Emotion Classification and Input Encoding Technique Analysis using SFEW Dataset

Chunze Fu[1,1]

[1] Research School of Computer Science, Australian National University,
108 North Rd, Acton ACT 2601, Australia

Abstract. Facial expression analysis is a critical area of machine vision research. Facial expression dataset such as SFEW database is worth investigating as their data are extracted from and can represent close to the real-world environment. In this paper, we proposed a 3 layer fully connected neural network for emotion classification task using SFEW dataset. As part of the analysis, various input encoding techniques were tested and compared for their effects on the network's performance. Furthermore, we proposed a deep learning solution by building a classification neural network based on the classic VGG16 network. The results shown that our deep learning network out-performs the simple fully connected neural net by 4.5% (38% to 33.5%). It also achieved higher averaged accuracy and SPI score under SPI evaluation baseline compared to SVM method used in the original SFEW paper. As an extension, various possible improvement methods were discussed based on the state-of-the-art solution to the problem.

Keywords: Emotion prediction, SFEW database, neural network, input encoding techniques, deep learning

1 Introduction

In recent years, automatic facial expression analysis has been at the center of computer vision and machine learning research. As an essential condition, effective facial expression analysis relies on quality facial data recorded in varied realistic environment [1]. As a result, researchers have put efforts in constructing various static facial expression databases. Early databases have been developed in highly controlled lab environments (such as the JAFFE Database [2]). However, as the task difficulties of machine vision problems escalated, more recent datasets were developed in realistic, unconstrained conditions including varied head gestures, facial expressions, illuminations and so on.

Among these datasets, Static Facial Expressions in the Wild (SFEW) [3] has been one of the earliest to address realistic, unconstrained facial expression data. It became a popular database that has been extensively used in the field. In this paper, we firstly developed a simple 3-layer fully connected neural network for emotion prediction task using the SFEW database. After comparing the classification results with the baseline results in the original SFEW database paper, four different input pre-processing techniques were investigated to improve the network's classification performance. Furthermore, a deep learning approach (Convolutional Neural Net modified from VGG16 [7]) was developed and tested as an extension to include more modern neural network techniques. Recommendations were made on how to further improve the our network based on the state-of-the-art solutions to this problem.

2 Method

2.1 Implementation Details

Input data We use the local phase quantization (LPQ) and the pyramid of histogram of oriented gradients (PHOG) data extracted from SFEW database image as our input data to the fully connected neural net. Local phase quantization is based on computing the short-term Fourier Transform on a local image window [4]. PHOG is an extension of histogram of gradient(HOG), which computes the frequency of occurrences of gradient orientations in local image portions [5]. Similar to the SFEW paper [3], we applied a principle component analysis to these data to reduce experiment complexity. Finally. the top five principle components of each type were used as our input data.

Instead of extracted PCA components, our deep learning approach directly uses the SFEW images as inputs. In particular, there are in total 675 images from seven different emotion categories. All images are in the format of 720-by-576 pixels RGB.

Class Label Both input data types consist of seven different labeled emotion classes, namely angry, disgust, fear, happy, neutral, sad and surprise, each labeled as an integer from 1 to 7. As the nature of the problem is discrete, we then transformed the labels as one-hot encoded vectors for computing the loss function.

Network Structure - Fully Connected Neural Net As shown in Fig. 1, we constructed a three-layer neural network for the classification task. Depending on the input data, input size of the network can be either 5 or 10 (For one of LPQ or PHOG data, or both combined). Number of hidden neurons were set at 30 by running a set of experiments for different hidden unit sizes and picking the highest classification accuracy. Details of the experiments can be found in the result and discussion section. A Logistic Sigmoid activation is applied to hidden neuron. The output has seven classes which corresponds to probability of the seven emotion categories. A softmax activation was then added to normalized the probability values.

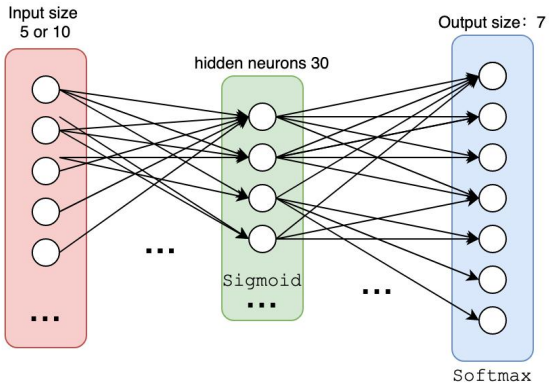


Fig. 1. Network Structure

Training Process - Fully Connected Neural Net In order to reduce overfitting and compare our results with the original paper, we trained the network using a five-fold cross validation method. First of all, 10% of the total data (68 out of 675 items) were set apart as validation set. Next, we train the network by applying five-fold cross validation on the rest, each time taking 1 out of 5 chunks of data as test set. We then compute the average test loss and back propagated through the network to update the parameters. Accuracy on the validation set was also computed in each epoch as the real indicator for the network performance.

Network Structure - Deep Neural Net As there are numerous hyper-parameters and network configurations to choose from for the deep neural net, our inspiration is to construct the network based on a widely-used classification CNN with proved high performance. We chose to modify the 16 layer VGG model [7] which achieved a 92.7% classification rate on the ImageNet dataset [9]. A graph of the VGG16 model configuration can be found in Appendix 2.

Table 1 shows a comparison between the standard VGG16 and our modified VGG model. Since our input image is much bigger than what's used in the VGG paper, we followed the general structure of VGG and added four extra convolutional layers separated by maxpooling in order to compensate the difference in input dimensions. Since we want to classify from seven emotion labels, the last fully connected was changed as a mapping from 4096 to 7 neurons before fed into the softmax layer.

Table 1. Comparison of standard VGG16 [7] and our modified VGG16 network configurations. The convolution layers parameters are denoted as conv(reception field size)-(number of channels). The ReLU activation function is not shown for brevity. Difference between our model and standard VGG16 is marked as bold.

Standard VGG16	Modified VGG16
Input (224 × 224 RGB image)	Input (720 × 576 RGB image)
	conv3-16 conv3-16
	maxpool
	conv3-32 conv3-32
	maxpool
conv3-64 conv3-64	conv3-64 conv3-64
maxpool	maxpool
conv3-128 conv3-128	conv3-128 conv3-128
maxpool	maxpool
conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256
maxpool	maxpool
conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512
maxpool	maxpool
conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512
maxpool	maxpool
FC-4096	FC-4096
FC-4096	FC-4096
FC-1000	FC-7
Soft-max	**Soft-max**

Training Process - Deep Neural Net We trained our deep neural net using the normal train-test split method. First of all, the original dataset was randomly split into training set and test set at a ration of 80% to 20%. We performed 100 epochs of training on the training set, and computed test accuracy using the test set for objective network performance. Hyper-parameters such as learning rate (0.01), batch size(32), number of epochs (100) were decided by following the suggestion from the original VGG16 paper [7]. As an extension, hyper parameters such as batch size were investigated for the optimal parameter choices. Finally, similar to the fully connected neural net, the SPI evaluation was also completed for comparing results with the SFEW paper.

2.2 SPI Protocol and Performance Evaluation

Apart from evaluation based on validation set accuracy, we also adopted the important testing scheme from SPEW paper by including the Strict Person Independent(SPI) Protocol. In SPI Protocol, the training data is split evenly per label into two sets in a person independent manner. The training algorithm is thus changed to be two-fold using data from set 1 and set 2. The performance evaluation is then presented by computing the average classification accuracy (baseline accuracy) over the two sets, as well as averaged factors including accuracy, precision, recall and specificity [3].

2.3 Input Encoding Techniques - for fully-connected neural net

Various input encoding techniques were tested to improve network performance. Most of the techniques are adopted from [6].

Normalization From the input data distribution (Fig. 2), we noticed the magnitudes of both LPQ and PHOG mostly lie between -0.01 to 0.01, which could impede the network performance. Apart from one input feature from LPQ input, the rest generally form a normal distribution with mean value around 0. Therefore, we used a simple linear squashing function to normalize the input to range 0 to 1.

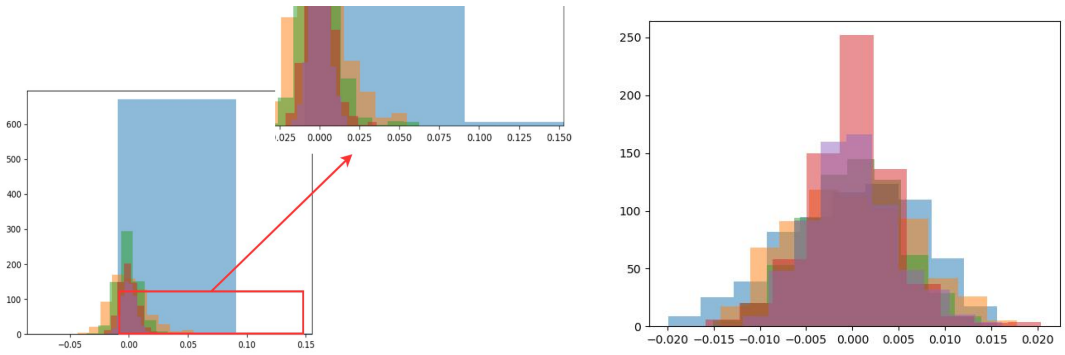


Fig. 2 Histogram of Input data. Left is LPQ data, right is PHOG data.

Nominal Data Transform From fig. 2, the outlier input feature (colored blue) in LPQ data appears to be a nominal value. There is only a step distribution which separates the data into less than or larger than 0.09. Therefore, instead of using a single continuous value as input, the input was transformed to a single class indicator. To comply with the normalization range, we represented data less than or greater than 0.09 as 0.1 and 0.9 respectively.

Adding Random Noise In order to reduce overfitting and improve network performance, we also tested training the neural net with normal and randomly distorted noisy data in 50 epoch alternations. Effects of different level of added randomness were also investigated.

Magnifying Input Data In order to change input data magnitude, instead of normalization, we also tried directly multiplying a magnifying parameter to all input data (e.g., 1000 times original inputs). This method does not change the input distribution, but successfully accelerated the learning speed and improved overall validation accuracy.

3 Results and Discussion

3.1 Hidden Neurons for fully-connected neural net

A series of experiments were run by combining the original LPQ and PHOG data. Table 2 below demonstrates the maximum experiment results. It shows that with hidden neurons less than input size, the network seems to perform worse due to its lack of complexity. However, when there are enough hidden neurons (more than 30), there is not a clear correlation between hidden neurons and prediction accuracy. Therefore, in order to balance network performance with computational complexity, 30 was chosen as hidden layer size.

Table 2. Prediction accuracy given different number of hidden neurons for the fully-connected neural net, run for 5000 epochs, learning rate 1

Number of hidden neurons	5	10	30	50	70
Prediction accuracy (%)	11.18	14.02	15.62	15.77	15.69

3.2 Accuracy from Original Input - fully-connected neural net

For the SPEW LPQ and PHOG principle components input data, experiments were run by using only LPQ or PHOG, as well as combining the two types of data. The classification accuracy 18.32% for LPQ and 15.92% for PHOG, which is significantly lower than that in the SFEW paper [3] (43.71% and 46.28%). This is partly because of the complex and close to real world conditions of the SFEW dataset. What's more, the principle component analysis results in loss of information. The top five principle components cannot very well capture all important patterns of the data. Thus, the classification accuracy compared to using full LPQ and PHOG data is much lower.

We have also combined LPQ and PHOG data to feed to the network. The highest prediction accuracy was 20.01%. It proved that with the increase of information, the network is able to classify inputs more accurately. But the improvement so far is very limited.

Regarding the SPI baseline, the average baseline accuracy is 21.78%, slightly higher than what's proposed in SFEW paper. The SPI baseline score was computed to be 0.32 by averaging score items accuracy, precision, recall and specificity over the two training folds. The detailed score items for these score items were also recorded and analyzed in the following sections. (see appendix 1)

3.3 Normalization and Nominal Data Replacement

We combined the linear squashing normalization and nominal data replacement as they both bring the input data to range 0 to 1. Again, the classification after input pre-processing improved to 26.23% for LPQ and 23.36% for PHOG. Moreover, by using both LPQ and PHOG data combined, we were able to improve the classification accuracy to 30.71%. All above results are significantly better than those using original input data. This is mostly due to that

normalization essentially magnifies the input from around 1e-3 to the range from 0 to 1. Therefore, the network will learn much faster with much greater gradients computed at each iteration. In other words, the underlying patterns from the input data was magnified from normalization. Moreover, values between 0 to 1 are ideal for logistic sigmoid activation function to compute the larger gradient values, therefore more effectively back propagate through the network.

Regarding the SPI baseline, the average baseline accuracy and SPI baseline score were 26.38% and 0.32 respectively. As expected, there is a significant improvement in the baseline accuracy.

3.4 Adding Random Noise

Adding noise generally helps to prevent overfitting and increase the network's ability to generalize. On top of normalization and nominal data replacement, we then trained the network alternatively on clean and noisy input data. Experimented results using different noise level are recorded in table 3 below.

As can be found, adding random noise to normalized input data does not have a clear influence on classification accuracy. It is probably due to the nature of input data being already random to some extent. Meanwhile, the use of cross validation and input normalization have already prevented the network from severe overfitting.

It is also worth noticing that adding random noise could result in increased oscillation of both test and validation losses such as in Fig. 3. This could be mitigated by reducing noise level as the training goes [6]. However, in our experiment, there is no clear need to add random noise to the input data as recommended practice.

Table 3. Prediction accuracy given different added random noise to normalized LPQ and PHOG inputs, trained for 8000 epochs, learning rate 1

Noise Variance	0	0.05	0.1	0.2	0.3
Prediction accuracy (%)	30.71	30.14	29.73	30.20	28.15

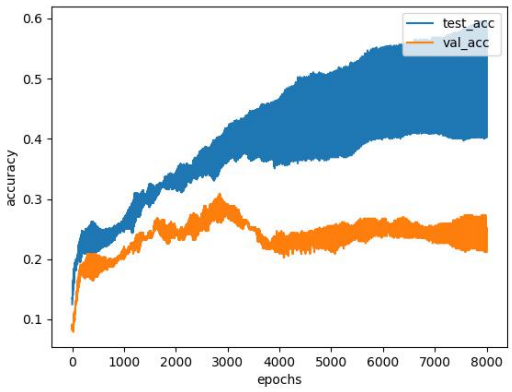


Fig. 3 Oscillation in test and validation losses with noise variance 0.2

3.5 Magnifying Input

Finally, different magnifying parameters were tested on raw LPQ and PHOG data. From table 4 it is clear that classification accuracy increases along with magnifying parameter, with the peak performance occurring at magnifying parameter equal 100. In comparison with experiment using normalized input, this method performs slightly better (33.57% compared to 30.71%) result. However, as magnifying parameter continues to increase, the accuracy does not keep improving.

Table 5 summarizes testing results based on SPI baseline. The optimal performance was reached when magnifying parameter is 1000. Again, this method performs slightly better than normalization (27.27% and 0.37 compared to 26.38 and 0.32).

In conclusion, it is the magnitude of input data that has the most significant influence on network performance and classification accuracy. From our experiment, the optimal magnitude parameter is 100 under cross validation, or 1000 under SPI baseline evaluation scheme.

Table 4. Classification accuracy after applying different magnifying parameters on raw LPQ and PHOG input data, trained for 5000 epochs, learning rate 1

Magnifying parameter	0	20	100	500	1000
LPQ (%)	18.32	21.53	30.02	26.03	25.60
PHOG (%)	15.92	19.26	29.86	24.15	24.87
LPG & PHOG (%)	20.01	28.54	33.57	29.67	28.05

Table 5. SPI evaluation given different magnifying parameters on raw LPQ and PHOG inputs combined, trained for 5000 epochs, learning rate 0.1

Magnifying parameter	0	20	100	500	1000	3000
SPI accuracy (%)	21.78	21.19	22.42	25.35	27.27	25.08
SPI baseline score	0.32	0.31	0.31	0.36	0.38	0.36

3.6 Modified VGG16

The experiment results using the initial network configuration achieves a 29% validation accuracy, which is not as good compared to our fully connected neural net (33.57% using LPG&PHOG, with magnifying parameters 100). Based on the instinct that a deep neural net by nature should not be less capable compared to a simple 3-layer fully connected neural net, we continued to experiment for optimal parameter choices.

First of all, batch size affects network performance by controlling the added randomness and converging speed. Thus smaller batch size often leads to increased general performance. We tested different batch size choices and their influence to the network performance. Table 6 below summarizes the results. It can be found that the prediction accuracy improves significantly as we used smaller batch size to train the network. The peak performance occurred at batch size 4, where the maximum prediction accuracy reached 38%. In comparison, our modified VGG16 was able to out-perform 3-layer fully connected neural net by roughly 4.5%.

Table 6. Batch size versus prediction accuracy on test set for modified VGG16 model. Trained for 100 epochs, learning rate 0.01.

Batch size	1	4	16	32
Max Prediction accuracy - test (%)	25	38	26	29

Due to the large scale of the network (in total 73.5 million parameters) and time consumption for every training attempt, it was not cost-efficient to conduct more experiment on other hyper-parameters given the hardware limitation. Therefore, with the remaining possibility of finding a better parameter choice, we decided to experiment in detail with the current optimal network configuration (batch size 4, learning rate 0.01).

Table 7 summarizes the training results in a range of 100 epochs. It can be found that the peak performance (38%) occurred at 70 epochs, which is still the highest prediction accuracy we obtained so far from the modified VGG16. For clarity, these results were further plotted into figure 4. The plot demonstrates a general pattern of neural network training curve, where the training accuracy keeps improving while the test accuracy firstly goes up and downwards later. Epoch 70 roughly marks the overfitting point, meaning the network start to overfit on the training set, thus losing its capability of generalization over unseen data (test set). Therefore, the optimal set of parameters should be obtained from the network after 70 epochs of training.

Table 8 summarizes the experiment results using SPI baseline. The optimal performance was reached again at 70 epochs. In comparison, the maximum prediction accuracy and SPI baseline scores are both significantly improved from results in section 3.5 using fully-connected neural net (34% vs. 27.27%, 0.466 vs. 0.38). Comparing to original SFEW paper [1], the averaged accuracy almost doubled (from 19% to 34%). Moreover, from what's show in in appendix 1, average class-wise precision, specificity and recall, as well as the final SPI score, were all improved significantly. The above results have proved that a more modern deep learning technique (modified VGG16) indeed out-perform both traditional machine learning classification technique such as Supported Vector Machine used in original SFEW paper, as well as simple 3-layer neural net which we have tested before.

Table 7. Training and testing accuracies versus number of epochs, trained on modified VGG16 network. Trained for 100 epochs with learning rate 0.01, batch size 4

Epochs run	10	20	40	60	70	80	100
Prediction accuracy - train (%)	13	20	48	81	93	97	99
Prediction accuracy - test (%)	12	13	25	31	38	32	31

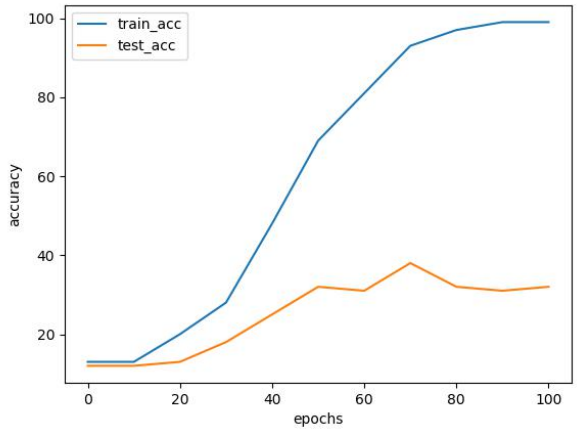


Fig. 4 Training and test accuracy plot from table 7 results

Table 8. SPI accuracy and baseline score at different number of epochs trained, trained on modified VGG16 network. Trained 100 epochs for both folds, with learning rate 0.01, batch size 4

Epochs run	10	20	40	60	70	80	100
SPI accuracy (%)	15	17.5	24	30	34	25.5	24
SPI baseline score	0.325	0.344	0.398	0.445	0.466	0.421	0.398

Even though our modified VGG16 did improve the classification accuracy, the result is still quite low compared to VGG network's past achievements on datasets such as ImageNet [9] (92.7%), and VOC2012 [10] (89.9%). This is mostly likely due to the complex nature of expression recognition dataset itself. Among the complexities resulting from unconstrained, real-world environment, occlusions and variant poses are two major problems which lead to significant changes of facial appearance, therefore lowering prediction accuracy [11]. It is obvious that traditional classification deep neural nets such as VGG were not designed specifically to tackle such problems. On top of this, the other possible reason is that a dataset consisting of 675 images can still be quite small and not generalized enough, especially in today's deep learning world. One could consider performing various data augmentation methods on the original dataset, in order for the network to learn from a richer input space.

State-of-the-art classification accuracy on SFEW datasets was achieved by Wang, et.al, [8], who reported a 56.4% prediction accuracy overall. They specifically targeted occlusion and post variations and designed a region attention network based on a number of cropped facial expression details as additional inputs. From their approach, we learned a few improvements that could be made to improved our method.

Cropping facial image The SFEW images are extracted from movies scenes, which means that apart from the actual face expressions, there can be much more irrelevant information included in an image, such as background scene and various objects. In our particular problem, it is not necessary to use the whole image as input since all extra those information may add unnecessary complexities and randomness to the neural net. Instead, we could use only the cropped faces/persons, and resize them as out input images.

Pretrain on facial expression dataset Because of the limited number of SFEW images, the neural net may not be able to learn very well for generalization. One way to solve this problem is by pretraining the network on existing facial expression dataset such as MS-Celeb-1M [12]. The idea is to let the network recognize faces and emotions to a certain degree, before fine tuning specifically for our SFEW dataset.

Newer version of SFEW dataset A point that can easily be ignored is that SFEW database has been updated annually with more images at the moment. Use newer version.

4 Conclusion and Future Work

In this paper, we have designed a classification network for emotion classification task using SFEW database. We compared the classification results with those in the original paper using both cross validation and SPI protocol. After testing the raw PCA inputs, we proposed four different input encoding techniques and analyzed their effects on the network performance. From the experiments, we concluded that the most significant factor affecting classification accuracy is the magnitude of input data, which can be solved by either applying a magnitude parameter to the input or standard normalization. Furthermore, we proposed a modified VGG16 network to include more modern deep learning solution. The results shown that our deep learning approach achieved the best outcome over all other methods. However,

after researching for state-of-the-art solution to the problem, we analyzed our current approach and proposed recommendations for further improving the network performance.

In regards our fully-connected neural net, as part of the extension, it is worthwhile to investigate the data used in SFEW paper [6] to find out the reason why our classification accuracies are generally lower than the original results. It is assumed that the principle component analysis does not keep all information from raw input, resulting in the loss of accuracy. However, it is worth researching to use original LPQ and PHOG data as inputs. There is also another possibility that the accuracy loss is due to the difference between simple 3-layer neuron net and support vector machine algorithm, which was used as classification algorithm in the original paper.

In regards the modified VGG16 network, firstly, more experiments need to be run in order to decide the optimal hyper-parameters such as learning rate, learning rate decay, number of epochs and so on. We missed these experiments mostly due to hardware limitations to such big neural net. Moreover, as discussed in section 3.6, there are many possible improvements to make including using cropped facial images, pretraining network on facial expression datasets, using attention-like network structures and performing data augmentation on the original dataset. With the problem's complexity in mind, it is suggested that these approached should be ranked with priorities and tested based on their importance in future experiments.

References

1. A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, 2011, pp. 2106-2112.
2. M. J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Cod- ing facial expressions with gabor wavelets. In Proceedings of the IEEE International Conference on Automatic Face Ges- ture Recognition and Workshops, FG'98, 1998.
3. A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, 2011, pp. 2106-2112.
4. V. Ojansivu and J. Heikkil. Blur Insensitive Texture Classi- fication Using Local Phase Quantization. In Proceedings of the 3rd International Conference on Image and Signal Pro- cessing, ICISP'08, pages 236–243, 2008.
5. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Interna- tional Conference on Computer Vision and Pattern Recogni- tion, CVPR'05, pages 886–893, 2005.
6. Bustos R.A., Gedeon T.D. (1995) Decrypting Neural Network Data: A Gis Case Study. In: Artificial Neural Nets and Genetic Algorithms. Springer, Vienna
7. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
8. K. Wang, X. Peng, J. Yang, D. Meng and Y. Qiao, (2019) Region Attention Networks for Pose and Occlusion Robust Facial Expression Recognition. IEEE Transactions on Image Processing, vol. 29, pp. 4057-4069, 2020, doi: 10.1109/TIP.2019.2956143.
9. Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.
10. Everingham, M., & Winn, J. (2011). The PASCAL visual object classes challenge 2012 (VOC2012) development kit. Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep.
11. Changxing Ding and Dacheng Tao. A comprehensive survey on pose- invariant face recognition. ACM Transactions on intelligent systems and technology (TIST), 7(3):37. 2016
12. Guo, Yandong, et al. "Ms-celeb-1m: A dataset and benchmark for large-scale face recognition." European conference on computer vision. Springer, Cham, 2016.

Appendix: Network output results

1. **SPI protocol results from SFEW paper [1], 3 layer fully-connected neural net, and modified VGG16 network – average expression class-wise precision, recall and specificity results on SFEW data base.**

Results from SPEW paper [3]

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.17	0.15	0.20	0.28	0.23	0.16	0.15
Recall	0.21	0.13	0.18	0.29	0.21	0.16	0.12
Specificity	0.48	0.66	0.64	0.51	0.61	0.60	0.66

Fully connected neural net, raw input LPQ and PHOG combined, 3000 epochs, 0.1 learning rate

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.08	0	0.04	0.54	0.32	0.12	0.14
Recall	0.08	0.1	0.06	0.52	0.32	0.11	0.14
Specificity	0.56	0.62	0.56	0.60	0.58	0.57	0.56

Fully connected neural net, normalized input with nominal data replacement, LPQ and PHOG, 3000 epochs, 0.1 learning rate

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.24	0	0.26	0.45	0.24	0.28	0.24
Recall	0.24	0	0.25	0.46	0.24	0.28	0.24
Specificity	0.56	0.66	0.70	0.66	0.66	0.67	0.67

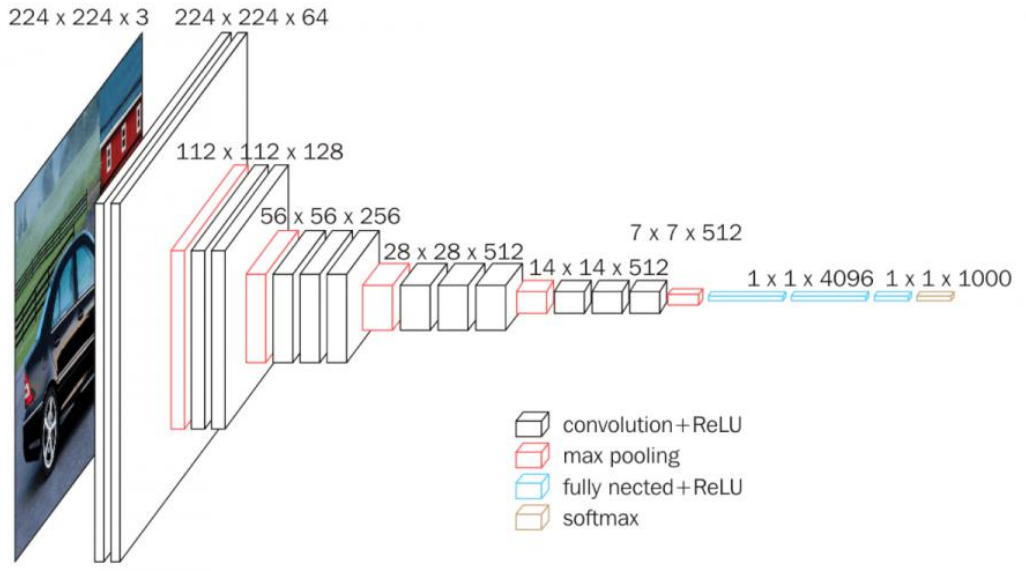
Fully connected neural net, magnified input with magnifying parameter as 1000, LPQ and PHOG, 3000 epochs, 0.1 learning rate

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.32	0.38	0.40	0.36	0.36	0.36	0.26
Recall	0.31	0.38	0.40	0.36	0.33	0.36	0.26
Specificity	0.75	0.82	0.76	0.76	0.76	0.76	0.73

Modified VGG16, 200 epochs, 0.01 learning rate, 4 batch size.

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.50	0.26	0.52	0.26	0.30	0.30	0.26
Recall	0.50	0.27	0.52	0.26	0.30	0.30	0.26
Specificity	0.91	0.90	0.92	0.87	0.88	0.88	0.87

2. Network configuration - VGG16 [7]



Retrieved from Neurohive website: https://neurohive.io/en/popular-networks/vgg16/