Deep Convolutional Neural Network Based Facial Keypoints Detection

Madhuchhanda Dasgupta[1(✉)] and Jyotsna Kumar Mandal[2]

[1] Department of CSE, JIS College of Engineering, Kalyani, India
madhu.banik@gmail.com
[2] Department of CSE, Kalyani University, Kalyani, India
jkm.cse@gmail.com

Abstract. Facial keypoints (FKP) detection is considered as a challenging task in the field of computer vision, as facial features vary from individual to individual. It becomes a more challenging proposition as the same person facial image may also vary due to change in position, size, pose, expression etc. Some methods exist in literature for detection of FKPs. In this paper, a deep architecture is used to locate the keypoints on gray-scale images. As baseline method one hidden layer neural network and convolutional neural networks are built in the proposed work. Additionally, a block of pretrained Inception module is used to extract the intermediate features. Specifically, the sparse structure of Inception model reduces the computational cost of the proposed method significantly. The methods are evaluated on standard dataset and compared with existing state-of-the-art CNN based methods. The obtained results are promising and also bring out the efficiency of the proposed work.

Keywords: Facial keypoints · Convolutional neural network · Gray-scale image · Sparse structure · Inception model

1 Introduction

People usually do face recognition effortlessly without much consciousness but it remains a challenging task yet in the field of computer vision. Face recognition is identifying an individual from the face images. With technological improvement it has been widely applied in our daily life such as tracking faces in images, biometrics, information security, health care, access control, law enforcement and surveillance systems.

In face recognition, Facial keypoints (FKP) are used as a building block. This keypoints detection is to predict certain positions such as corners of the eyes, eyebrows, nose and mouth on face images. Over the last decade lots of work has been done in face detection and recognition. PCA, LDA, GABOR, LBP etc. are the traditional face recognition algorithms. Human neurons inspired artificial neural network is an adaptive system based on learning procedure that provides high accuracy. Convolutional Neural Networks (CNN) provides higher accuracy in computer vision field. Nowadays, CNN are the state-of-the-art performers for a wide variety of tasks. This model is also suitable for facial landmark detection with fast convergence and high accuracy.

J. K. Mandal et al. (Eds.): CICBA 2018, CCIS 1030, pp. 47–56, 2019.
https://doi.org/10.1007/978-981-13-8578-0_4

In this paper, first Neural Network (NN) and then CNN based approaches are built as baseline methods to predict FKPs accurately and effectively.

In computation, efficient distribution of computing resources with finite computational budget is very necessary. To remove computational bottleneck and dimension reduction, Szegedy et al. [8] proposed Inception model which has low computational cost for its sparsely connected architecture. In proposed CNN this sparse architecture is added to reduce the number of computations effectively.

In the remaining part of the paper, literature review is provided in Sect. 2. In Sect. 3, proposed baseline methods are explained and experimental result analysis is given in Sect. 4. Section 5 concludes the document.

2 Related Work

With growing applications, huge work has been done in past on FKPs detection. Traditional methods have used feature extraction and different types of graphic models to detect facial keypoints. Vukadinovic [11] proposed Gabor feature based boosted classifiers to detect 20 different facial keypoints. Boosted regression and graph modes based method is presented by Valstar et al. [12]. Belhumeur et al. [14] used local detectors with a non-parametric set of global models for the part locations of faces. In some applications shape models and branch and bound are used for optimal landmark detection [13]. On the other hand, Wang et al. [4] addressed FKPs detection by applying histogram stretching for image contrast enhancement, followed by principal component analysis [5]. Cao et al. [26] proposed a generative Bayesian transfer learning algorithm for the face verification problem.

In the last few years, CNN has shown outstanding results and it also shows rapid advances in FKPs detection. Several papers propose CNN learning for FKPs detection [6, 7, 10, 21]. Deep neural network is used in face recognition [15, 25] as well. Shi [1] applied and discussed different algorithms starting from K Nearest Neighbours (KNN), linear regression, Decision tree to CNN for locating FKPs. Longpre, Sohmshetty [7] applied data augmentation techniques to expand the number of training examples to generalize the network for keypoints detection. Sun et al. [4] estimated FKPs by using three level convolutional neural network and outputs of multiple networks were merged for robust and correct estimation. They extracted high-level features over the whole face to locate high accuracy keypoints. Again the geometric ratios and constraints of keypoints are used to train the network to predict the keypoints simultaneously. Zhang and Meng [6] used a sparsely connected inception model to extract features and input those features to CNN to reduce computational complexity for detecting FKPs. In the proposed work similar approach is adopted for detection with enhancements.

3 Proposed Methods

3.1 Dataset

Dataset is provided by Dr. Yoshua Bengio, University of Montreal. Now this dataset is available in Kaggle open research dataset [2]. It has 7049 training data of gray-scale facial images and their corresponding 15 FKPs. Here, image dimension is 96 × 96 that is 9216 pixels and each pixel is represented by 8-bits. All 15 keypoints are in 2D images with [x, y] co-ordinates. So, 30 numerical values are to be predicted. The given dataset has 7049 rows with 31 columns where in each row first 30 columns represent 15 FKPs and last column has 9216 pixel values to represent one image. All 15 facial keypoints are shown in Fig. 1. Among the 7049 training images, there exist only 2140 training images which are completely and accurately labelled for all the 15 keypoints. To expand the training dataset, data augmentation technique is applied. 80% of the data is considered as training data and 20% data is used for validation purpose. The proposed model tested on 1783 test images.

Fig. 1. Augmented face with 15 facial keypoints

3.2 Data Augmentation

To train the convolutional neural networks, a large number of training samples are required to avoid over-fitting. Data Augmentation technique is applied to generate more image data to increase accurately labelled training dataset. Among most effective data augmentation techniques, horizontal reflection is one of them. It is quite straight forward technique in which the images are flipped horizontally with their keypoint labels and then remap the keypoint labels to their new representations so that left eye center becomes right eye center and vice versa. In this way, almost double amount of data are generated and creates augmented training dataset.

3.3 Network Architecture

Neural Network (NN)
Artificial neural networks are popular machine learning techniques [20]. As a baseline model a neural network is designed using Keras which is a Python based high-level neural network library. In the neural network, first reshape the input image of 96 × 96 into 9216 × 1 as the input of the network, then one hidden layer consists of 500 neurons and the output layer with 30 units as 15 sets of coordinates for 15 FKPs of facial images. The loss function is defined by mean square error (MSE) between the actual value and the obtained output from the keypoints vectors. To converge gradient descent faster, Nesterov momentum is used as an update rule. During the training phase, the network iterates for 400 epochs.

Convolutional Neural Network (CNN)
In the proposed work a convolutional neural network [23] is designed to achieve high accuracy. The main reason of popularity of CNN is its algorithm and improved network architecture. CNN has a typical structure – stacked convolutional layers followed by one or more fully connected layers [19]. In Convolutional layers feature maps are comprised of convolution operations followed by max pooling [9, 17]. The feature maps of CNN consists with only considering receptive fields so that smaller numbers of neurons are present in comparison with fully connected neural network and this is the unique property of CNN.

The proposed CNN architecture and hyper-parameters of it are given in Fig. 2 and Table 1 respectively. The layers of proposed CNN model are structured as following.

- As input size of the image is 96 × 96 and being a gray-scale image, the input layer size is 96 × 96 × 1.
- First convolutional layer consists of 32 filters sized 3 × 3, stride 1 followed by max pooling of filter size 2 × 2 with stride 2.
- In second conv. layer 64 filters of size 3 × 3, stride 1 followed by max pooling 2 × 2 with stride 2.
- Third conv. layer consists of 128 filters of size 3 × 3, stride 1 and max pooling filter 2 × 2 with stride 2.
- The fourth conv. layer forms with 256 filters with dimension 3 × 3, stride 1 followed by max pooling 2 × 2 with stride 2.
- Next two fully connected layers consist with 256 and 128 hidden units respectively.

Finally output layer consists of 30 units. A set of coordinates [x, y] of 15 facial keypoints are stored on these 30 units [18].

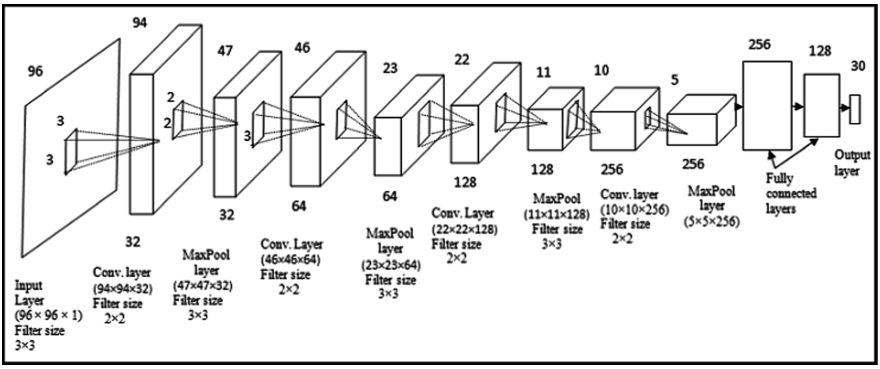

Fig. 2. Convolutional Neural Network architecture

Table 1. Hyper-parameters of CNN

	Convolution layer 1		Convolution layer 2		Convolution layer 3		Convolution layer 4	
	Conv	Pool	Conv	Pool	Conv	Pool	Conv	Pool
No. of filters	32	32	64	64	128	128	256	256
Filter size	3 × 3	2 × 2	3 × 3	2 × 2	3 × 3	2 × 2	3 × 3	2 × 2
Stride	1	2	1	2	1	2	1	2
Pad	0	0	0	0	0	0	0	0

The whole network is trained through backpropagation method and trying to minimize the error by using optimization algorithm. To evaluate the gradient of the error function initially high learning rate is set and gradually minimize the learning rate to get closer point of minimum loss to converge the model for an optimal set of weights. The network is tuned using different parameters and update rules [28] to make an optimal model. Deep networks with a large number of parameters are very powerful machine learning systems. But combining all parameters occurs overfitting which slow down the system. To address such problem dropout [16] is introduced which randomly drop units along with their connections from the neural network during training phase.

Inception Convolutional Neural Network (INCNN)

In multi-layer CNN architecture due to increased number of layers and parameters, the network is inclined to overfit and large numbers of computational resources are consumed. To overcome these, some adjustments are needed in the architecture level. Szegedy et al. [8] proposed Inception Model which considers sparsely connected architecture in the convolutions where lower dimension filters are used for convolution

operations and after that they are concatenated to form expected size filter for feature extraction.

In Inception module, 1 × 1 convolutions are applied before expensive 3 × 3 and 5 × 5 convolutions to reduce computational cost. Besides dimension reduction, they also include rectified linear activation to add non-linearity to the network. In Inception network, modules are stacked one upon another and max pooling layers is applied to reduce the grid size.

In a typical block of Inception Model shown in Fig. 3, multiple filters with different sizes 1 × 1, 3 × 3 and 5 × 5 are used and concatenated to form the next layer unlike one filter in traditional CNN layers. This process achieves global sparsity as one filter splits into several groups corresponding to different filter sizes. Additionally, 1 × 1 filters are used in front of all filters which drastically reduce computational cost. In the following experiments, pretrained Inception model is adopted to extract features and input these features into CNN architecture and this is named as INCNN in the proposed work.

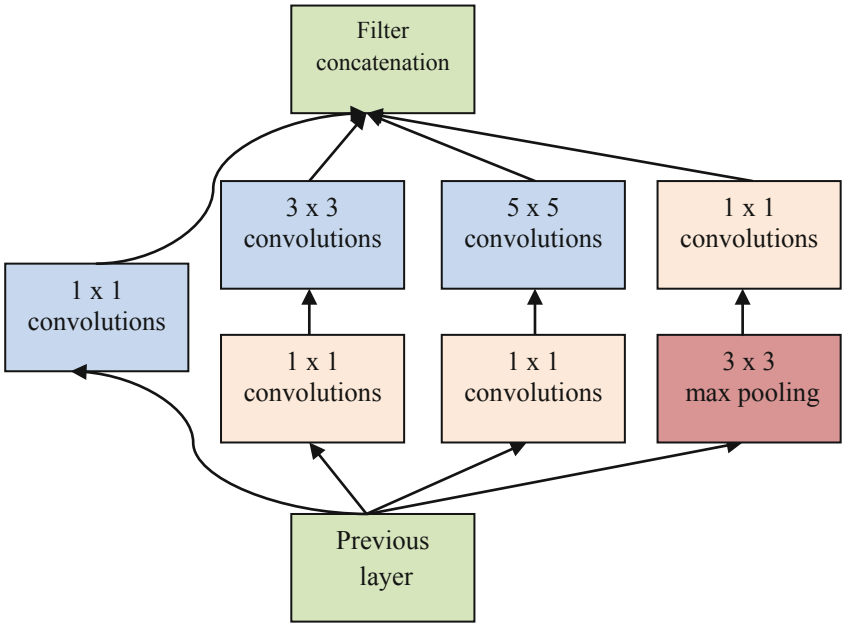

Fig. 3. Inception module with dimension reductions

4 Results

4.1 Experimental Setup

The experimental platform is Intel Core i5 Processor 2.2 GHz CPU plus 8 GB memory laptop. In the framework TensorFlow based model is used for building convolution neural network. In addition, the following packages are used in the proposed algorithm: Pandas, Numpy, Keras, Scikit, Matplotlib and main language Python is used for implementation of the algorithm.

4.2 Accuracy Detection

In this section, the performance of the proposed algorithm in terms of detection accuracy is discussed. Root Mean Square Error (RMSE) is an effective measure of the deviations in distances, has been used to compare accuracy measures of proposed method with other existing standard methods.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Root Mean Square Error formula is shown above. The loss is calculated between the real and predicted FKPs. Here n = 30 given and y_i is the expected output and $\hat{y}_i$ is the predicted value of the i-th keypoint. Each model is evaluated with different update rules [22] and number of epochs and adjusts the hyper-parameters accordingly to achieve best possible RMSE.

A large learning rate caused loss to explode whereas too small learning rate showed a stagnant loss curve. Here 0.1 is chosen as initial learning rate and gradually decreases it to 0.001 for convergence of gradient descent [27]. To add non-linearity in the network Rectified Linear Unit (ReLU) activation function is used as it is observed that it works faster in deep convolutional neural network than equivalent tanh units [3]. Adam optimizer [24] is chosen for the network as in deep learning applications it performs best compared with other optimization algorithms. Batch size is set as 128 and epochs count set as 400 here. During experiment a large gap is observed in between training and validation errors. Regularization is added by using dropout value of 0.2 to minimize this gap.

Data augmentation technique is applied in the training dataset and shows a significant improvement in validation RMSE. In Table 2 training, validation losses are tabulated with epoch numbers in the form of mean square error (MSE).

Table 2. Training and validation loss in terms of MSE for NN, CNN, INCNN.

Epochs	NN		CNN		INCNN	
	Train loss	Validation loss	Train loss	Validation loss	Train loss	Validation loss
100	0.00567	0.00627	0.00222	0.00243	0.00129	0.00236
200	0.00423	0.00598	0.00179	0.00199	0.00078	0.00085
300	0.00376	0.00512	0.00132	0.00143	0.00060	0.00068
400	0.00291	0.00382	0.00124	0.00139	0.00038	0.00042

The target coordinates are divided by 48 to scale them in [–1, 1] and for that to find RMSE, RMSE = $\sqrt{}$MSE * 48 is used here. Results have shown that INCNN has minimum loss and its validation RMSE value is 0.987.

Comparisons of proposed method with existing state-of-the art methods in terms of RMSE are shown in Table 3.

Table 3. Performance analysis in terms of RMSE

	RMSE
Shi [1] CNN model	1.874
LeNet [7]	1.349
NaimishNet [10]	1.03
Proposed method	0.987

The annotated faces with 15 keypoints of 3 individuals are shown in Fig. 4.

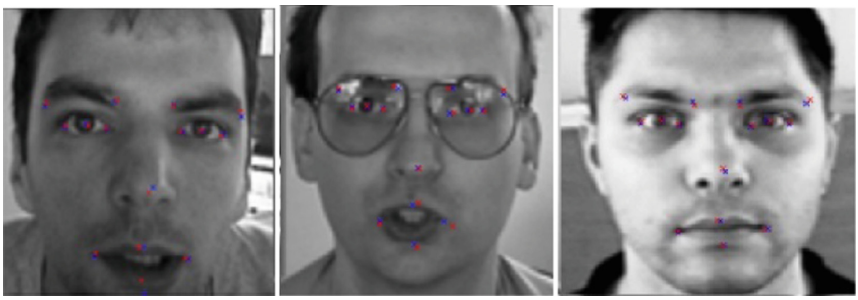

Fig. 4. Marked in blue for original and in red for predicted 15 facial keypoints. (Color figure online)

5 Conclusion

In the proposed work, one hidden layer neural network, convolutional neural network (CNN) and Inception model adopted convolutional neural network (INCNN) are designed for facial keypoints detection. It is observed from the performed experiments that applying CNN gives a more accurate result than simple NN. But after introducing the sparse architecture of INCNN, a far better accuracy with a very significant improvement in performance is achieved in facial keypoints prediction.

As a scope of future work, tuning the network by adjusting the hyper-parameters may improve the network performance. Again, applying more advanced image augmentation techniques to generalize the network and GPU implementation of the network for improved time complexity may be achieved.

References

1. Shi, S.: Facial Keypoints Detection, arXiv preprint arXiv:1710.05279, 15 October 2017
2. Kaggle dataset. https://www.kaggle.com/c/facial-keypoints-detection
3. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
4. Sun, Y., Wang, X., Tang, X.: Deep convolutional network cascade for facial point detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3476–3483 (2013)
5. Wang, Y., Song, Y.: Facial Keypoints Detection. Stanford University (2014)
6. Zhang, S., Meng, C.: Facial keypoints detection using neural network (2016)
7. Longpre, S., Sohmshetty, A.: Facial Keypoint Detection. Stanford University (2016)
8. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
9. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
10. Agarwal, N., Krohn-Grimberghe, A., Vyas, R.: Facial Key points Detection using Deep convolutional Neural Network – NaimishNet (2017)
11. Vukadinovic, D., Pantic, M.: Fully automatic facial feature point detection using gabor feature based boosted classifiers. In: 2005 IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1692–1698. IEEE (2005)
12. Valstar, M., Martinez, B., Binefa, X., Pantic, M.: Facial point detection using boosted regression and graph models. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2729–2736. IEEE (2010)
13. Amberg, B., Vetter, T.: Optimal landmark detection using shape models and branch and bound. In: 2011 IEEE International Conference on Computer Vision (ICCV). IEEE (2011)
14. Belhumeur, P.N., et al.: Localizing parts of faces using a consensus of exemplars. IEEE Trans. Pattern Anal. Mach. Intell. **35**(12), 2930–2940 (2013)
15. Parkhi, O.M., Vedaldi, A., Zisserman, A.: Deep face recognition. In: BMVC, vol. 1, no. 3, p. 6 (2015)
16. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)

17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
18. Peter, S.: Detecting facial features using Deep Learning. https://towardsdatascience.com/detecting-facial-features-using-deep-learning-2e23c8660a7a
19. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53
20. Aggarwal, C.C.: Chapter 1 An Introduction to Neural Networks. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94463-0_8
21. Sun, Y., Wang, X., Tang, X.: Deep convolutional network cascade for facial point detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
22. Daniel Nouri's blog. http://danielnouri.org/notes/2014/12/17/using-convolutional-neural-nets-to-detect-facial-keypoints-tutorial/
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for largescale image recognition. arXiv preprint arXiv:1409.1556 (2014)
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
25. Wang, W., Yang, J., Xiao, J., Li, S., Zhou, D.: Face recognition based on deep learning. In: Zu, Q., Hu, B., Gu, N., Seng, S. (eds.) HCC 2014. LNCS, vol. 8944, pp. 812–820. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15554-8_73
26. Cao, X., Wipf, D., Wen, F., Duan, G., Sun, J.: A practical transfer learning algorithm for face verification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3208–3215 (2013)
27. Sutskever, I., Martens, J., Dahl, G.E., Hinton, G.E.: On the importance of initialization and momentum in deep learning. In: Proceedings of the 30th International Conference on Machine Learning ICML 2013, Atlanta, GA, USA, volume 28 of JMLR Proceedings, pp. 1139–1147, 16–21 June 2013
28. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy, volume 9 of J. Mach. Learn. Res. pp. 249–256 (2010)