Faces Expression Detection with Residual Network and BiDirectional Neural Networks (BDNN) and Class Prototypes

Rizqi Ekoputris[1],

[1] Research School of Computer Science, Australian National University
u7033006@anu.edu.au

Abstract. Nowadays technology to detect facial expressions are widely used in many aspects. For that reason, we need to research this technology in order to get high performance. This research of facial expression detection will use Local Phase Quantization (LPQ) and Pyramid of Histogram of Gradients (PHOG) and Residual Network (ResNet) as feature extraction and Bi-Directional Neural Networks (BDNN) and Class Prototypes as the classifier. We use SFEW-SPI dataset as the baseline to create the automated model. We found that this BDNN method has better performance than using non-linear SVM with the same dataset and extraction features. Moreover, although this extension of BDNN method has lower performance than conventional BDNN and forward-pass neural network, this model is less overfit than these conventional methods.

Keywords: Bi-directional Neural Network, Class Prototype, Facial Expression, LPQ, Neural Network, PHOG, ResNet

1 Introduction

The technology to detect facial expression become popular nowadays. This technology will be used in various aspects such as marketing (the salesman can detect better the emotions of the customers), entertainments (games and movies), and even politics (lie detection to the politicians during speeches towards certain issues). Because of that reasons, we need to ponder how to make the facial expression technology that have a good performance.

A facial expression is formed by the positions or motions of the muscles in the skin's face. These motions might reveal the emotions of a person. This type of expression is one of a non-verbal communication which contributes to the human interactions and socialisations. These expressions not only appear in human but also other species as well.

Nowadays, computer visions can detect this type of emotions. Different expressions create different pattern on every parts of faces including but not limited to the changes of eyes and mouths, the muscle movements, and wrinkles. The computer vision technique, with the help of machine learning, learns from various patters so that computer vision, in theory, is possible to detect this facial expression uniqueness.

We need to implement feature extractions because these facial expressions are captured in the form of images or videos which categorised as unstructured data. Machine learning technique finds these patterns only from the structured data so that this feature extraction is needed to convert the images into representable vectors or descriptors. In this research, we use two feature extractions, local phase quantisation (LPQ) and pyramid of histogram of oriented gradients (PHOG). These methods are commonly used for obtaining some important features of the images in many tasks in which one of them is detecting the facial expression [1], [2]. In another experiment, we also implement one of deep learning technique, that is residual neural network (ResNet), as both classifier and feature extraction in separate experiment. The ResNet approach have been used in many image classification tasks and has good performance in classifying object [7] and detecting facial expression [8], [9].

For the machine learning technique, we use Bi-directional Neural Network (BDNN) to detects the facial expressions from the generated vectors from the feature extractions. Unlike traditional neural network which only performs one direction (with the other direction only do the backpropagation), this method performs in both direction for forward pass as well as the backpropagation as illustrated in Fig. 1. The idea behind this method is making a neural network behaves similar with the generative models which not only to learn to make prediction but also generate a new data [3]. The prediction and the generated data will be compared with the true labels and original data respectively in order to get the model performance. This method is believed to be more robust compared with traditional neural network [3].

In this research, we use two improved BDNN methods such as BDDN as content addressable memory and as cluster points finders. This model enables the casual effects between output-input as well as input-output so that we can get more accurate extracted rules [3]. These approaches have a good performance in some real-world tasks which mainly have noisy attributes that are *Student Final Marks* (SFM) dataset and UNCTAD *Gross National Product* (GDP) dataset [4].

We use *Static Facial Expressions in the Wild* (SFEW) database with *Strictly Person Independent (SPI) Protocol*. In this paper, we use 675 images with 7 different expressions (angry, disgust, fear, happy, neutral, sad, and surprised). The data should contain 700 images, but we miss 25 images with disgust expressions. Later on, these images are extracted using LPQ and PHOG then we obtain the first 5 principal component of each descriptors.

This research will demonstrate how well the BDNN with class prototypes predict the facial expression given the SFEW-SPI datasets by using LPQ and PHOG as well as ResNet. We will compare the performance this type of BDNN

with other models with some statistical measurements. Moreover, we will discuss what are the aspects that taken into account these BDNN method from those performance results.

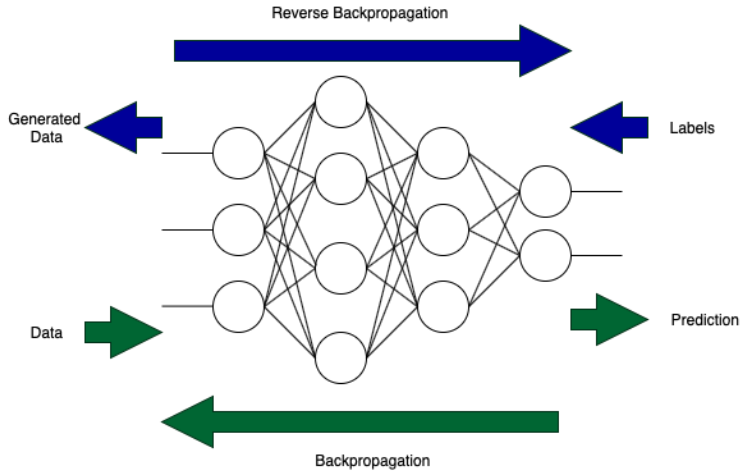

Fig. 1. BDNN architecture which use not only forward direction but also reversed direction as well. The forward direction gives prediction results while the reversed architecture gives a new generated data

2 Feature Extraction

In this research, we use three feature extraction methods, Local Phase Quantisation (LPQ) and Pyramid of Histogram of Gradient (PHOG), and the features generated from Residual Network (ResNet).

2.1 Local Phase Quantisation (LPQ)

Local Phase Quantisation (LPQ) is an improvement of Local Binary Pattern (LBP). The LBP obtains binary values by applying thresholding the pixels around the centre of LBP windows [2]. However, LPQ compute the local phase information by using short-term Fourier Transform (STFT) from the rectangular neighbourhood of each pixel [2]. LPQ uses four different frequency points for each pixel position. Then, we examine the signs of real and imaginary parts of each frequency, denoted as g_i, by using simple scalar quantisation in the Equation (1).

$$e_i = \begin{cases} 1 & \text{if } g_i \geq 0 \\ 0 & \text{if } g_i < 0, \end{cases} \tag{1}$$

After that, the value of LPQ for each pixel position is computed all eight e_i by using Equation (2).

$$f_{\mathrm{LPQ}}(x, y) = \sum_{i=1}^{8} e_i(x, y)\, 2^{i-1}. \tag{2}$$

2.2 PHOG

Pyramid of Histogram of Gradient is the extension of Histogram of Gradient which counts the orientation of gradient occurrences in localised portions of an image. PHOG, however, take the spatial property which represented by “tiling the image into regions at multiple resolutions based on spatial pyramid matching” [5]. This PHOG method have been implemented in various computer vision tasks [5], [6].

2.3 Residual Neural Network (ResNet)

The problem in CNN could be on the vanishing gradients or exploding gradient problem even though the use of Batch Normalization ensures that the gradients have healthy norms. Thus, the deeper networks may fail to perform better than

the shallow networks. The author of [7] propose a solution to this problem in training deep network by using Residual Block. This Residual Block implement additions from the output of the previous layers (x) and the output of the next layers ($F(x)$). The illustration of the Residual Block is shown in Fig. 2.

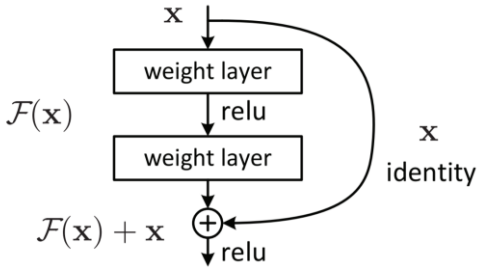

Fig. 2. Residual Block. Implementing another operation that is adding the previous value (x) and the result some layer function ($F(x)$).

Sometimes the value of x and $F(x)$ will not have the same dimension. Then, we can perform a linear projection W to expand the channels of shortcut to match the residual as shown in Equation 3. This skip connection between layers can allow the ability to train much deeper networks than conventional CNN as this approach add the outputs from previous layers to the outputs of stacked layers.

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}. \tag{4}$$

In this experiment, we use two ResNet architecture that are ResNet18 and ResNet34 which is shown in Table 1. The building blocks are shown in brackets with some certain numbers. The down sampling operation is performed in the conv3_1, conv4_1, and conv5_1 with stride of 2.

Table 1. Architectures for ImageNet. Building blocks are shown in brackets with the numbers of blocks stacked. Down sampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2

Layer name	Output size	ResNet-18	ResNet-34
Conv1	112×112	7×7, 64, stride 2	
Conv2_x	56×56	3×3 max pool, stride 2	
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times 2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times 3$
Conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times 2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times 4$
Conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times 2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times 6$
Conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times 2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times 3$
	1×1	average pool, 1000-d fc, softmax	

3 BDNN as Class Prototypes

This research will use the extension of Bi-directional Neural Network (BDNN) that is proposed by Nejad [4]. Nejad [4] believes that there are some statistical information that the output gives the unique effect to the input. More information can be extracted if we enable the casual effects between output-input as well as input-output. In this research, we use two BDNN approaches from [4]: BDNN as the content addressable memories and as cluster centroids finders.

3.1 BDNNs as Content Addressable Memories

In this method, we use the model whose input and output vectors forms a one-to-one property [4]. We know that machine learning can work for one or more input which has the same output value (many-to-one). But the reverse pass in BDNN, one input vector has many output values (one-to-many). Because of that reason, we perform one-to-one relation so that the BDNN model for both forward and reverse pass will behave like other machine learning techniques that can predict the output from the unique inputs.

However, not all the input and output vectors in dataset have one-to-one property. In order to get a one-to-one property between the input and output vectors, we need to add an extra node in the output layers [4]. We assign a random value to this extra node so that the input vectors maps to the new output vectors with one-to-one relation. After that we train this model similarly as conventional BDNN.

3.2 BDNNs as Cluster Centre Finders

We know that the reverse pass of conventional BDNN maps one input to many output values. Because of that, without any modification, the output of conventional BDNN gives the expected values of features for each label [4]. We can say that these vector values can be represented as the middle points of each label if the datasets follow an even distribution [4].

We obtain these cluster points by performing the reverse pass from the BDNN. We assign the one-hot-encoded of each label as input of this reverse pass BDNN. From this, we can get the output features of each label and we set it as cluster centre points. Finally, we get the labels of each data points in the dataset by finding the closest cluster points that affiliate to certain label.

4 Implementation

We use SFEW Dataset by using SPI Protocol which contains 675 images with seven labels (angry, disgust, fear, happy, neutral, sad, and surprised). Then we perform feature extraction by using LPQ and PHOG and only capturing the first five principal components of each features. In other words, we have ten features consists of five LPQ features and five PHOG features.

Then we do the pre-processing from these features. First, we split the dataset into training set and testing set. After splitting, we reduce the dataset by deleting the nan value and outliers. The method of finding the outlier is using Z-score then we filtered these data whose rows have Z-score between -3 and 3. Later, we implement standardization to both sets. However, we use the mean and standard deviation of each feature in training set to be applied on the standardization for the testing set.

We perform four different architectures: one directional neural network (traditional neural network), BDNN, BDNN as the context of addressable memory, and BDNN as cluster finders. All architectures have two hidden layers. The first experiment we use 30 and 15 nodes as the first and second hidden layers respectively. Then the second experiment we use 50 and 20 nodes respectively. We use an activation function Relu in hidden layers, softmax function in the output of forward pass, and linear function in the output of the backward pass. For BDNN, we change the direction for every 50 epochs. Here we use Adam optimizer with weight decay as the optimizer to update the weights.

Lastly, we calculate the performance of these architectures on training and testing data separately. These performance criteria are accuracy, precision, recall, F1-score, and mean squared error (MSE) loss for both forward and reversed architectures (if any) which use the formula of (4), (5), (6), (7), and (8) respectively.

$$\text{Accuracy} = \frac{tp+tn}{tp+fp+fn+tn} \tag{4}$$

$$\text{Precision} = \frac{tp}{tp+fp} \tag{5}$$

$$\text{Recall} = \frac{tp}{tp+fn} \tag{6}$$

$$\text{F1-Score} = \frac{2}{\frac{1}{Precision}+\frac{1}{Recall}} \tag{7}$$

$$\text{MSE Loss} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i}) \tag{8}$$

Here, tp is true positive, tn is true negative, fp is false positive, and fn is false negative. Moreover, in the MSE Loss, N is number of data, y_i is the true labels (or true data in reversed architecture case) and $\hat{y}_i$ is the predicted labels (or the generated data in reversed architecture case).

We also use two ResNet model, ResNet18 and ResNet34, as the comparator. At first, we train the ResNet normally using linear function as the fully connected network. After that, we use the 512 extraction features from these ResNet model (The ResNet features before the fully connected network) to be trained similarly with the implementation above (one directional neural network (traditional neural network), BDNN, BDNN as the context of addressable memory, and BDNN as cluster finders). Because we have 512 features, we add another experiment that is using 300 and 100 nodes respectively in the two hidden layers model.

In ResNet approach, we do pre-process first to all the images. First, we resize the images from 720×576 to 320×256 then we crop the images to 224×224. Then, we normalise the images by using ImageNet standards that is using mean equals to [0.485, 0.456, 0.406] and standard deviation equals to [0.229, 0.224, 0.225]. After that we perform the methods of both ResNet as classifiers and as features.

5 Experiment results and Discussion

In all different number of hidden neurons, traditional NN has the best performance on training data compare with other models but the improved BDNN can handle the overfitting better than other models for all extraction features methods. As we can see in the Table 2,

Table 3, and Table 4, the conventional NN has the highest training performance while conventional BDNN, BDNN with content addressable memory, and BDNN as cluster points relatively are in second, third, and forth place. In Table 2,

Table 3, and Table 4, we can see that NN has the highest testing performance by using ResNet features while conventional BDNN, BDNN with content addressable memory, and BDNN as cluster points relatively are in second, third, and forth place. However, conventional BDNN become the highest testing performance if we use LPQ-PHOG features while conventional NN, BDNN as cluster points, and BDNN with content addressable memory are in second, third, and forth place as we can see in Table 2,

Table 3, and Table 4. Then the models that use ResNet as features, has higher performance when we use traditional NN compared with the LPQ-PHOG features but has lower performance when we use BDNN. From these tables, although BDDN as cluster points generally has low performance, but this architecture handles the overfitting better than other models and ResNet features perform well if we use conventional NN but not for BDNN.

Table 2. Performance (Accuracy, Precision, Recall, F1-Score, Loss, and Loss for reversed model) of all different type of models with two layers of 30 and 15 nodes**.** We can see from the table, traditional NN has the best performance on training data compare with other models but the improved BDNN can handle the overfitting better than other models for all extraction features methods. Then the models that use ResNet as features, has higher performance when we use traditional NN compared with the LPQ-PHOG features but has lower performance if we use BDNN.

		LPQ-PHOG				ResNet18 as features				ResNet34 as features			
		NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP
Training	Acc (%)	86.22	84.84	65.94	26.38	99.31	64.20	33.49	23.79	96.33	52.48	28.08	21.17
	Prec (%)	88.20	84.72	67.57	26.02	99.26	67.98	36.93	24.12	96.59	58.25	31.00	22.39
	Rec (%)	85.84	84.80	66.00	26.42	99.31	63.62	33.63	24.42	96.34	52.83	27.26	22.15
	F1 (%)	86.58	84.58	65.83	25.44	99.27	64.04	32.92	22.70	96.38	52.70	26.00	19.43
	Loss	0.0255	0.0309	0.0693	0.2103	0.0013	0.0822	0.1377	0.2177	0.0064	0.1011	0.1458	0.2252
	Loss Rev	-	0.9760	0.9423	-	-	0.9681	0.9171	-	-	0.9844	0.9095	-
Testing	Acc (%)	25.37	28.36	21.64	22.39	27.43	16.81	15.93	20.35	23.08	22.22	11.97	18.80
	Prec (%)	22.85	28.25	19.94	22.49	26.49	17.29	18.72	26.54	22.88	23.42	15.03	26.70

	Rec (%)	24.92	28.48	21.67	22.89	29.70	18.75	15.22	21.36	23.72	23.32	12.37	19.76
	F1 (%)	22.80	27.85	20.20	19.98	27.71	16.92	15.58	20.40	22.50	22.69	10.68	18.03
	Loss	0.1986	0.1867	0.1715	0.2217	0.1801	0.1979	0.1936	0.2276	0.1941	0.1778	0.2144	0.2320
	Loss Rev	-	0.9539	0.9716	-	-	0.9396	0.9424	-	-	1.1361	1.2086	-

Table 3. Performance (Accuracy, Precision, Recall, F1-Score, Loss, and Loss for reversed model) of all different type of models with two layers of 50 and 20 node**s**. We can see from the table**,** traditional NN has the best performance on training data compare with other models but the improved BDNN can handle the overfitting better than other models for all extraction features methods**.** Then the models that use ResNet as features, has higher performance when we use traditional NN compared with the LPQ-PHOG features but has lower performance if we use BDNN.

		LPQ-PHOG				ResNet18 as features				ResNet34 as features			
		NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP
Training	Acc (%)	93.50	90.55	72.24	26.57	99.54	62.36	33.26	23.79	99.78	67.17	34.13	21.17
	Prec (%)	93.58	90.74	72.31	26.22	99.48	69.39	49.40	24.11	99.73	71.14	42.56	22.28
	Rec (%)	93.55	90.38	70.61	26.76	99.56	61.92	31.76	24.42	99.79	66.46	33.26	22.15
	F1 (%)	93.27	90.46	70.53	25.63	99.51	63.80	30.12	22.70	99.76	67.82	32.80	19.43
	Loss	0.0117	0.0175	0.0572	0.2098	0.0008	0.0867	0.1247	0.2177	0.0004	0.0742	0.1305	0.2252
	Loss Rev	-	0.9760	0.9277	-	-	0.9681	0.8733	-	-	0.9844	0.9183	-
Testing	Acc (%)	23.88	29.85	29.10	21.64	25.66	15.93	16.81	20.35	23.93	23.08	21.37	18.80
	Prec (%)	23.53	27.48	31.17	21.27	25.12	15.53	15.68	26.54	25.08	24.71	25.52	26.68
	Rec (%)	24.44	29.44	29.39	22.17	26.41	16.11	17.89	21.36	24.63	23.74	20.56	19.76
	F1 (%)	23.58	27.56	27.56	19.08	25.36	15.49	14.37	20.40	24.15	22.80	19.69	18.00
	Loss	0.1978	0.1816	0.1636	0.2239	0.1719	0.2053	0.1923	0.2276	0.1846	0.1789	0.2038	0.2320
	Loss Rev	-	0.9539	1.0116	-	-	0.9396	1.0172	-	-	1.1362	1.1763	-

Table 4. Performance (Accuracy, Precision, Recall, F1-Score, Loss, and Loss for reversed model) of all different type of models with two layers of 300 and 100 nodes. We can see from the table, traditional NN has the best performance on training data compare with other models but the improved BDNN can handle the overfitting better than other models for all extraction features methods. Then the models that use ResNet as features, has higher performance when we use traditional NN compared with the LPQ-PHOG features but has lower performance if we use BDNN**.**

		LPQ-PHOG				ResNet18 as features				ResNet34 as features			
		NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP	NN	BDNN	BDNN as CAM	BDNN as CP
Training	Acc (%)	98.43	99.02	84.84	26.97	100.00	86.84	57.51	23.79	99.78	72.79	58.10	21.17
	Prec (%)	98.67	98.90	85.19	26.58	100.00	88.68	70.68	24.11	99.81	74.97	66.11	22.51
	Rec (%)	98.51	99.06	84.62	27.09	100.00	85.83	55.86	24.42	99.81	71.96	57.88	22.15
	F1 (%)	98.52	98.97	84.70	26.05	100.00	86.54	56.95	22.70	99.81	72.54	59.02	19.43
	Loss	0.0026	0.0016	0.0418	0.2087	0.0000	0.0313	0.0922	0.2177	0.0004	0.0583	0.0892	0.2252
	Loss Rev	-	0.9760	0.8332	-	-	0.9680	0.8310	-	-	0.9844	0.8443	-

Testing	Acc (%)	26.87	25.37	29.10	21.64	21.24	18.58	12.39	20.35	26.50	16.24	17.09	18.80
	Prec (%)	27.02	24.86	28.53	21.97	20.10	17.88	13.80	26.54	27.48	18.94	14.76	26.68
	Rec (%)	26.82	25.48	29.56	22.17	22.48	19.08	14.02	21.36	27.64	16.90	16.76	19.76
	F1 (%)	26.24	25.01	27.81	19.58	20.76	17.93	11.98	20.40	26.87	16.77	14.05	18.00
	Loss	0.1775	0.1803	0.1859	0.2239	0.1701	0.1827	0.2283	0.2276	0.1619	0.1818	0.2143	0.2320
	Loss Rev	-	0.9539	1.0879	-	-	0.9398	1.0475	-	-	1.1361	1.2636	-

We also compare the performance between using these feature extractions and directly trained using deep learning approaches, namely ResNet-18 and ResNet-34. From Table 5, we can see that ResNet-18 has better accuracy than the ResNet-34 in both training and testing. However, the loss value for the testing in ResNet-34 is lower than the ResNet-18, meaning the difference between loss value of correct class and one of the wrong class is close enough in ResNet-34. However, because we choose the highest loss as the predicted class, the model is likely to choose the wrong class as its loss is slightly higher than the correct class.

Table 6 shows the accuracy between all models. For model that use features, we average the performance between all different hidden layers. From this table, we can see that conventional neural network with ResNet-18 features has the highest training accuracy but ResNet-18 has the best testing accuracy. In addition, the performance of ResNet-18, although has lower training accuracy, the testing accuracy is the highest among all methods, meaning ResNet-18 as classifiers has better performance at handling the overfitting.

Table 5. Performance (Accuracy, Precision, Recall, F1-Score, Loss, and Loss for reversed model) of ResNet classifiers. we can see that ResNet-18 has higher accuracy than the ResNet-34 in both training and testing. However, the loss of ResNet-34 using testing data is better than ResNet-18.

		ResNet-18	ResNet-34
Training	Acc (%)	43.15	38.70
	Prec (%)	48.55	42.23
	Rec (%)	42.16	36.54
	F1 (%)	39.41	31.57
	Loss	0.0389	0.0429
Testing	Acc (%)	38.70	22.96
	Prec (%)	42.23	16.13
	Rec (%)	36.54	24.21
	F1 (%)	31.57	17.25
	Loss	0.0429	0.0124

Table 6. The accuracy of these four different architectures on training and testing set in different scenarios. we can see that conventional neural network with ResNet-18 features has the highest training accuracy but ResNet-18 has the best testing accuracy**.** we also can see that generally the ResNet has lower training performance than the methods that use feature extractions

		Training	Testing
LPQ-PHOG	NN	92.72	25.37
	BDNN	91.47	27.86
	BDNN as CAM	74.34	26.61
	BDNN as CP	26.64	21.89
ResNet	ResNet-18	43.15	38.70
	ResNet-34	38.70	22.96
	NN	99.61	24.78

ResNet-18 as features	BDNN	71.13	17.11
	BDNN as CAM	41.42	15.04
	BDNN as CP	23.79	20.35
ResNet-34 as features	NN	98.63	24.50
	BDNN	64.15	20.51
	BDNN as CAM	40.10	16.81
	BDNN as CP	21.17	18.80

From Table 6, we also can see that generally the ResNet has lower training performance than the methods that use feature extractions. However, the testing performance of ResNet which is directly trained using the raw images is slightly higher. In other words, using ResNet as a classifier directly from raw images gives the less overfit results than other methods that separate the process by using feature extractions.

The testing accuracy of these models are relatively low but this accuracy still higher than using non-linear SVM [2]. [2] implements similar dataset and feature extractions with the accuracy 19%. As stated by [2], the complexity of this dataset contributes to this low performance. In other words, all of the models performed in this experiment have higher performance than non-linear SVM.

From the result from the models that use feature extractions, the forward pass is the main cause of overfitting and the backward pass helps reducing the overfitting. The conventional NN performs only forward direction (and only backward propagation); as a result, the model is drastically overfit. The generative model (the reverse direction) in BDNN helps reducing this overfitting. BDNN with content addressable memory is better handling overfitting than BDNN because we apply pre-processing so that this model become a one-to-one function. This one-to-one behaviour helps the generative process works better because each input (right sides) has only one relation with the output (left sides). BDNN which is used to find the cluster points gives the most realistic results because we do not apply any forward NN approach in this model. We can see that the forward direction of neural networks contributes the most to the overfitting and the backward pass helps prevent the overfitting. Nevertheless, the ResNet as classifier method has higher performance than the BDNN and better at handling overfitting.

6 Conclusion

We have seen BDNN can be trained by applying it as the associative memories and cluster centre finders. We also implemented these models by using LPQ and PHOG features and ResNet features which extracted from SFEW-SPI dataset. Before we had implemented the models, we did the pre-processing to the images that are resizing, cropping, and normalising the images. We also do pre-process to these extracted features which consists of removing nan, deleting the outliers, and standardise the data.

From the experiment results, we can see that this extension of BDNN contribute much to generalise the error if we apply the feature extraction approaches. The overfitting can be reduced by implementing the reverse pass to the neural network models although the performance is degraded. The BDNN that has one-to-one relation between its input and output helps the reverse pass to perform better. Then, we can use the result of backward pass of conventional BDNN to be cluster centres of each label. As we can see from the experiment results, we get less overfit result by obtaining the labels using these cluster centres even though this model has lower performance. However, the ResNet approach as classifiers not only better at generalisation but also the accuracy performance compared with the BDNN.

7 Future Work

We need to find a proper function in the reverse pass of the BDNN. In this experiment, we use linear activation function in the output of reverse pass because the range of the data is varied unlike the labels that are binary values. We also apply MSE loss function because the loss will be more than zero when applying reverse pass. We have implemented various loss functions, but the loss result is in negative value. Because of those problems, we need to find appropriate activation function to the output of reverse pass and loss function as well.

Although the BDNN with class prototypes has lower performance than the ResNet, the performance can be improved if we can apply the combination between ResNet and BDNN that is Bi-directional Residual Network. We know that the implementation of ResNet normally do in forward pass. With combining these approaches, we can get better performance results to detect the facial expression as well as give a less-overfit performance.

These BDNN methods can help us to understand the black-box behaviour of the neural network. We can use the extracted features generated by the reverse pass of BDNN because these features can help us to know what the neural

network did during the training process. Moreover, these extracted features can helps us understand the behaviour of the dataset as well [4].

References

1. Dhall, A. Asthana, A. Goecke, R. Gedeon, T.: Emotion recognition using PHOG and LPQ features. 2011 IEEE Int Conf Autom Face Gesture Recognit Work FG 2011, pp. 878--883 (2011)
2. Dhall, A. Goecke, R. Lucey, S. Gedeon, T.: Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. Proc IEEE Int Conf Comput Vis, pp. 2106--2112 (2011)
3. Pontes-Filho, S. Liwicki, M.: Bidirectional Learning for Robust Neural Networks. Proc Int Jt Conf Neural Networks 2019-July: (2019)
4. Nejad, AF. Gedeon, TD.: Bidirectional neural networks and class prototypes. IEEE Int Conf Neural Networks - Conf Proc 3, pp.1322–1327 (1995)
5. Huang, HM. Liu, HS. Liu, GP.: Face recognition using pyramid histogram of oriented gradients and SVM. Adv Inf Sci Serv Sci 4, pp.1--8 (2012)
6. Bosch, A. Zisserman, A. Munoz, X.: Representing shape with a spatial pyramid kernel. Proc 6th ACM Int Conf Image Video Retrieval, CIVR 2007, pp. 401--408 (2007)
7. He, K. Zhang, X. Ren, S. Sun, Jian.: Deep Residual Learning for Image Recognition. 2016 IEEE Conf on Comput Vis Pat Recog, pp. 770--778. Las Vegas (2016)
8. Cai, Jie & Meng, Zibo & Khan, Ahmed Shehab & Li, Zhiyuan & O'Reilly, James & Tong, Yan.: Probabilistic Attribute Tree in Convolutional Neural Networks for Facial Expression Recognition (2018)
9. Otberdout, N., Kacem, A., Daoudi, M., Ballihi, L., & Berretti, S.: Automatic Analysis of Facial Expressions Based on Deep Covariance Trajectories. IEEE transactions on neural networks and learning systems (2019)