Improved Bilinear Model for Facial Expression Recognition

M. Amine Mahmoudi$^{1,3(*)}$, Aladine Chetouani$^{2}$, Fatma Boufera$^{1}$, and Hedi Tabia$^{3}$

$^{1}$ Mustapha Stambouli University of Mascara, Mascara, Algeria
$^{2}$ PRISME Laboratory, University of Orleans, Orleans, France
$^{3}$ Université Paris Saclay, IBISC, Univ Evry, Evry, France

Abstract. Facial Expression Recognition (FER) systems aims to classify human emotions through facial expression as one of seven basic emotions: happiness, sadness, fear, disgust, anger, surprise and neutral. FER is a very challenging problem due to the subtle differences that exist between its categories. Even though convolutional neural networks (CNN) achieved impressive results in several computer vision tasks, they still do not perform as well in FER. Many techniques, like bilinear pooling and improved bilinear pooling, have been proposed to improve the CNN performance on similar problems. The accuracy enhancement they brought in multiple visual tasks, shows that their is still room for improvement for CNNs on FER. In this paper, we propose to use bilinear and improved bilinear pooling with CNNs for FER. This framework has been evaluated on three well known datasets, namely ExpW, FER2013 and RAF-DB. It has shown that the use of bilinear and improved bilinear pooling with CNNs can enhance the overall accuracy to nearly 3% for FER and achieve state-of-the-art results.

Keywords: Facial expression recognition · Bilinear pooling · Improved bilinear pooling

1 Introduction

Facial expression recognition is a research area which consists of classifying the human emotions through the expressions on their faces as one of seven basic emotions: happiness, sadness, fear, disgust, anger, surprise and neutral. FER finds applications in different fields including security, intelligent human-computer interaction, robotics, and clinical medicine for autism, depression, pain and mental health problems.

With the resurgence of deep learning techniques, the computer vision community has witnessed an era of blossoming result thanks to the use of very large training databases. Big data is crucial to avoid the model being prone to overfitting. This was a very limiting factor for the use of deep learning for FER.
at the beginning, due to the limited size of facial expression datasets. This is not the case anymore with the emergence of very large in-the-wild datasets of facial expressions (e.g. FER2013 [7], ExpW [25], AffectNet [20], etc.). Yet, these datasets are more challenging because facial expressions are more affected by the in-the-wild conditions than other.

Bilinear CNN model is a combination of two CNNs A and B that takes as input the same image and output two feature maps. These feature maps are then multiplied at each location using tensor product. The result is pooled to obtain a global image descriptor of the image. The latter is passed to a classifier throughout make a prediction. Compared to single CNNs, bilinear CNN models have shown to achieve very good results on various visual tasks. For instance, semantic segmentation, visual questions answering and fine-grained recognition.

Fine-grained recognition is a research area that is interested in developing algorithms for automatically discriminating categories with only small subtle visual differences. Given that FER datasets contain very few categories that are nearly identical, we believe that any solution which is efficient for fine-grained recognition, like bilinear CNNs, may perform as well for FER.

In this paper, in addition of using bilinear CNN models, we propose to use an improved bilinear pooling with CNNs models for FER. In this framework, various ways of normalization are used to improve the accuracy, including the matrix square root, element-wise square root and L2 normalization.

The remainder of this paper is organized as follow: Sect. 2 reviews similar works that have been done on FER and bilinear CNN models. Section 3 gives more details about this approach. Section 4 presents our experiments, datasets and results; and Sect. 5 concludes the paper.

2 Related Work

Studies on FER using deep learning techniques used either self-built networks from scratch or fine-tuning on well-known pre-trained models. Many self-built architectures were proposed in the literature achieving various results on different datasets. For instance, multitask networks takes into consideration various factors like the head pose, illumination, facial landmarks, facial action units and subject identity. These factors are combined to conduct a simultaneous multitask learning which may lead to model that is close to the real world conditions. Some studies like [4,22] suggested that simultaneously conducted FER with other tasks, such as facial landmark localization and facial action units detection, can jointly improve FER performance. Other works have used network ensemble on different dataset for FER [9] achieving pretty high performance. Finally, the cascaded network, in which various modules for different tasks are combined sequentially to construct a deeper network, where the outputs of the former modules are utilized by the latter modules. In [5], Deep Belief Networks were trained to first detect faces and expression-related areas. Then, these parsed face components were classified by a stacked auto-encoder.

Self-built networks from scratch can achieve better result, but they needs to be trained on very large datasets. The dataset issue was a very limiting criteria
in the beginning, because of the lack of sufficiently large datasets. But this is not
the case anymore with the emergence of many large dataset containing thousands
of facial expression images (e.g. FER2013 [7], ExpW [25], AffectNet [20]...etc.).
An extensive survey has been proposed by Li and Deng [11] for more details.

Several methods have been proposed to improve the performance of CNNs. In [16] a bilinear pooling method for fine-grained recognition was proposed. Inspired from the second order pooling model introduced by [24], this model can capture higher interaction between image locations, which makes the model more discriminant than a simple model. This method have been used for FER by Zhou et al. [26] and noticed that they significantly outperformed their respective baselines. However, these models are high dimensional and could be impractical for a multitude of image analysis. In [6] two compact bilinear representations of these models have been proposed. They reached results as the full bilinear representation, yet with only a few thousand dimensions. This compact representations have also been used, by [21], in a multi-modal emotion recognition, combining facial expressions and voice sound. The latter was further generalized in the form of Taylor series kernel in [3]. The proposed method captures high order and non-linear feature interactions via compact explicit feature mapping. The approximated representation is fully differentiable, and the kernel composition can be learned together with a CNN in an end-to-end manner. Lin et al. have furthered their bilinear CNN model, by applying matrix normalization functions. Two matrix functions have been used, namely matrix logarithm and matrix square-root. All these methods are plugged at the end of the network, right between the convolution layers and the fully connected layers. They act as a basis expansion layers, increasing thereby the discrimination power of the fully connected layers, This discrimination power is back-propagated through the convolution layers. These methods have attracted increasing attentions, achieving better performance than classical first-order networks in a variety of tasks. Even-thought these methods increase the CNN performance, they are unable to learn by themselves and rely entirely on the CNN architecture. Furthermore, effectively introducing higher-order representation in earlier pooling layers, for improving non-linear capability of CNNs, is still an open problem.

More recently, Mahmoudi et al. [19] addressed this problem and proposed a novel pooling layer that not only reduces input information but also extracts linear and non-linear relations between features. It leverage kernel functions which allow to generalise linear pooling while capturing higher order information. Mahmoudi et al. [18] also introduced a novel FC layer based on kernel function. It applies a higher order function on its inputs instead of calculating their weighted sum. The proposed Kernelized Dense Layers (KDL) permits to improve the discrimination power of the full network and it is completely differentiable, allowing an end-to-end learning. The strength of these methods relies on the fact that they capture additional discriminant information compared to conventional pooling techniques.

To the best of our knowledge, improved bilinear CNN models have never been used for FER. We believe that these models can enhance the CNN performance
also for FER, given that FER is very similar to fine grained recognition. In the following sections, we will give more details about the bilinear and the improved bilinear CNN models. We will also explore the effect of using them on FER.

3 Approach

In this section we will describe the approach we used for our FER task. This technique, called bilinear CNN model, was inspired by Lin et al. [16]. It performed very well on fine-grained visual recognition tasks, and was later improved in [15]. We will describe bellow in more details bilinear CNN models and the improved version.

3.1 Bilinear CNN Models

Bilinear pooling models were first introduced by Tenenbaum and Freeman [24]. Also called second order pooling models, they were used to separate style and content. These models have been later used for fine grained recognition and semantic segmentation using both hand-tuned and learned features.

For image classification, we can generally formulate a bilinear model $B$ as a quadruple $B(f_A, f_B, P, C)$ (Fig. 1). Where $f_A$, and $f_B$, are feature functions, $P$ a pooling function and $C$ a classification function. A feature function takes an image $Img$ and a location $l \in Loc$ as inputs and produces a feature vectors, for each location in $Loc$, as follows:

$$f(l, Img) \mapsto \mathbb{R}^c$$  \hspace{1cm} (1)

We then combine these feature functions outputs vectors using the tensor product (Eq. 2) at each location. Here, $A$ and $B$ are feature vectors produced by the feature functions $f_A$, and $f_B$ respectively.
\[
A \otimes B = \begin{bmatrix}
a_1 \\
a_2 \\
\vdots \\
a_c \\
y_b
\end{bmatrix} \otimes \begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
y_b
\end{bmatrix} = \begin{bmatrix}
 a_1b_1 & a_1b_2 & \cdots & a_1b_c \\
a_2b_1 & a_2b_2 & \cdots & a_2b_c \\
\vdots & \vdots & \ddots & \vdots \\
a_nb_1 & a_nb_2 & \cdots & a_nb_c
\end{bmatrix}
\] (2)

Formally, the bilinear feature combination of \( f_A \) and \( f_B \) at a location \( l \in \text{Loc} \) is given by:

\[
\text{Bilinear}(l, \text{Img}, f_A, f_B) = f_A(l, \text{Img}) \otimes f_B(l, \text{Img})
\] (3)

The pooling function \( P \) combines the bilinear features throughout the different locations in the image (Eq. 4), which will produce a global image descriptor. One of the most used pooling functions are the sum and the max-pooling functions of all the bilinear features. Both functions ignore the location of the features and are hence orderless [16].

\[
P(\text{Loc}, \text{Img}, f_A, f_B) = \sum_{l \in \text{Loc}} f_A(l, \text{Img}) \otimes f_B(l, \text{Img})
\] (4)

A natural candidate for the feature function \( f \) is a CNN consisting of a succession of convolutional and pooling layers. According to [16], the use of CNNs is beneficial at many levels. It allows to use pre-trained CNNs in which we take only the convolutional layers including non-linearities as feature extractors. This can be beneficial specially when domain specific data is scarce. Another benefit of using only the convolutional layers is that the resulting CNN can process images of an arbitrary size in a single forward-propagation step. It produces outputs indexed by the location in the image and feature channel, in addition of reducing considerably the network's parameters number. Finally, the use of CNNs for a bilinear model allows this model to be trained in an end-to-end fashion. This technique has been used in a number of recognition tasks. For instance object detection, texture recognition and fine-grained classification and shown to give very good results.

Lin et al. [16] proposed bilinear CNN Models for fine-grained visual recognition (Fig. 2). The model consists of two CNNs, each trained to recognize special features. The resulting feature maps are sum-pooled to aggregate the bilinear features across the image. The resulting bilinear vector is then passed through signed square-root step, followed by L2 normalization, which improves performance in practice. Finally, the result will be fed to a classifier.
3.2 Improved Bilinear Pooling

Lin et al. [15] have also investigated various ways of normalization to improve the representation power of their bilinear model. In particular, a class of matrix functions were used to scale the spectrum (eigenvalues) of the co-variance matrix resulting of the bilinear pooling. One example of such normalization is the matrix-logarithm function defined for Symmetric Positive Definite (SPD) matrices. It maps the Riemannian manifold of SPD matrices to an Euclidean space that preserves the geodesic distance between elements in the underlying manifold (Fig. 3). An other normalization is the matrix square-root normalization which offers significant improvements and outperforms the matrix logarithm normalization when combined with element-wise square-root and L2 normalization. This improved the accuracy by 2–3% on a range of fine-grained recognition datasets leading to a new state-of-the-art.

The strength of bilinear models relies in the fact that they capture higher interaction between image locations, which makes the model more discriminant than a simple model. This allowed them to achieve impressive results in various image recognition tasks including FER. For instance, bilinear pooling has recently been used for FER in fine-grained manner [26]. Compact bilinear pooling has also been used in a multi-modal emotion recognition combining facial
expressions and voice sound [21]. But as far as we know, the improved bilinear pooling has never been used for FER. In the following section, we will explore the effect of using an improved bilinear CNN for FER. We will also implement a bilinear CNN to further appreciate the enhancement that can the improved bilinear CNN provide.

4 Experiments

In this section we will give more details about the experiments we performed in order to evaluate the approach described above. First, we give a brief description of the datasets we have used. After that, we describe architecture of the used models and training process. Finally, we discuss the obtained results.

4.1 Datasets

Our experiments have been conducted on three well-known facial expression datasets, namely the RAF-DB [13], ExpW [25] and FER2013 [7]. Facial expression datasets contain few classes that are nearly identical, which makes the recognition process more challenging.

- The RAF-DB [13] stands for the Real-world Affective Face DataBase. It is a real-world dataset that contains 29,672 highly diverse facial images downloaded from the Internet. With manually crowd-sourced annotation and reliable estimation, seven basic and eleven compound emotion labels are provided for the samples. This dataset is divided in training and validation subsets.
- The ExpW [25] stands for the EXPression in-the-Wild dataset. It contains 91,793 faces downloaded using Google image search. Each of the face images was manually annotated as one of the seven basic expression categories.
- The FER2013 database was first introduced during the ICML 2013 Challenges in Representation Learning [7]. This database contains 28709 training images, 3589 validation images and 3589 test images with seven expression labels: fear, happiness, anger, disgust, surprise, sadness and neutral.

In order to have the same dataset structure for all datasets, we divided the validation subset in RAF-DB [13] into validation and test subsets by a ratio of 0.5 each. We have also divided ExpW dataset with a ratio of 0.7 for training, 0.15 for validation and 0.15 for test.

4.2 Model Architecture and Training Process

For our experiment, we have used both a VGG-16 pre-trained on ImageNet database and a model built from scratch. For the VGG-16 we only took the convolution layers without the top fully connected ones. We added a batch normalization layer after each convolution layer (this enhances the model’s accuracy by nearly 1%). We added only one fully connected layer of size 512 and a final Softmax layer of seven output classes.
On the other hand, our model architecture, as shown in Fig. 4 is quite simple and can effectively run on cost-effective GPUs. It is composed of five convolutional blocks. Each block consists of a convolution, batch normalization and rectified linear unit activation layers. The use of batch normalization [27] before the activation brings more stability to parameter initialization and achieves higher learning rate. Each of the five convolutional blocks is followed by a dropout layer. In the following we refer to this network architecture as (Model-1).

![Fig. 4. Base model architecture (Model-1).](image)

The only pre-processing which we have employed on all experiments is cropping the face region and resizing the resulting images to 100 × 100 pixels. We have used Adam optimiser with a learning rate varying from 0.001 to 5e−5. This learning rate is decreased by a factor of 0.63 if the validation accuracy does not increase over ten epochs. To avoid over-fitting we have first augmented the data using a range degree for random rotations of 20, a shear intensity of 0.2, a range for random zoom of 0.2 and randomly flip inputs horizontally. We have also employed earl stopping if validation accuracy does not improve by a factor of 0.01 over 20 epochs.

4.3 Ablation Study

This section explores the impact of using bilinear pooling and improved version on the overall accuracy of the two base models (VGG-16 and Model-1). All the following experiments follow the same training process described above.

First we fine tuned the VGG-16 model on the three datasets and trained our model from scratch. Secondly, we took only the convolution part of the two trained models and add bilinear pooling (as shown in Fig. 2) with the following configurations: a) bilinear pooling on top of VGG-16, b) bilinear pooling on top of Model-1 and c) bilinear pooling on top of both VGG-16 and Model-1. We begin with fine tuning the bilinear pooling part only by freezing the underlying models. After that we train model in an end-to-end fashion. Finally, we repeated the same process of bilinear pooling with the improved version. That is to take the convolution part only of the fine-tuned VGG-16 and Model-1 and add the improved bilinear pooling (as shown in Fig. 3). We followed the same three configurations used for bilinear pooling.

Table 1 present the result of the two base models with comparison to these models with bilinear pooling and improved bilinear pooling. The VGG-16 model attains an accuracy rate of 65.23%, 67.61% and 85.23% on FER2013, ExpW and
RAF-DB respectively. Whereas Model-1 attains an accuracy of 70.13%, 75.91% and 87.05% respectively on FER2013, ExpW and RAF-DB.

### Table 1. Accuracy Rates of the proposed approach

| Models                  | Dataset     | FER2013 | ExpW | RAF-DB |
|-------------------------|-------------|---------|------|--------|
| VGG-16                  |             | 65.23%  | 67.61% | 85.23% |
| VGG-16-BP<sup>a</sup>   |             | 68.15%  | 68.82% | 85.77% |
| VGG-16-IBP<sup>b</sup>  |             | 68.71%  | 69.1%  | 86.34% |
| Model-1                 |             | 70.13%  | 75.91% | 87.05% |
| Model-1-BP<sup>a</sup>  |             | 71.63%  | 76.59% | 88.48% |
| Model-1-IBP<sup>b</sup> |             | 72.65%  | 77.81% | 89.02% |
| (VGG-16/Model-1)-BP<sup>a</sup> |     | 70.37%  | 73.57% | 86.47% |
| (VGG-16/Model-1)-IBP<sup>b</sup> |      | 71.22%  | 74.41% | 87.13% |

<sup>a</sup>BP: Bilinear Pooling.
<sup>b</sup>IBP: Improved Bilinear Pooling.

On the other hand one can notice that the use of bilinear pooling on top of a model increases considerably its accuracy. As reported in Table 1, the use of bilinear pooling on top of VGG-16 increases the accuracy for nearly 3% for FER2013 and more than 1% for both ExpW and RAF-DB. Similarly, the use of bilinear pooling on top of Model-1 increases the accuracy for about 1% on all datasets. However using bilinear pooling on top of both models gives an average accuracy rate between the underlying models accuracies. The resulting accuracy rates are 70.37%, 73.57% and 86.47% for FER2013, ExpW and RAF-DB respectively. This is due to the difference in accuracy between the two underlying models in the first place.

Finally, the use improved bilinear pooling increases further the accuracy rate for about 1% for all models with all datasets, compared to bilinear pooling. For instance, the accuracy rate of improved bilinear pooling on top of VGG-16 is 68.71%, 69.1% and 86.34% for FER2013, ExpW and RAF-DB respectively. Similarly, improved bilinear pooling on top of Model-1 gives 72.65%, 77.81% and 89.02% accuracy rates respectively for FER2013, ExpW and RAF-DB. The accuracy rate also increases when using improved bilinear pooling on top of both models. The later gives 71.22%, 74.41% and 87.13% on FER2013, ExpW and RAF-DB respectively.

These results demonstrate that the use of bilinear pooling and specially improved bilinear pooling, in the case of FER problem, are beneficial for the overall accuracy of the model. These techniques enhance the discriminative power of the model, compared to an ordinary fully connected layers.
4.4 Comparison with the State-of-the-Art

In this section, we compare the performance of the bilinear and improved bilinear CNN with respect to several state-of-the-art FER methods. The obtained results are reported in Table 2.

Table 2. Accuracy rate of the proposed approach and state of the art approach

| Models                  | Dataset   | FER2013 | ExpW | RAF-DB |
|-------------------------|-----------|---------|------|--------|
| VGG-16-BP\(^a\)        |           | 68.15%  | 68.82% | 85.77% |
| VGG-16-IBP\(^b\)       |           | 68.71%  | 69.1%  | 86.34% |
| Model-1-BP\(^a\)       |           | 71.63%  | 76.59% | 88.48% |
| Model-1-IBP\(^b\)      |           | 72.65%  | 77.81% | 89.02% |
| (VGG-16/Model-1)-BP\(^a\) |       | 70.37%  | 73.57% | 86.47% |
| (VGG-16/Model-1)-IBP\(^b\) |         | 71.22%  | 74.41% | 87.13% |
| Tang et al. [23]       |           | 71.16%  | –     | –      |
| Guo et al. [8]         |           | 71.33%  | –     | –      |
| Kim et al. [10]        |           | 73.73%  | –     | –      |
| Bishay et al. [2]      |           | –       | 73.1% | –      |
| Lian et al. [14]       |           | –       | 71.9% | –      |
| Acharya et al. [1]     |           | –       | –     | 87%    |
| S Li et al. [12]       |           | –       | –     | 74.2%  |
| Z. Liu et al. [17]     |           | –       | –     | 73.19% |

\(^a\)BP: Bilinear Pooling.
\(^b\)IBP: Improved Bilinear Pooling.

According to Table 2, the bilinear and improved bilinear CNN outperforms the state-of-the-art methods on the ExpW dataset. The best accuracy rate is 77.81% and has been reached using the improved bilinear pooling on top of Model-1. Bilinear pooling on top of Model-1 gives, for his turn, 76.59%. Moreover bilinear and improved bilinear on top of both VGG-16 and Model-1 gives respectively 73.57% and 74.41%. Whereas bilinear pooling and the improved version on top of VGG-16 give lower rates than state-of-the-art methods [2] (73.1%).

On RAF-DB dataset, the accuracy of our models is also superior to state of the art methods. The best accuracy rate is 89.02% and has been reached using the improved bilinear pooling on top of Model-1. Bilinear pooling on top of Model-1 gives, for his turn, 88.48%. Moreover improved bilinear on top of both VGG-16 and Model-1 gives 87.13%. Whereas bilinear pooling and the improved version on top of VGG-16, as well as bilinear pooling on top of both VGG-16 and Model-1 give lower rates than state-of-the-art methods [1] (87%).
For FER2013, even thought using the bilinear and improved bilinear pooling improves considerably the models accuracy, the obtained results are still under the state of the art results. The best accuracy rate for this dataset, namely 72.65%, was reached using improved bilinear pooling on top of Model-1. Which 1% less than the state-of-the-art method [10] (73.73%).

5 Conclusions and Perspectives

This study proposes a FER method based on the improved bilinear CNN model. In this framework, various ways of normalization are used to improve the accuracy, including the matrix square root, element-wise square root and L2 normalization. To validate our method, we have used three large, well known, facial expression databases which are FER2013, RAF-DB and ExpW. In order to evaluate the improvement of our method, we have first implemented a CNN from scratch and fine-tuned pre-trained VGG-16 on our facial expressions datasets. After that we have implemented a bilinear model on top of the above models individually and on top of both of them. Finally, we repeated the same procedure with the improved bilinear model. The experiments show that this framework improves the overall accuracy for about 3%.

Bilinear models have been shown to achieve very good accuracy results on different visual recognition domains, like fine grained recognition, semantic segmentation and face recognition. Nevertheless, the dimensions of bilinear features are very high, usually on the order of hundreds of thousands to a few million. The reason why they are not practical for many visual recognition fields. Moreover, matrix square root function and bilinear pooling function are very memory and CPU consuming, which decrease the performance of the model. Therefore, many improvements have been applied to CNN, for instance compact bilinear pooling [6], reaching the same discriminative power as the full bilinear representation but with a representations having only a few thousand dimensions. An other improvement is the kernel pooling for CNNs [3] which is a general pooling framework that captures higher order interactions of features in the form of kernels.

Our Future work will focus on using more compact alternatives of the methods used in this work. Moreover, our perspective is to use multiple input data types (text, image and sound) in parallel, thus forming a multilinear FER model.

References

1. Acharya, D., Huang, Z., Pani Paudel, D., Van Gool, L.: Covariance pooling for facial expression recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 367–374 (2018)
2. Bishay, M., Palasek, P., Priebe, S., Patras, I.: SchiNet: automatic estimation of symptoms of schizophrenia from facial behaviour analysis. IEEE Trans. Affect. Comput., 1 (2019)
3. Cui, Y., Zhou, F., Wang, J., Liu, X., Lin, Y., Belongie, S.: Kernel pooling for convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2930 (2017)

4. Devries, T., Biswaranjan, K., Taylor, G.W.: Multi-task learning of facial landmarks and expression. In: 2014 Canadian Conference on Computer and Robot Vision, pp. 98–103 (2014)

5. Fathallah, A., Abdi, L., Douik, A.: Facial expression recognition via deep learning. In: 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), pp. 745–750 (2017)

6. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 317–326 (2016)

7. Goodfellow, I.J., et al.: Challenges in representation learning: a report on three machine learning contests. In: Lee, M., Hirose, A., Hou, Z.-G., Kil, R.M. (eds.) ICONIP 2013. LNCS, vol. 8228, pp. 117–124. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42051-1_16

8. Guo, Y., Tao, D., Yu, J., Xiong, H., Li, Y., Tao, D.: Deep neural networks with relativity learning for facial expression recognition. In: 2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–6. IEEE (2016)

9. Hamester, D., Barros, P., Wermter, S.: Face expression recognition with a 2-channel convolutional neural network. In: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2015)

10. Kim, B.K., Dong, S.Y., Roh, J., Kim, G., Lee, S.Y.: Fusing aligned and non-aligned face information for automatic affect recognition in the wild: a deep learning approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 48–57 (2016)

11. Li, S., Deng, W.: Deep facial expression recognition: a survey. arXiv preprint arXiv:1804.08348 (2018)

12. Li, S., Deng, W.: Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition. IEEE Trans. Image Process. 28(1), 356–370 (2018)

13. Li, S., Deng, W., Du, J.: Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2584–2593. IEEE (2017)

14. Lian, Z., Li, Y., Tao, J.-H., Huang, J., Niu, M.-Y.: Expression analysis based on face regions in real-world conditions. Int. J. Autom. Comput. 17(1), 96–107 (2019). https://doi.org/10.1007/s11633-019-1176-9

15. Lin, T.Y., Maji, S.: Improved bilinear pooling with CNNs. arXiv preprint arXiv:1707.06772 (2017)

16. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear CNN models for fine-grained visual recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1449–1457 (2015)

17. Liu, Z., Li, S., Deng, W.: Boosting-pool: boosting part based one vs one feature for facial expression recognition in the wild. In: 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), pp. 967–972. IEEE (2017)

18. Mahmoudi, M.A., Chetouani, A., Boufera, F., Tabia, H.: Kernelized dense layers for facial expression recognition. In: 2020 IEEE International Conference on Image Processing (ICIP), pp. 2226–2230 (2020)

19. Mahmoudi, M.A., Chetouani, A., Boufera, F., Tabia, H.: Learnable pooling weights for facial expression recognition. Pattern Recogn. Lett. 138, 644–650 (2020)
20. Mollahosseini, A., Hasani, B., Mahoor, M.H.: AffectNet: a database for facial expression, valence, and arousal computing in the wild. IEEE Trans. Affect. Comput. 10, 18–31 (2017)

21. Nguyen, D., Nguyen, K., Sridharan, S., Dean, D., Fookes, C.: Deep spatio-temporal feature fusion with compact bilinear pooling for multimodal emotion recognition. Comput. Vis. Image Underst. 174, 33–42 (2018)

22. Pons, G., Masip, D.: Multi-task, multi-label and multi-domain learning with residual convolutional networks for emotion recognition. arXiv preprint arXiv:1802.06664 (2018)

23. Tang, Y.: Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239 (2013)

24. Tenenbaum, J.B., Freeman, W.T.: Separating style and content with bilinear models. Neural Comput. 12(6), 1247–1283 (2000)

25. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: From facial expression recognition to interpersonal relation prediction. Int. J. Comput. Vis. 126(5), 550–569 (2018). https://doi.org/10.1007/s11263-017-1055-1

26. Zhou, F., Kong, S., Fowlkes, C., Chen, T., Lei, B.: Fine-grained facial expression analysis using dimensional emotion model. arXiv preprint arXiv:1805.01024 (2018)

27. Zou, X., Wang, Z., Li, Q., Sheng, W.: Integration of residual network and convolutional neural network along with various activation functions and global pooling for time series classification. Neurocomputing, 367, 39–45 (2019)