approximates real-world conditions and has excellent value in research. We experiment with CasCade-Correlation architecture(CasCor) preprocessed by LPQ and PHOG descriptors.[1], and VGG-16 on the original images. The result is compared with a non-linear SVM in the SPI baseline, [1], and it shows that VGG-16 performs significantly better than CasCor and SVM, while CasCor does not give remarkable performance. Further experiments show that VGG can achieve a 71.11% on test set and have huge potential on facial expression classification, while CasCor also has advantages on its architecture.

1 Introduction

Facial expressions are generated by the muscle change of individuals[1] and have a broad application space in Human-computer interaction (HCI), psychology, and the communication industry. Within the past decade, researchers have spent significant effort analyzing facial expressions because of the limitations of datasets. [4] The individual differences among subjects are one of the problems. Eye colors, textures, hairstyles, and many other features are used to distinguish individuals. To address the varieties of individual features, the dataset is supposed to contain a large sample of faces of varying backgrounds. [4]. Classification of emotions is another difficulty. The determination of the number of emotions and the reliability of labeling data is hard to justify. Most facial expression data are obtained in a lab-controlled environment, and subjects are asked to perform certain emotions. Although this approach can solve the uncertainty in justification, it is of limited use in research.

The major drawback of lab-controlled data sets is because the emotions in the real world are more complex. Head orientations, the background of the images, the degree of expressions, and many other aspects vary in the actual scene. Therefore, the SFEW is of great value in research since the images are based on movie frames that have high similarity to real-world scenarios. It considers the uncertainty like head movements, varied illumination, age, gender, and occlusion. [1] Regarding the classification of emotions, six emotions, enjoyment or happiness, sadness, fear, anger, disgust, and surprise, are widely accepted as the categories. [5] Combined with the neutral class, the seven categories are regarded as the labels for the SFEW.

1.1 Dataset and problem

The SFEW dataset contains 675 images, and each category has 100 images except that **Disgust** label has only 75 samples. We have another dataset which is the SFEW processed by the texture-based descriptors, LPQ and PHOG [1]. Texture features of each image are extracted by the descriptors and encoded as principal components, and the first five are selected for each descriptor. Local Phase Quantization (LPQ) is based on the discrete Fourier transform (DFT), which is highly insensitive to blur and illumination [2], while the Pyramid of Histogram of Oriented Gradients (PHOG) is a descriptor based on the HOG and has shown good performance in image classification. [3] Descriptors processed data is used for the CasCade-Correlation architecture (CasCor), while the original SFEW is directed used in VGG-16.

The goal of the task is to take in the inputs, either images or principal components described above, and classify each sample into 7 categories: *angry, disgust, fear, happy, sad, surprise* and the *neutral* class. The facial expression classification is worth solving because this technique is required in human-machine interfaces such as user authentication, video surveillance, caption generation. [6]

1.2 Outline of Investigation

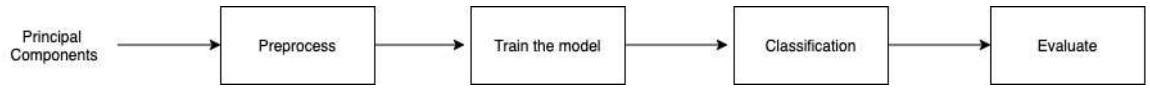

Fig. 1. Outline of Steps

There are four steps in the facial emotions classification task. (see Fig. 1)

The first step is preprocessing the input. This includes extracting features with descriptors (LPQ and PHOG) followed by normalization and transforming the original images to an appropriate size for VGG-16.

The second step is to build the model, a CasCor architecture, and a VGG-16 model. The initial parameters of CasCor are random initialization while we use the pre-trained VGG-16 model[10, 11]. After building the model, the parameters are tuned during the training.

The third step is to label the input sample, in which a winner-takes-all method is used to classify the models' output. Winner-takes-all is to use the largest prediction value of the label as the prediction result.

The final step is an evaluation which includes evaluating the model and comparing the result of different models. We use K-fold cross-validation to validate the VGG model.

The detail of methods applied in each step is described in the next section.

2 Method

2.1 Preprocess

For the descriptors processed dataset, we get a table of size 675×12 in which there are 5 columns for the principal components of LPQ, and 5 columns for the principal components of PHOG. The other two columns are for the label (integer between $1 - 7$) and the source of the original movie frame. The input features to the model is either the LPQ or PHOG principal components separately or combining them together. To investigate the performance of different input, we split the data into three sets in which only the related features and labels are selected. (675×6 for LPQ dataset, 675×6 for PHOG dataset and 675×10 for combined dataset). After inspecting the three sets, one

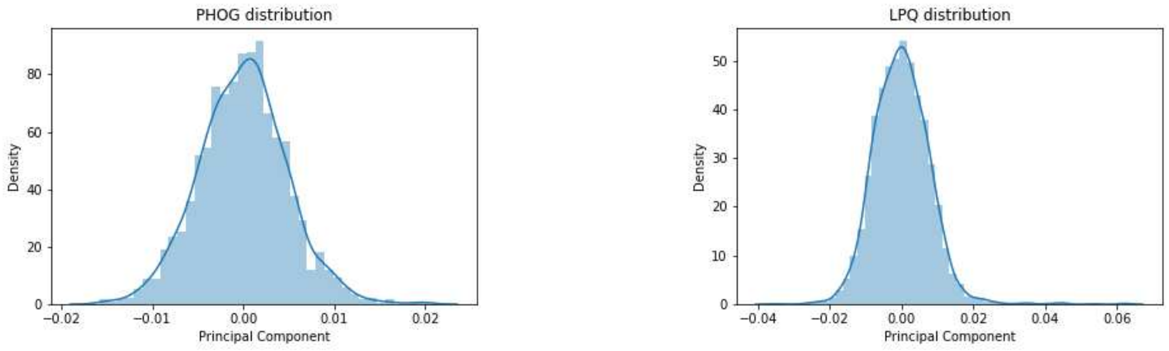

Fig. 2. Original distribution of LPQ and PHOG separately

of the sample of PHOG is missing, and dropping it leads to 674 samples for PHOG and combined dataset respectively.

The distribution of the LPQ and PHOG data sets are plotted, (see Fig. 2) and the Kernel density estimation (KDE) shows the features are well distributed around 0. However, the hidden problem is the values are floating point numbers and are too small which might cause problem in the training due to the precision. We use z-score to normalize the data which converts all values to a large scale with an average of zero and standard deviation of one.

For the original images input to the VGG-16, we resize the shorter edge of each image to 256 while preserving the ratio of width and height. After that, we random crop a 224×224 image from the resized sample and normalize with mean $0.485, 0.456, 0.406$ and standard deviation $0.229, 0.224, 0.225$, which are obtained from the ImageNet [10]. The final input to the VGG-16 is 675 tensors of size $3 \times 224 \times 224$ along with the label (from 0 to 6).

2.2 Model and evaluation

The Cascade-Correlation (CasCor) architecture is a supervised learning algorithm for artificial neural networks without determining its size and topology. It can build deep nets without the dramatic slow down in back-propagation neural networks with more than one or two hidden layers. [8] Many variations of CasCor, such as LoCC, which builds a priori knowledge of the task into the network and shows good performance on face recognition tasks. [9]

The CasCor architecture is a learning algorithm that starts with input and output units and adds a hidden unit in each iteration. The input features are augmented with a bias, which is all '1s', and then added to the model. The initial state of the model contains only input and output units which is the same as a neural network with no hidden neurons. In the first iteration, the live weights shown in the diagram are initialized randomly and then updated with the back-propagation technique. The hyper-parameters and methods used are cross-entropy loss, adam optimizer with learning rate 0.001, epochs of 500, and the batch size of 0.001. To overcome the overfitting problem in training, I reserve 5% of the data for final testing, and 90% of the rest data for training, and 10% for validation. After the weights associated with output units (live weights) have been tuned, the validation set is used to evaluate the model. If the validation loss is acceptable (< 0.1), we accept the model. Otherwise, a new hidden unit will be added to the network.

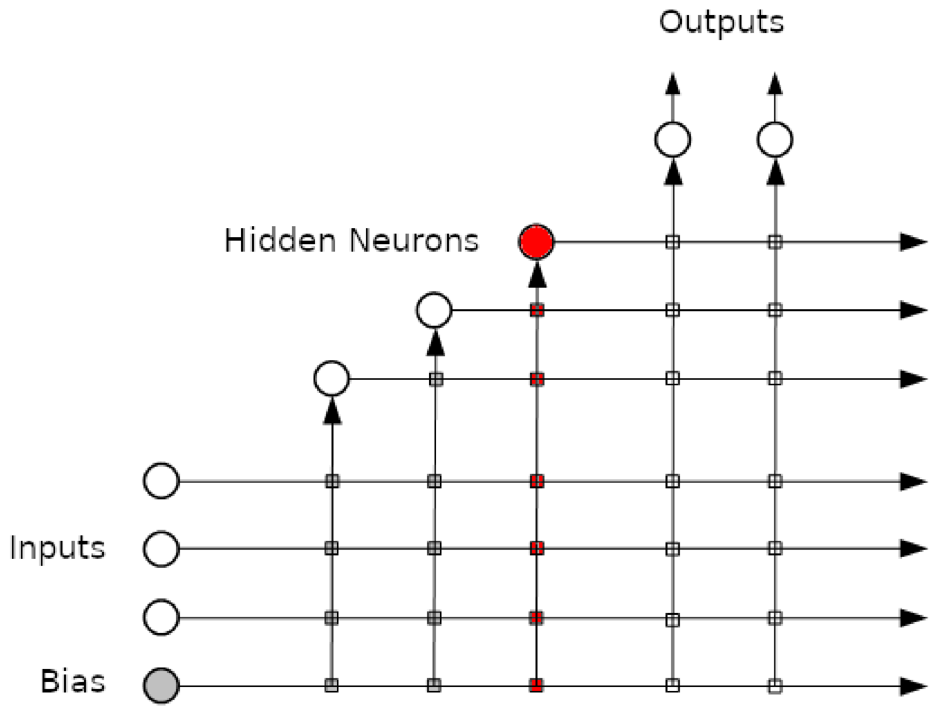

Fig. 3. Adding the hidden neuron. Red boxes represent live weights. Others are frozen weights [12]

The hidden unit is selected from a candidate pool such that the candidate has the maximum S, the sum over all output units of the magnitude of the correlation between the candidate unit's value V and E_{err}, the residual output error. [8] For this multi-classification task, the residual output error E_{err} is defined as the difference of the predicted probability of the **correct** class and 1. The predicted probability for each class is from applying softmax to the output units trained previously. As shown in Fig. 3, when adding a candidate unit, the input connections are from all of the network's external inputs and all pre-existing hidden units which need to be trained iteratively. [8]. The candidate unit V is thus a linear multiplication of inputs and existing hidden units' values. Therefore, S is defined as

$$S = \sum_{o} \left| \sum_{p} \left(V_p - \bar{V}\right) \left(E_{p,err} - \overline{E_{err}}\right) \right|$$

where o is the network output and p is the training pattern. $\bar{V}$ and $\overline{E_{err}}$ are V and E_{err} averaged over all patterns. [8] The input weights of the candidate unit are trained with the back-propagation same as the previous one with E_{err} as the true label and $1/S$ as the loss function. Therefore, the input weights can be adjusted to maximize S. The candidate pool is a set of candidate units with different sets of random initial weights. We use 4 for the candidate pool size and randomly initialize the incoming weights to the candidate unit. The candidate unit with the maximum S is selected and is added to the neural network. In implementation, the input weights to the new hidden unit are frozen, and the weights connecting to the output units are live again. The incoming weights to the new hidden unit will not be saved, but the new hidden unit's value will be calculated and augmented to the training, validation, and test inputs respectively, which acts as saving the model. This is the end of the first iteration. The next iteration starts from training weights of output units described above. In total, there are 32 iterations and 32 hidden units will added to the CasCor architecture in the end. The final model is evaluated on the reserved test set.

VGG is a deep convolutional network with small convolution filters (3×3) and performs well on large-scale image recognition tasks. [11] The architecture is shown in Table 1. Each convolutional layer uses a 3×3 filter with padding 1, and Max-pooling is performed over a 2×2 window with stride 2. The last three layers are the fully connected (FC) layers with output channels $4096, 4096, 7$ respectively, and the first two FC layers are followed by a **Dropout** with probability 0.5. Except for the last layer, all layers are using the ReLU activation function. During the training, we use the cross-entropy loss, stochastic gradient descent (SGD) with momentum 0.9, and the starting learning rate 0.01. The learning rate decreases by a factor 0.1 after each epoch. To evaluate the model, 10% of the images are reserved for final testing, and the rest are for training combined with 5-fold cross-validation. In each train-validation splitting, a batch size of 64 and max-epoch 10 are used. The initialization of parameters is from a pre-trained VGG-16 model, which has been trained on a large number of images. [10]

3 Experiments and Results

The baseline classification with the non-linear SVM calculated by averaging the accuracy for the LPQ and PHOG sets is 19.0%. [1] Using the parameters described above, the CasCor architecture gives an accuracy of 16.9% and loss of 2.0 on the reserved test data set. If the two sets are combined as the input to the model, CasCor gives the accuracy of 16.1%. CasCor does not show better performance compared with the benchmark result.

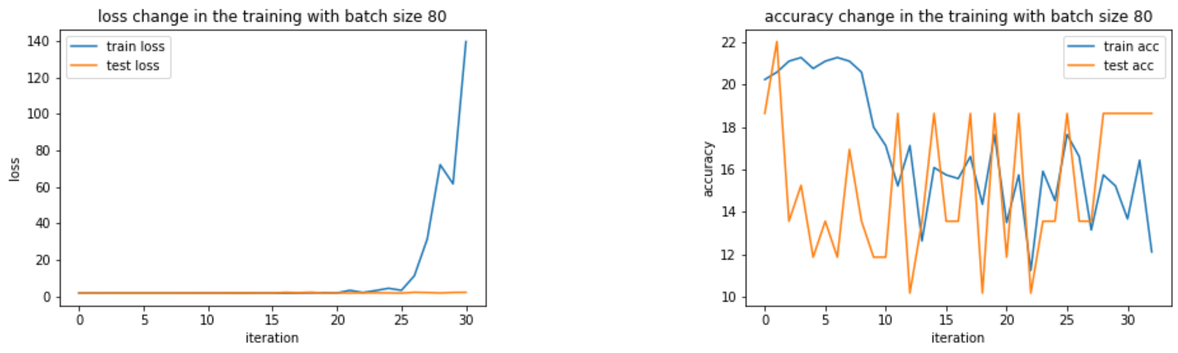

Fig. 4. Loss and accuracy change during the training for LPQ dataset - CasCor

Table 1. Architecture of VGG-16: convolutional layer parameters are conv-(receptive field size)-(output channel)

Layer number	ConvNet Configuration	Activation
1	conv-3-64	ReLU
2	conv-3-64	ReLU
	MaxPool	
3	conv-3-128	ReLU
4	conv-3-128	ReLU
	MaxPool	
5	conv-3-256	ReLU
6	conv-3-256	ReLU
7	conv-3-256	ReLU
	MaxPool	
8	conv-3-512	ReLU
9	conv-3-512	ReLU
10	conv-3-512	ReLU
	MaxPool	
11	conv-3-512	ReLU
12	conv-3-512	ReLU
13	conv-3-512	ReLU
	MaxPool	
14	FC-4096 (Dropout prob-0.5)	RELU
15	FC-4096 (Dropout prob-0.5)	RELU
16	FC-7	None

With the settings described in the last section, the loss and accuracy change during the training for the LPQ data set are plotted. (see Fig. 4) The train and test accuracies are disturbing a lot in training, while the loss gives a much smoother curve. Training loss starts to increase after the $25th$ iteration, while the test loss remains in a small range. It means the model cannot learn more features from the dataset. One disadvantage of batch training in CasCor is that the training speed slows down a lot due to many back-propagations (4 candidate units' weights and output units' weights in one iteration).

The evaluation of the model relies on the reserved dataset, and the traditional K-fold cross-validation is not suitable for CasCor, a kind of dynamic model. Once a candidate unit is selected from the candidate pool, the parameters of connections into that unit are frozen. If we need to use a new split of train and validation sets, we have to extend the network, resulting in an extremely deep network. From Fig. 4, a network with 25 hidden units (depth is 25) started performing worse.

The result of VGG-16 on the reserved test data is 35.29%. The comparison of three models are displayed in Table 2. The accuracy increase by about 80% compared with the baseline model.

Many experiments have proved that CNN-based models can perform well on image recognition, especially with large training samples. For example, 75.2% test accuracy on FER2013 data set [14] and 48% on Kaggle Facial Expression Recognition Challenge data [13]. Since the size of the SFEW data set is small, we perform data augmentation to generate more images, where we add three other images for each image: flip, rotation with 90 degrees and rotation with 180 degrees. Hence, the new data set contains 2700 images. With the same setting as before, we train and evaluate VGG-16 on this dataset, and we get 62.96% test accuracy in the end.

Table 2. The comparison of accuracy on different models

SVM(baseline)	CasCaor	VGG-16
19.0%	16.9%	35.29%

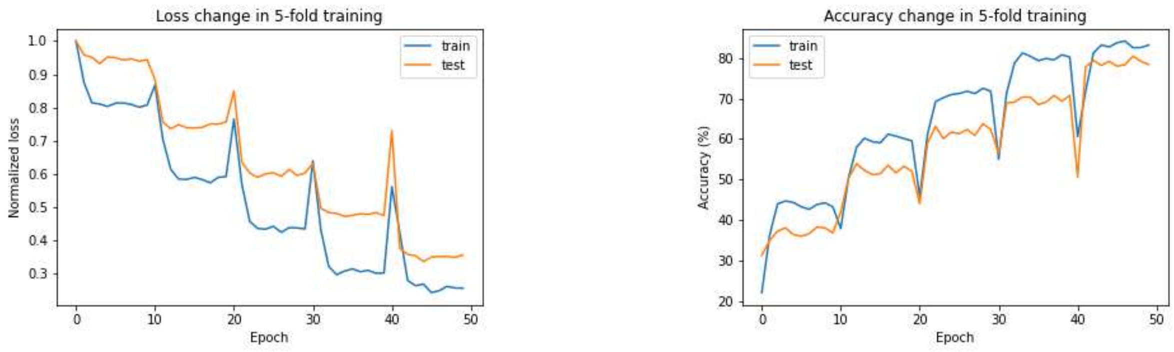

Fig. 5. Loss and accuracy change during the training - VGG

The loss and accuracy change during training are plotted. (see Fig. 5) It is a 5-fold training, and each has 10 epochs. The little drop in accuracy figure and the little increase in the loss figure correspond to the first epoch in each splitting. The model needs to learn from the new train and validation data sets for several epochs. The figure does not imply overfitting or convergence, which indicates the VGG-16 still has possibilities to learn from image features. A further experiment is to use 10-fold training, which involves more training and reuses the data more times. The accuracy on the reserved test set is 71.11%. The results of VGG-16 are shown in Table 3, which indicates the data augmentation is beneficial for VGG model training.

Table 3. The comparison of test accuracy with VGG-16 on different datasets and validation techniques. SFEW_aug is the augmented data set

SFEW 5-fold	SFEW_aug 5-fold	SFEW_aug 10-fold
35.29%	62.96%	71.11%

One disadvantage of VGG-16 is the training time, which is significantly longer than CasCor. The approximate training time is shown in Table 4. It is measured on the GPU of **NVIDIA GeForce RTX 3090** with the SFEW data set. The training time on augmented data set is $2-3$ times longer than on the original data set.

CasCor can achieve quicker training with the quick prop algorithm which is calculus efficient based on second-order differentiation, and out-performs other backprop-like algorithms [8]. However, we do not use the activation function in this task, so it is not suitable here, although it is widely used in other situations.

Table 4. The approximate training time of different models on SFEW

CasCor	VGG-16
12 minutes	60 minutes

4 Conclusion and Future Work

Based on the experiment results, the CasCor architecture performs worse than the non-linear SVM shown in the benchmark, while VGG-16 can achieve much better accuracy on the SFEW dataset.

Cascade-correlation architecture can be regarded as a special kind of neural network whose hidden layers contain only one hidden unit, and the unit is connected to both the input and output units. A possible variation is adding cascade chunks with a fixed size set prior to training instead of adding one hidden unit each time. [9]

Another modification is regarding the training speed of CasCor, and we can train candidate units in the candidate pool in parallel because they do not interact with one another or affect the active network during training. [8] With parallel computation, we can increase the pool size to ensure the performance of the model.

We can also use different descriptors to extract more accurate features from the original images, which then act as the input to the CasCor.

Regarding the VGG-16 model, data augmentation helps increase the facial expression classification significantly and gives a 71.11 test accuracy, which implies the ability to perform classification in a real-world environment. Further improvement can focus on increasing data size with different augmentation techniques. Related research has shown that GAN can generate different styles from a source image and help improve the image classification accuracy. [15]

Also, we can modify the VGG models with different techniques such as regularization, batch normalization, or combined them with different methods to improve the model. Batch normalization is an effective method that can accelerate training and saves much time [16].

References

1. Dhall A, Goecke R, Lucey S, et al. Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark[C]//2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). IEEE, 2011: 2106-2112.
2. Ojansivu V, Heikkilä J. Blur insensitive texture classification using local phase quantization[C]//International conference on image and signal processing. Springer, Berlin, Heidelberg, 2008: 236-243.
3. Bosch A, Zisserman A, Munoz X. Representing shape with a spatial pyramid kernel[C]//Proceedings of the 6th ACM international conference on Image and video retrieval. 2007: 401-408.
4. Kanade T, Cohn J F, Tian Y. Comprehensive database for facial expression analysis[C]//Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580). IEEE, 2000: 46-53.
5. J. D. Velazquez. (1997). Modeling Emotions and Other Motivations in Synthetic Agents. In Proceedings of the Fourteenth National Conference on Artificial Intelligence, 10-16. Menlo Park, Calif.: American Association for Artificial Intelligence, 1997.
6. Hemalatha G, Sumathi C P. A study of techniques for facial detection and expression classification[J]. International Journal of Computer Science and Engineering Survey, 2014, 5(2): 27.
7. Abiodun O I, Jantan A, Omolara A E, et al. State-of-the-art in artificial neural network applications: A survey[J]. Heliyon, 2018, 4(11): e00938.
8. Fahlman S E, Lebiere C. The cascade-correlation learning architecture[R]. CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 1990.
9. Khoo S, Gedeon T. Generalisation Performance vs. Architecture Variations in Constructive Cascade Networks[C]//International Conference on Neural Information Processing. Springer, Berlin, Heidelberg, 2008: 236-243.
10. Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database[C]//2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009: 248-255.
11. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
12. Wikipedia: The Free Encyclopedia. Wikimedia Foundation Inc. Updated 22 July 2004, 10:55 UTC. Encyclopedia on-line. Available from https://en.wikipedia.org/wiki/Endangered Species. Internet. Retrieved 10 August 2004.
13. Raghuvanshi A, Choksi V. Facial expression recognition with convolutional neural networks[J]. CS231n Course Projects, 2016, 362.
14. Pramerdorfer C, Kampel M. Facial expression recognition using convolutional neural networks: state of the art[J]. arXiv preprint arXiv:1612.02903, 2016.
15. Perez L, Wang J. The effectiveness of data augmentation in image classification using deep learning[J]. arXiv preprint arXiv:1712.04621, 2017.
16. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]//International conference on machine learning. PMLR, 2015: 448-456.