Facial Emotions Classification Supported in an Ensemble Strategy

Rui Novais, Pedro J. S. Cardoso, and João M. F. Rodrigues(✉)

LARSyS & ISE, Universidade do Algarve, Faro, Portugal
{a49095,pcardoso,jrodrig}@ualg.pt

Abstract. Humans are prepared to comprehend each other's emotions from subtle body movements or facial expressions, and from those, they change the way they deliver messages when communicating between them. Machines, user interfaces, or robots need to empower this ability, in a way to change the interaction from the traditional "human-computer interaction" to a "human-machine cooperation", where the machine provides the "right" information and functionality, at the "right" time, and in the "right" way. This paper presents a framework for facial expression prediction supported in an ensemble of facial expression methods, being the main contribution the integration of outputs from different methods in a single prediction consistent with the expression presented by the system's user. Results show a classification accuracy above 73% in both FER2013 and RAF-DB datasets.

Keywords: Facial emotions · Ensembles · Computer vision · Machine learning

1 Introduction

Emotion and sentiment analysis methods are the automated processes of analyzing information to determine the emotion (e.g., happiness, sadness, fear, surprise, disgust, anger, and neutral) or sentiment (e.g., positive, negative, and neutral) expressed by the user. The sentiment influences the emotion, and the emotions influence the sentiment. Humans are prepared to comprehend each other's emotions from subtle body movements, facial expressions, the way they speek, or simply by the tone of voice. They use this capacity when communicating between them, changing the way they pass the message based on those responses/emotions/sentiments.

We are living in the so-called Information Society, Society 4.0. However, we are starting to notice that the cross-sectional sharing of knowledge is not enough. So, in Japan appeared a new designation, Society 5.0, which should be one that "through the high degree of merging between cyberspace and physical space, will be able to balance economic advancement with the resolution of social problems by providing goods and services that granularly address manifold latent needs regardless of locale, age, sex, or language." [1]. Simplifying, Society 5.0 is a super-smart, people-centric society.

To achieve this degree of development, one of the keys is to empower machines, user interfaces, or robots with the same communication capabilities that humans have between them. This changes the interaction between machines and humans from the traditional

M. Antona and C. Stephanidis (Eds.): HCII 2022, LNCS 13308, pp. 477–488, 2022.
https://doi.org/10.1007/978-3-031-05028-2_32

"human-computer interaction" (HCI) to a "human-machine cooperation" (HMC) [2], where the machine provides the "right" information and functionality, at the "right" time, and in the "right" way.

One of the solutions to achieve HMC relies on machine learning algorithms with a performance that depends greatly on the quality of the algorithm (and proper tuning), but also on the data's (high) quality. There are several ways to improve algorithms results, being the more usual way to train them repeatedly with all available data, with different settings, until the best possible result is achieved (fine-tuning the algorithm). Training might be extremely time-consuming, as well as it implies spending a lot of energy during the training phase, also increasing the algorithm's "carbon footprint".

Of course, there are ways in the literature to mitigate this problem, one of those is Active Learning [3]. The idea behind Active Learning is that the algorithms can achieve greater accuracy with fewer labelled training instances if they are allowed to pick the training data from which they learn, achieved by letting the learners to ask queries in the form of unlabeled instances to be labelled by an oracle (e.g., human annotator). This filtered use of data might even have a greater effect on performance and costs since many times labelled data is scarce and extremely expensive to obtain (unlabeled data may be abundant but labels are difficult, time-consuming, or expensive to acquire).

A different solution is applying assembly techniques, using for instance the results from algorithms previously thought and available in the community, e.g., open-source code. This use of hybridization and ensemble techniques allows empowering computation, functionality, robustness, and accuracy aspects of modelling [4], as well as allows to reduce the "carbon footprint" of the algorithm, once we use already trained algorithm(s), and now we are working with those results to develop the ensemble model. In short, the ensemble aggregates by (possibly) training with the results from the adopted methods. For instance, as it will be the case in this paper, the ensemble/aggregator method uses floating-point numbers returned by running established algorithms over an image, each number corresponding to a class of the emotion detection algorithm, instead of using an original color image.

This paper explores the last solution, a framework supported in the use of ensembles/aggregation of algorithms/methods to make the facial emotion classification from video clips or live streaming. The complete method receives information in the form of images (or frames) that will be passed to different types of facial emotion classifiers (available as open-source code), returning the same type or different types of results (corresponding to the emotions classes), which are then combined to return a (single) final result. The main contribution of the paper is the ensemble tool, which shows generically better results than using the methods individually.

In this Section, it was introduced the goals of the paper. Next sections present some related work (Sect. 2) and the proposed ensemble facial expression classification method (Sect. 3), followed by the developed tests and results in Sect. 4. Section 5 draws some conclusions and defines some potential future work.

2 Related Work

Expression recognition to interpersonal relation prediction needs input from different sources, e.g., sound, body, and facial expressions, as well as age or cultural environment. Zhang *et al.* [5] devise an effective multitask network that is capable of learning from rich auxiliary attributes such as gender, age, and head pose, beyond just facial expression data. Noroozi *et al.* [6] presented a survey on emotional body gesture recognition. While works based on facial expressions or speech abound, recognizing affect from body gestures remains a less explored topic. The authors in [6] present a new comprehensive survey hoping to boost research in the field. They first introduce emotional body gestures as a component of what is commonly known as "body language" and comment on general aspects as gender differences and cultural dependence. Then they define a complete framework for automatic emotional body gesture recognition.

Other solutions were also presented as, for example, the fusing of body posture with facial expressions for the recognition of affect in child-robot interaction [7]. The opposite also exists, i.e., the dissociation between facial and body expressions (in emotion recognition), as in a study done with impaired emotion recognition through body expressions and intact performance with facial expressions [8]. Further recent examples exist in the literature, such as, mood estimation based on facial expressions and postures [9] or, e.g., in the following works [10–14].

In the present case, we are focusing on a single aspect which is facial expression. Ekman and Friesen demonstrated that facial expressions of emotion are universal, i.e., the human way of expressing an emotion is supposed to be an evolutionary, biological fact, not depending on the specific culture [15]. Nevertheless, different methods for facial expression classification return different results when presented with the same input (face). The idea of facial expression recognition (FER) using an ensemble of classifiers is not new. For example, Zavaschi *et al.* [16] presented in 2011 a pool of base classifiers created using two feature sets: Gabor filters and Local Binary Patterns (LBP). Then a multi-objective genetic algorithm has used to search for the best ensemble using as objective functions the accuracy and the size of the ensemble. Later (in 2019), Renda *et al.* [17] compared several ensemble deep learning strategies applied to facial expression recognition (for static images only). Ali *et al.* [18] presented an ensemble approach for multicultural facial expressions analysis. Intending to get high expression recognition accuracy, the study presents several computational algorithms to handle those variations. They use facial images from participants in the multicultural dataset that originate from four ethnic regions, including Japan, Taiwan, "Caucasians", and Moroc.

Wang *et al.* [19] presented OAENet (oriented attention ensemble for accurate facial expression recognition). The authors used an oriented attention pseudo-siamese network that takes advantage of global and local facial information. Their network consists of two branches, a maintenance branch that consisted of several convolutional blocks to take advantage of high-level semantic features, and an attention branch that possesses a UNet like architecture to obtain local highlight information. The two branches are fused to output the classification results. As such, a direction-dependent attention mechanism is established to remedy the limitation of insufficient utilization of local information. With the help of the attention mechanism, their network not only grabs a global picture

but can also concentrate on important local areas. In [20] the authors present a facial emotion recognition system that addresses automatic face detection and facial expression recognition separately, the latter is performed by a set of only four deep convolutional neural networks concerning an ensembling approach, while a label smoothing technique is applied to deal with the miss-labelled training data.

The LHC is a Local (multi) Head Channel (self-attention) method [21], which is based on two main ideas. First, the authors hypothesize that in computer vision the best way to leverage the self-attention paradigm is a channel-wise application, instead of the more explored spatial attention, and that convolution will not be replaced by attention modules like recurrent networks were in NLP (natural language processing); second, a local approach has the potential to better overcome the limitations of convolution than global attention. With LHC, the authors managed to achieve a new state of the art over the FER2013 dataset [22], with significantly lower complexity and impact on the "host" architecture in terms of computational cost.

Py-Feat [23] is an open-source Python toolbox that provides support for detecting, preprocessing, analyzing, and visualizing facial expression data. Py-Feat allows experts to disseminate and benchmark computer vision models and also for end-users to quickly process, analyze, and visualize face expression data.

For two recent (2021) surveys on various deep learning algorithms for efficient facial expression classification and human face recognition techniques, please refer to the works of Banerjee *et al.* [24] and Revina & Emmanuel [25].

All the above-mentioned methods need a huge amount of data (images) from which they learn from. Differently, we intend that our method learns from the result of previously established models, simplifying the learning phase and reducing the time required to teach the classification model, as well as computing power that is needed for that (decreasing this way the local "carbon footprint" of the framework). The next section explores the proposed framework in more detail.

3 Facial Emotions Prediction Supported in Ensembles

As mentioned before, the framework to develop an emotion classifier should be supported in several sources/attributes, such as facial expression, body expression, speech, text, environment etc. Figure 1 illustrates that principle: the combination of an "undetermined" number of primary classifiers, that is dynamically added/removed/updated to/from the framework, to return a final prediction. The main idea behind the framework presented in Fig. 1 is that the primary methods are off-the-shelf methods, i.e., methods that have their code publicly available and can be easily added into the ensemble/aggregation model, by providing final and raw classification results that will be processed by the ensembled/aggregator for the final classification prediction. We stress that the framework does not intend to improve any of the primary models, but only to work with the results they return. In this context, the emotions classifications models used in this paper, had their code extracted from some repository and no changes of any kind were done in the code, meaning that the individual results presented by the emotion classifier, when applied to the datasets, are the collected and presented results, despite many times those are not coincident with the ones in the original publication.

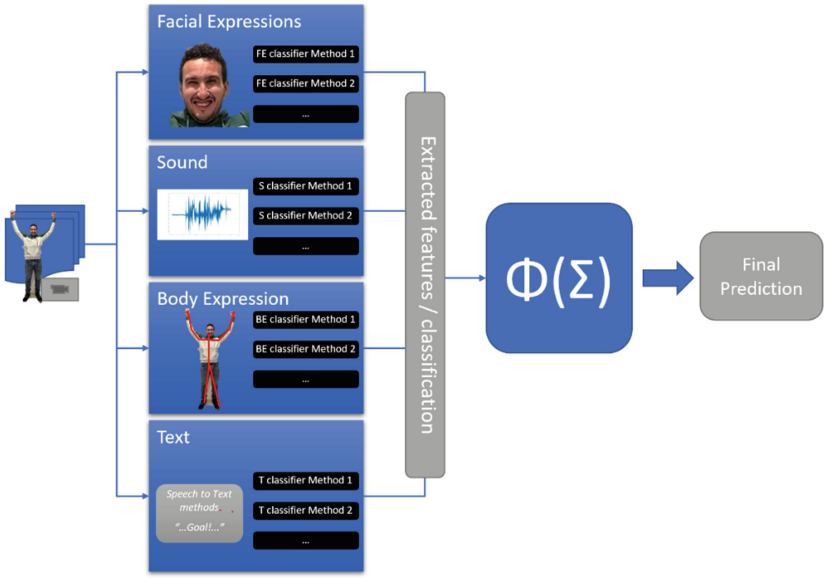

Fig. 1. Emotion classification framework scheme.

3.1 Framework

To facilitate the description, this paper only addresses the use of static images (i.e., it does not consider sequences of images, sounds, text, and body expressions) and considers that primary methods return 7 values corresponding to the different emotions, as we will see next. However, the framework is easily adaptable to different outputs from the primary classifiers, e.g., the number and type of returned features.

Within the expressed restriction, the pipeline of the framework consists in presenting the same image, to (i) n primary emotions classifiers (in the present case $n = 3$). Then, from the input image, each primary classifier returns a value between 0.0 (less likely to be) and 1.0 (more likely to be) for each of the seven classes/emotions (happiness, sadness, fear, surprise, disgust, anger, and neutral). (ii) The returned values are then injected to the ensemble/aggregation model, to produce a single final classification. This means that for each presented image there will be $n \times 7$ inputs to the aggregation model and an expression/emotion as output.

For the initial part (i) of the framework pipeline, the following primary emotion classifiers were used: (i) LHC [21] with its code available at [26]; (ii) Py-Feat [23] with its code available at [27]; and (iii) FERjs which is a free implementation done by Justin Shenk and has its code available at [28]. The reason to choose these three methods to build the baseline was: (a) they present state of the art results, (b) are recent methods, from 2021, (c) have publicly available code (implementation), and (d) represent different architectures. Again, it is important to stress that there is a huge number of different methods that could be used, as mentioned in [24, 25].

For the second part (ii) of the framework pipeline, the models used were: (a) Voting; (b) Random Forest [34]; (c) AdaBoost [35]; and (d) a Multi-layer Perceptron/Neural Network (MLP/NN) [36]. Models (b–d) were tested without ranking the values returned by the initial classifiers, as we will see later. In more detail, the (a) Voting method used the classes predicted by the primary classifiers to make a prediction if there is a majority of opinion between the guesses, that is, if at least two of the predictors guess the same emotion. If all return different emotions, then the Voting method returns no prediction. The Voting method can be considered as a naïve method but serves as a baseline for building more advanced aggregation strategies. The (b) Random Forest [34] is *per se* an aggregator of predictors. It starts by the draw of k bootstrap samples from the original data and then, for each of the bootstrap samples, grow an unpruned classification tree, with the following modification: at each node, rather than choosing the best split among all predictors, randomly sample m of the predictors and choose the best split from among those. The (c) AdaBoost method [35], as the name suggests, uses boosting which involves combining the predictions from many weak learners, being a weak learner a (very) simple model, although it has some skill on the dataset. The AdaBoost algorithm uses short (one-level) decision trees as weak learners that are added sequentially to the ensemble. Each subsequent model attempts to correct the predictions made by the model before it in the sequence. This is achieved by weighing the training dataset to put more focus on training examples on which prior models made prediction errors. Finally, the (d) Multi-layer Perceptron [36] is a feedforward artificial neural network model that maps sets of input data onto a set of outputs. An MLP consists of multiple layers and each layer is fully connected to the following one. There can be one or more non-linear hidden layers between the input and the output layer.

The next section details the tests and achieved results.

4 Tests and Results

For the tests, it was used, as mentioned, two different datasets, namely: (i) FER2013 [22, 32], where the data consists of 48 × 48-pixel grayscale images of faces. The faces have been automatically registered so that they are centered and occupy about the same amount of space in each image. Each face is annotated to a facial expression, of the seven previously mentioned categories. The training set consists of 28,709 examples and the public testing set consists of 3,589 examples; The second dataset used is (ii) RAF-DB [33], which is a facial expression database with 29,672 facial images, downloaded from the Internet. Images in this database have great variability in subjects' age, gender and ethnicity, head poses, lighting conditions, occlusions (e.g., glasses, facial hair, or self-occlusion), post-processing operations (e.g., various filters and special effects) etc. The images were classified into two different subsets: single-label subset, including the 7 classes of basic emotions (same as FER2013), and two-tab subset, including 12 classes of compound emotions. It also includes 5 accurate landmark locations, 37 automatic landmark locations, bounding boxes, race, age range, and gender attributes annotations per image. As usual, to be able to objectively measure the performance for the followers' entries, the database has been split into a train set and a test set, where the size of the training set is five times larger than the one of the testing set, and expressions in both sets have a near-identical distribution.

It is important to stress that in the case of RAF-DB, before applying the emotion classifier, a face detector was applied. As the goal of the paper is not to select the best detector to apply in this situation, it was applied one of the most well-known face detectors - Haar-Cascate face detector [34].

Regarding both datasets, Table 1 shows the accuracy for each primary emotion classifier and the Voting model, serving as a reference for the latter tested and more advanced aggregation models.

Table 1. Accuracy for the individual emotion classifiers and voting model.

Dataset	LHC	Py-Feat	FERjs	Voting
FER2013	70,59%	**77,98%**	65,67%	**73,37%**
RAF-DB (2)	60,68%	62,04%	62,19%	62,60%

Besides the Voting model, the other aggregators' methods were tuned using a grid search stratified 5-fold cross-validation, i.e., for each set of classifier parameters, the training data was divided into 5 folds with the same classes ratio as the training set and then the algorithms were trained and tested 5 times, where each time a new set (a fold) is used as testing set while remaining sets are used for training. The scoring (the accuracy in this case) for each set of parameters is computed as the mean score over the 5 train-test runs. Next, the full training dataset and best scored set of parameters are used to obtain the final model for each of the methods (i.e., Random Forest, AdaBoost, and MLP/NN). Used the Scikit-learn machine learning library for the Python programming language [35] (version 1.0.1), besides the default values, the parameters used to tune the methods are summarized in Table 3 (Appendix). The remaining section's results were attained in a personal computer running Kubuntu 21.10 over an Intel(R) Core(TM) i7-4770 CPU @ 3.40 GHz with 16 GiB of RAM.

Considering the methods and datasets above introduced, different combinations of those were used, obtaining different aggregation models. So, Table 2 shows the results of the different models applied to the different datasets where the results per row were obtained in the following manners. The FER2013 row shows the results considering that the outputs of the primary methods (LHC, Py-Feat, and FERjs), namely the methods' estimated confidence of being each of the emotions, are injected into the aggregators' methods (Random Forest, AdaBoost, and MLP/NN). In this case, 21 features are injected, since 3 primary methods returning 7 expressions were used, either directly with no transformation or ranked within each method (resulting in the "Without ranking" and "With ranking" table's columns). Furthermore, in the aggregators training phase, the injected values were the results of applying the primary methods to the FER2013 training dataset, and the results shown in the table are the values obtained by applying the final aggregator model (the model with parameters obtained from the grid-search cross-validation phase and trained over the full FER2013 training dataset) to the FER 2013 testing dataset. Row RAF-DB (1) shows the results of applying the above models (the model trained for table's RAF2013 row) directly to the full RAF-DB dataset, which in this case can be considered as a testing dataset since the model never saw that data. Finally,

row RAF-DB (2) was built similarly to row RAF2013, with the difference being that the training and testing datasets were RAF-DB training and testing datasets, respectively. The parameters obtained from the grid-search cross-validation are summarized in Table 4 (Appendix).

Table 2. Results of the different models applied to the different datasets.

Dataset	Without ranking			With ranking		
	Random Forest	AdaBoost	MLP/ NN	Random Forest	AdaBoost	MLP/NN
FER2013	71,08%	71,41%	70,95%	70,68%	70,79%	70,84%
RAF-DB (1)	**60,24%**	60,17%	60,01%	59,47%	59,53%	59,75%
RAF-DB (2)	**76,17%**	64,94%	74,14%	38,98%	38,98%	67,07%

Some conclusions can be drawn from Table 1 and Table 2. The first conclusion is that, although in some cases the values are close, the ranking of the values is not justified as it always returned worst accuracy than the corresponding method without ranking, i.e., it seems to be a better solution to inject the values from primary methods directly into the aggregation methods. Considering the results over the FER2013 test dataset, the best aggregation method was the Voting model with an accuracy of 73,37%, which is worse than the accuracy of the Py-Feat model (77,98%). As a curiosity, which is not presented in the tables, is the fact that the accuracy of the models over the FER2013 training dataset was 99.18%, 81.84%, and 74.96%, respectively. This seems to indicate overfitting of the first method since it drops from 99.18% accuracy over the training dataset to 70,59% accuracy in the testing dataset. In the reverse, Py-Feat suffered a very small drop from 81.84% to 77,98% of accuracy. This is relevant since having a 99.18% accuracy, the aggregation methods might have had some somehow misleading predictions from method LHC – this was an expectable risk and provides us with further studies to mitigate this threat. Applying the aggregated model trained for FER2013 to RAF-DB is interesting by the fact that it produces results very similar to the primary methods without the need to train them with that dataset. In more detail, LHC, Py-Feat and FERjs trained with RAF-DB training dataset produced an accuracy of 60,68%, 62,04%, and 62,19%, respectively, which is very similar to the aggregation methods trained with FER2013 accuracies (60,24%, 60,17%, and 60,01%, respectively), but without the need to train a new model. If the aggregation models were trained using the prediction from LHC, Py-Feat and FERjs for the RAF-DB training set, then their prediction improve the base methods in all (the without ranking) cases, i.e., the best accuracy was 62.19% for method FERjs, and the aggregation methods attained an accuracy of 76,17%, 64,94%, and 74,14% (for Random Forest, Ada-Boost, and MLP/NN, respectively).

Between the aggregation methods, AdaBoost was the one performing worst. Random forest and MLP/NN had similar results, being the Random Forest slightly better in the tested cases.

5 Conclusions

This paper presents a simplified version of a facial expression/emotions predictor framework supported in ensembles. The pipeline of the frameworks consists in presenting an image, to several (primary, pre-trained) emotions classifiers. Then, each classifier returns for each image and for each of the seven considered classes/emotions (happiness, sadness, fear, surprise, disgust, anger, and neutral) its confidence values. Those results are then fed to an ensemble/aggregator model returning a single predicted class.

The best results for the aggregators methods in the case of FER2013 dataset were achieved with the Voting model (supported on the majority of the models' predictions), being above two of the primary emotion classifiers but below one of them. This result is achieved probably because the emotion classifiers were taught with FER2013 but have different accuracy behaviors (one of the classifiers is probably overfitted since the accuracy dropped from almost 100% on the training set to nearly 70% on the test set). In the case of RAF-DB, the best result was achieved with the model Random Forest aggregator, and the result is above all the results achieved individually by the primary emotion classifiers.

In future work we intend to explore different datasets, like the ones mentioned in [36, 37] and datasets that have motion (video or streaming). We will also try to improve the final results by increasing the number of emotion classifiers, and studying the influence of their characteristics, like the fact that they are over or underfitted.

Acknowledgements. This work was supported by the Portuguese Foundation for Science and Technology (FCT), project LARSyS - FCT Project UIDB/50009/2020.

Appendix

This appendix presents the parameters (Tables 3 and 4) used for Random Forest, AdaBoost and MLP/Neural Network for the results presented in Sect. 4.

Table 3. Grid search parameters (although the majority of the naming of the parameters is self-explicative, we suggest that the readers refer to the library's documentation [35] for a more detailed explanation).

Random Forest	
Number of trees in the forest (n_estimators)	{25, 50, 100, 500}
Function to measure the quality of a split. (criterion)	{gini, entropy}
Maximum depth of the tree(max_depth)	{None, 2, 5, 10, 20}
Minimum number of samples required to split an internal node (min_samples_split)	{2, 5, 10}
Minimum number of samples required to be at a leaf node (min_samples_leaf)	{1, 2, 5, 10}
Number of features to consider when looking for the best split (max_features)	{1, 2, sqrt, log2}
Number of samples to draw from X to train each base estimator (max_samples)	{None, 0.1}
AdaBoost	
The maximum number of estimators at which boosting is terminated (n_estimators)	{25, 50, 100, 500}
Boosting algorithm (algorithm)	{SAMME, SAMME.R}
MLP/Neural Network	
The i-th element represents the number of neurons in the i-th hidden layer (hidden_layer_sizes)	{(10,), (100,), (10, 10), (100, 100), (10, 10, 10), (100, 100, 100)}
Learning rate schedule for weight updates (activation)	{identity, logistic, tanh, relu}
L2 penalty (regularization term) parameter (alpha)	$\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$
Learning rate schedule for weight updates (learning_rate)	{constant, invscaling, adaptive}

Table 4. Sets of parameters used to obtain the results for the different models (tuned using grid search stratified cross-validation).

		With ranking		Without ranking	
		FER2013	*RAF-DB*	*FER2013*	*RAF-DB*
Random Forest	n_estimators	100	500	50	100
	criterion	gini	entropy	gini	gini
	max_depth	10	None	None	20
	min_samples_split	10	5	2	10
	min_samples_leaf	10	1	10	1
	max_features	Sqrt	2	2	1
	max_samples	0.1	None	None	None
AdaBoost	n_estimators	500	50	500	500
	algorithm	SAMME.R	SAMME	SAMME.R	SAMME.R

(*continued*)

Table 4. (*continued*)

		With ranking		Without ranking	
		FER2013	*RAF-DB*	*FER2013*	*RAF-DB*
MLP/NN	hidden_layer_sizes	(100, 100, 100)	(100,)	(10, 10)	(100,)
	activation	identity	relu	tanh	tanh
	alpha	0.01	0.1	0.1	1
	learning_rate	invscaling	constant	constant	constant

References

1. Deguchi, A., et al.: What is Society 5.0. Chapter 1 in Society 5.0 – A People-centric Super-smart Society. Hitachi-UTokyo Laboratory (eds.), pp. 1–23. Springer (2020). https://doi.org/10.1007/978-981-15-2989-4
2. Rothfuß, S., Wörner, M., Inga, J., Hohmann, S.: A study on human-machine cooperation on decision level. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, pp. 2291–2298. IEEE (2020)
3. Kumar, P., Gupta, A.: Active learning query strategies for classification, regression, and clustering: a survey. J. Comput. Sci. Technol. **35**(4), 913–945 (2020). https://doi.org/10.1007/s11390-020-9487-4
4. Ardabili, S., Mosavi, A., Várkonyi-Kóczy, A.R.: Advances in machine learning modeling reviewing hybrid and ensemble methods. In: Várkonyi-Kóczy, A.R. (ed.) INTER-ACADEMIA 2019. LNNS, vol. 101, pp. 215–227. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-36841-8_21
5. Zhang, F., Zhang, T., Mao, Q., Xu, C.: Joint pose and expression modeling for facial expression recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3359–3368 (2018)
6. Noroozi, F., Kaminska, D., Corneanu, C., Sapinski, T., Escalera, S., Anbarjafari, G.: Survey on emotional body gesture recognition. IEEE Trans. Affect. Comput. **12**(2), 505–523 (2018)
7. Filntisis, P.P., Efthymiou, N., Koutras, P., Potamianos, G., Maragos, P.: Fusing body posture with facial expressions for joint recognition of affect in child–robot interaction. IEEE Robot. Autom. Lett. **4**(4), 4011–4018 (2019)
8. Leiva, S., Margulis, L., Micciulli, A., Ferreres, A.: Dissociation between facial and bodily expressions in emotion recognition: a case study. Clin. Neuropsychol. **33**(1), 166–182 (2019)
9. Canedo, D., Neves, A.J.: Mood estimation based on facial expressions and postures. In: Proceedings of the RECPAD 2020, pp. 49–50 (2020)
10. Bänziger, T., Mortillaro, M., Scherer, K.R.: Introducing the geneva multimodal expression corpus for experimental research on emotion perception. Emotion **12**(5), 1161 (2012)
11. Kleinsmith, A., BianchiBerthouze, N.: Affective body expression perception and recognition: a survey. IEEE Trans. Affect. Comput. **4**(1), 15–33 (2012)
12. Senecal, S., Cuel, L., Aristidou, A., Magnenat-Thalmann, N.: Continuous body emotion recognition system during theater performances. Comput. Animat. Virtual Worlds **27**(3–4), 311–320 (2016)
13. Ahmed, F., Bari, A.H., Gavrilova, M.L.: Emotion recognition from body movement. IEEE Access **8**, 11761–11781 (2019)

14. Liang, G., Wang, S., Wang, C.: Pose-aware adversarial domain adaptation for personalized facial expression recognition. arXiv preprint arXiv:2007.05932 (2020)
15. Ekman, P., Friesen, W.V.: Constants across cultures in the face and emotion. J. Pers. Soc. Psychol. **17**(2), 124 (1971). https://doi.org/10.1037/h0030377
16. Zavaschi, T.H., Koerich, A.L., Oliveira, L.E.S.: Facial expression recognition using ensemble of classifiers. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1489–1492 (2011). https://doi.org/10.1109/ICASSP.2011.5946775
17. Renda, A., Barsacchi, M., Bechini, A., Marcelloni, F.: Comparing ensemble strategies for deep learning: an application to facial expression recognition. Expert Syst. Appl. **136**, 1–11 (2019)
18. Ali, G., et al.: Artificial neural network based ensemble approach for multicultural facial expressions analysis. IEEE Access **8**, 134950–134963 (2020)
19. Wang, Z., Zeng, F., Liu, S., Zeng, B.: OAENet: oriented attention ensemble for accurate facial expression recognition. Pattern Recognit. **112**, 107694 (2021)
20. Benamara, N.K., et al.: Real-time facial expression recognition using smoothed deep neural network ensemble. Integr. Comput.-Aid. Eng. (Preprint) **28**, 1–15 (2021)
21. Pecoraro, R., Basile, V., Bono, V., Gallo, S.: Local multi-head channel self-attention for facial expression recognition. arXiv preprint arXiv:2111.07224 (2021)
22. Goodfellow, I., 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
23. Cheong, J.H., Xie, T., Byrne, S., Chang, L.J.: Py-Feat: Python facial expression analysis toolbox. arXiv preprint arXiv:2104.03509 (2021)
24. Banerjee, R., De, S., Dey, S.: A survey on various Deep Learning algorithms for an efficient facial expression recognition system. Int. J. Image Graph., 2240005 (2021)
25. Revina, I.M., Emmanuel, W.S.: A survey on human face expression recognition techniques. J. King Saud Univ.-Comput. Inf. Sci. **33**(6), 619–628 (2021)
26. LHC-NET: Local multi-head channel self-attention (code) (2021). https://github.com/bodhis4ttva/lhc_net. Accessed 28 Dec 2021
27. Py-FEAT: Python facial expression analysis toolbox (code) (2021). https://pythonrepo.com/repo/cosanlab-py-feat-python-deep-learning. Accessed 28 Dec 2021
28. Shenk, J.: Facial expression recognition (code) (2021). https://github.com/justinshenk/fer. Accessed 28 Dec 2021
29. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
30. Hastie, T., Rosset, S., Zhu, J., Zou, H.: Multi-class Adaboost. Statistics and its. Interface **2**(3), 349–360 (2009)
31. Ayyadevara, V.K.: Pro Machine Learning Algorithms. Apress, Berkeley (2018)
32. FER2013: Learn facial expressions from an image (2021). https://www.kaggle.com/msambare/fer2013. Accessed 28 Dec 2021
33. RAF-DB: Real-world affective faces database (2021). http://www.whdeng.cn/raf/model1.html. Accessed 28 Dec 2021
34. OpenCV: OpenCV: Cascade classifier – face detection (2021). https://docs.opencv.org/4.5.5/db/d28/tutorial_cascade_classifier.html. Accessed 28 Dec 2021
35. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
36. Li, S., Deng, W.: Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition. IEEE Trans. Image Process. **28**(1), 356–370 (2019)
37. Cheong, J.H., Xie, T., Byrne, S., Chang, L.J.: Py-Feat: Python facial expression analysis toolbox. arXiv preprint arXiv:2104.03509 (2021)