Benchmarking deep networks for facial emotion recognition in the wild

Antonio Greco1 · Nicola Strisciuglio2 · Mario Vento1 · Vincenzo Vigilante1

Received: 27 September 2021 / Revised: 20 December 2021 / Accepted: 24 February 2022 / Published online: 26 March 2022
© The Author(s) 2022, corrected publication 2022

Abstract
Emotion recognition from face images is a challenging task that gained interest in recent years for its applications to business intelligence and social robotics. Researchers in computer vision and affective computing focused on optimizing the classification error on benchmark data sets, which do not extensively cover possible variations that face images may undergo in real environments. Following on investigations carried out in the field of object recognition, we evaluated the robustness of existing methods for emotion recognition when their input is subjected to corruptions caused by factors present in real-world scenarios. We constructed two data sets on top of the RAF-DB test set, named RAF-DB-C and RAF-DB-P, that contain images modified with 18 types of corruption and 10 of perturbation. We benchmarked existing networks (VGG, DenseNet, SENet and Xception) trained on the original images of RAF-DB and compared them with ARM, the current state-of-the-art method on the RAF-DB test set. We carried out an extensive study on the effects that modifications to the training data or network architecture have on the classification of corrupted and perturbed data. We observed a drop of recognition performance of ARM, with the classification error raising up to 200% of that achieved on the original RAF-DB test set. We demonstrate that the use of the AutoAugment data augmentation and an anti-aliasing filter within down-sampling layers provide existing networks with increased robustness to out-of-distribution variations, substantially reducing the error on corrupted inputs and outperforming ARM. We provide insights about the resilience of existing emotion recognition methods and an estimation of their performance in real scenarios. The processing time required by the modifications we investigated (35 ms in the worst case) supports their suitability for application in real-world scenarios. The RAF-DB-C and RAF-DB-P test sets, trained models and evaluation framework are available at https://github.com/MiviaLab/emotion-robustness.

Keywords Affective computing · Robustness · Emotion recognition · Expression recognition · Face analysis · Deep learning · Benchmarking

This research was partially supported by the Italian MIUR within PRIN 2017 grants, Projects Grant 20172BH297 002CUP D44I17000200005

Antonio Greco
agreco@unisa.it

Extended author information available on the last page of the article.
1 Introduction

Convolutional Neural Networks (CNNs) are very effective in dealing with computer vision tasks, with deeper and more complex architectures scoring very high accuracy on publicly available benchmarks. However, huge network architectures are sensitive to slight variations of the input data. Goodfellow et al. [18] showed one aspect of this problem in the form of adversarial attacks: images can be slightly modified with tailor-made noise, such that they remain visually identical for a human eye but strongly affect the response of neurons in a neural network.

The robustness problems of neural networks are also revealed when recognition methods are deployed in real scenarios [48]. Variations such as small changes in the framing of the shot, motion blur, focus or the amount of Gaussian noise do not typically affect a human observer but may jeopardize the performance of CNNs [22]. Vasiljevic et al. [40] demonstrated that even a moderate blur can severely affect the reliability of object recognition systems, when the architecture is trained on a data set of generally sharp images. Zheng et al. [48] explored the instability of the learned representations with respect to distortions such as JPEG compression, image scaling and cropping. Geirhos et al. [16] showed how image degradations reduce the performance of the trained models. Recently, the AutoAugment strategy for data augmentation was proposed, which consists of a set of policies optimized on the data set at hand [9]. It was demonstrated to improve the robustness of image classification models [45]. On a different line, architectural modifications to existing models were proposed to improve their robustness to corruptions and perturbations. An anti-aliasing filter was deployed before sub-sampling operations by Zhang [47], and a new push-pull layer was proposed by Strisciuglio et al. [39] to learn feature extractors in CNNs that are intrinsically more robust to noise and corruptions.

Applications like cognitive robotics and intelligent surveillance require to process face images acquired in unconstrained conditions [43], where many of the mentioned corruptions may occur and affect the performance of the recognition methods. For example, the CMOS sensors deployed in embedded cameras produce noise that can be modeled as the combination of Gaussian noise and shot noise, the former being mainly due to the sensor temperature and the latter being more prominent with high exposure. The limited dynamic range of these sensors produces images with low useful contrast when the scene includes bright and dark parts. In such conditions, the acquired faces look very dark, e.g. when the shot is taken in back-light. Furthermore, memory and bandwidth constraints may require the use of image compression algorithms (e.g. JPEG) that may in turn include artifacts in the images. The limitations of the acquisition devices lead to blurred images due to poor focus and to motion blur artefacts caused by the movement of the subject or the camera itself. The image may present occlusions caused by dirt or water on the camera lens, such as in video surveillance images.

Further challenges come from the variations that occur in image sequences. In real-world applications, recognition methods analyze continuous streams of video data. Each frame usually exhibits coherent content but has slight differences with respect to the previous frame. This is peculiar for face analysis: the appearance of a face can have slight variations (e.g. small pose or expression changes) that, independently of other types of corruptions, represent a challenge and require the learning of robust features to perform consistent analyses. When dealing with videos, one would expect the CNN output to be stable from one frame to the next, while it has been observed that variations such as image shifts severely affect classification performance of current deep network models [47].
The performance of existing emotion recognition methods is often computed on benchmark data sets, with limited or absent consideration of corruptions and perturbations that can occur in the real-world. This type of analysis, although important for the design of improved methods and the progress of the field, does not allow to assess the performance of the developed methods when deployed in practice. We show that the classification error of SOTA methods for emotion recognition easily increases by more than 70% when input data is subjected to unexpected changes. Robustness-by-design is thus required for engineering AI systems to reliably work in real environments.

We study the robustness of SOTA methods for emotion recognition from faces with respect to common image corruptions and perturbations. Following on the work of Hendrycks and Dietterich [22], we define a set of image corruptions typical for the application at hand, and a set of perturbations on subsequent frames of a video sequence. We evaluate the effect of architectural and data-related changes on the robustness of the methods, namely the use of an anti-aliasing filters before down-sampling operations [48] and the AutoAugment policies [9] for augmenting training data. We constructed a new benchmark data set by modifying the RAF-DB data set with custom corruptions and perturbations of different intensity. We generated a new validation set for each corruption (18 corruption times 5 intensity levels) and for each perturbation type (10 sets).

We benchmark the performance of networks that ranked among the best performing on the RAF-DB data set, namely SENet, Xception, DenseNet and ARM, when the input images are subjected to corruptions and perturbations. We use the VGG architecture as the baseline network for our evaluation, since it has been widely used for face analysis [38]. We also evaluated an handcrafted feature-based methods, namely LBP histograms with a Support Vector Machine classifier [36], which achieved the state-of-the-art performance before deep networks became the de-facto standard for many computer vision applications, including facial expression recognition [10]. We also included ARM method [37] in our benchmark, that is based on a complex deep network and currently holds the best emotion recognition accuracy on the RAF-DB test set.

The contributions of the paper are three-fold:

1. An evaluation of the robustness of SOTA networks for emotion recognition on a new data set that contains common corruptions and perturbations that happen in real scenarios. We showed that current benchmarks are limited by not properly evaluating generalization and robustness properties of state-of-the-art models (e.g. ARM), which are key factors for deploying them in real scenarios. To the best of our knowledge, this is the first such analysis done on facial emotion recognition.

2. A study of the impact that techniques for augmenting training data, i.e. AutoAugment, and to reduce aliasing introduced by down-sampling operations (e.g. pooling and strided convolution) in CNNs, i.e. an anti-aliased sampling layer, have on the robustness of existing methods.

3. The code, the data set, the trained network models and the AutoAugment policies for RAF-DB are publicly available.

The paper is structured as follows. In Section 2 we discuss existing methods for emotion recognition in the wild. In Section 3, we describe the experimental framework, data and evaluation metrics, the considered methods and the training hyper-parameters, and the modifications we deployed to improve the robustness of existing models. In Section 4 we report and discuss the results that we achieved, and in Section 5 we draw conclusions.
2 Related works

The growing application of facial emotion recognition in social robotics and business retail stimulated the design of better methods during the past years, many of them based on deep learning models. While part of the focus shifted towards the fusion of data modalities [8, 15, 42], the analysis of face images has been considering more and more complex scenarios [20], including the challenging cognitive robotics [19]. The publication of ‘in the wild’ data sets acquired in challenging conditions, like Aff-Wild [26], and methods that tackle the challenges of those new data sets witness an interest in improving existing approaches. To fulfill the needs of Deep Learning based methods, datasets need to be both representative of real conditions and increasingly large. With such constraints, the design and efficient gathering of such datasets becomes a challenge itself [27].

Goodfellow et al. [17] created the FER-2013 data set, which contains more than 28k grayscale images (of size 48 × 48 pixels) from Google Search and is still one of the most used datasets for categorical emotion recognition. Later, Mollahosseini et al. [32] made available AffectNet, which was collected with a similar approach, with almost 1 million images. AffectNet is automatically annotated in part, with about 60% annotator agreement only, making it unreliable for thorough evaluation of classification methods. In general, face emotion recognition data sets have noisy labels, due to the subjective nature of emotion perception. Subsequent works aimed at reliably labeling data sets with redundant information coming from multiple annotators. Barsoum et al. [4] developed the FERPlus data set including the same images from the FER-2013 data set with annotation improved by crowd-sourcing. Each image is tagged by ten different annotators. Li et al. [28] obtained the RAF-DB data set, that we used for our experiments, by adopting the same approach to annotate 30k facial images from the internet, with 40 annotations per image on average.

Many techniques were proposed throughout the years to improve facial expression recognition methods. The authors of the RAF-DB data set proposed Deep Locality-Preserving (DLP) learning, where the loss function explicitly addresses the intra-class variance to reduce the categorical accuracy imbalance due to the different number of samples available for the 7 emotion classes [28]. It encourages the activation of the last hidden layer for samples of the same class to have a common centroid in the feature space. With this approach, they reached an accuracy of 74.2%. Kim et al. [25] trained a ‘contrastive’ encoder in a double encoder-decoder setting. The learned features are used for generating two images, the original one and a version with neutral-expression. The encoder trained in this setting is used as input for a fully connected classifier; however, the method was not able to climb the leaderboard on the RAF-DB test set. Fan et al. [13] proposed a Multi-region Ensemble CNN (MRE-CNN). Three significant sub-regions were cropped from the face (the left eye, the nose and the mouth) and each was given as input to a double-input network, alongside with the full face image. Their final result was an accuracy of 76.7% using a VGG-16 backbone. Li et al. [29] designed a neural network trained using Global-Local Attention (gACNN) and a VGG-16 backbone. Facial landmarks were used to compute local attention from patches, then the information was integrated with global attention. The use of attention makes this method particularly robust to occlusions, obtaining an overall 85.1% accuracy on the RAF-DB test set. Acharya et al. [1] reported an accuracy equal to 87% using covariance pooling, based on the intuition that second order statistics better model the face changes that represent an emotion rather than max or average pooling. SPDNet layers were used to reduce the dimensionality of covariance matrices while preserving the spatial structure. The center loss function, a simplified version of DLP, was used for training. Ly et al. [31] used 3D
reconstruction as a method for face frontalization to reduce the variability of the pose of the faces in the images fed to the classifier. An Inception-ResNetV1 architecture pretrained on the VGG-Face2 data set was then fine-tuned on the RAF-DB data set, achieving 85.1% accuracy. Farzaneh and Qi [14] optimized a ResNet-18 for facial emotion recognition with a novel Deep Attentive Center Loss (DACL), designed to adaptively select a subset of discriminative features. This method achieved 87.8% accuracy on RAF-DB. Various attention mechanisms are also explored by [49], that achieves 89.1% accuracy with a LResNet50E-IR CNN. Better than the previous ones (89.7%) did [44] with a very complex network based on multi-head attention, called DAN, consisting of three parts: a feature clustering network to extract robust emotion features, a multi-head attention network to focus on various facial regions simultaneously and an attention fusion network that computes a global attention map. A pyramid with super-resolution (PSR) network allowed [41] to obtain 89.0% accuracy with a VGG-16, with higher resolution images. The method by [37] holds the best (90.4%) accuracy on the RAF-DB test set, by using an Amending Representation Module (ARM) that replaces the pooling layers of a ResNet-18 and a decomposition of the facial features which allows to simplify the representation learning.

Many efforts were made to collect in-the-wild data sets, with a high degree of variability and a reliable annotation. Progress were also made to craft methods that achieve high recognition accuracy. Despite the efforts, the data sets always consist of images from the web and do not reproduce noisy conditions, and none of the methods is supported by an analysis of robustness and stability towards realistic image corruptions and perturbations.

3 Experimental framework

We defined a benchmark framework for evaluation of classifier robustness, based on the approach proposed in [22]. We trained several methods on the images of the original training set of the RAF-DB data set and tested on corrupted and perturbed versions of the test set. The benchmark does not involve training on a corrupted or perturbed version of the training set.

We designed the corruptions and perturbations of the test images to simulate out-of-distribution samples that occur in real applications of emotion recognition. We call RAF-DB-C and RAF-DB-P the corrupted and perturbed test sets, respectively.

We trained enhanced methods for increased robustness. On one hand we studied the effects of architectural changes in the network design and on the other hand, we evaluated the modification of the training data by means of specific data augmentation policies.

In the rest of the section, we provide details about the experimental framework, namely the data, the methods and the evaluation protocol that we adopted.

3.1 Data set and evaluation metrics

The RAF-DB data set is one of the most popular benchmarks for emotion recognition [28]. It consists of 29,672 face images, of which 15,339 are annotated with the six basic emotions theorized by Ekman et al. [12], plus a neutral class that represents the absence of emotion. The images are divided in a training (12,271 instances) and a test set (3,068 instances); we used the detected, cropped and aligned faces, according to the indications of the authors.

The data set is widely used because of its reliable ground truth: each image is annotated by 40 different individuals and the multi-label annotation is available as a seven dimensional
vector, in which the number of votes for each class are provided. This allows the training procedure to take advantage of the intrinsic ambiguity of emotion evaluation.

In the training phase we discarded the samples with ambiguous annotation, according to the criteria described by Barsoum et al. [4], and computed the class probability distribution removing the outlier votes. In detail:

- the votes of the classes with less than 10% of the total votes is set to zero;
- the samples as no face or unknown are discarded;
- the samples with more than two classes with equal votes are discarded;
- the samples for which the winner class has less than 50% of the votes are discarded;
- the label vector is normalized to length 1.

3.1.1 RAF-DB-C

We created the RAF-DB-C data set by applying to the images contained in the RAF-DB test set 18 corruptions from a set $C$ that we will describe in this paragraph. We extended the corruptions proposed by Hendrycks and Dietterich [22] with five others that are common in face analysis problems. Each type of corruption $c \in C$ is applied to the original images with five intensity levels $s \in S$, where $S = \{1, 2, 3, 4, 5\}$.

We grouped the corruptions in four categories, namely blur, noise, digital and mixed; details about their implementation and the values of the used parameters1 are reported in Table 1. In Fig. 1, we show some examples of the considered corruptions (rows) with different severity (columns).

The people framed in real environments are typically not aware of the presence of the camera. Therefore, the movement of their faces, often very sudden, can cause blur on the acquired image. In addition, manual blur can be added in the face pre-processing to improve the image given as input to a neural network. To take into account these possible corruptions, we consider the blur category, including Gaussian blur, defocus blur, zoom blur and motion blur. The Gaussian blur is typically applied as a pre-processing step on the acquired image to mitigate the effect of acquisition noise and to enhance the image patterns at different scales. The defocus blur occurs when using cameras with limited depth of field (DoF) deployed in scenarios with large DoF. Zoom blur appears when a person moves towards the camera rapidly, so increasing the size of the face captured by the sensor. Motion blur occurs when a subject suddenly moves, quickly changing his face pose.

Image noise is due to the electronic noise produced by the image sensor at high temperatures or with long exposure; it appears as random speckles that can substantially degrade the image quality. In particular, the noise corruptions include Gaussian noise and shot noise. The first corruption increases proportionally with the temperature of the CMOS sensor; considering that in real environments the camera is always active and the sensor works perpetually at high temperatures, this source of noise is very common. The shot noise is a corruption occurring in case of high exposure, that is typical for installations in shop windows or for cameras pointing to the store entrance.

Other corruptions very relevant in real environments are due to manual camera settings, image compression and image transformation. To improve the image rendering, the automatic gain control (AGC) and the digital wide dynamic range (DWDR, also called Dynamic Contrast) dynamically modify the brightness and the contrast of the acquired image according to the environmental conditions. Often, the image processing is performed on an external

---

1 The code is available at https://github.com/MiviaLab/emotion-robustness
| Corruption                  | Parameter | Value (per severity) | Description                                                                                                                                 |
|----------------------------|-----------|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
|                            |           | s = 1               | s = 2                             | s = 3                             | s = 4                             | s = 5                             |
| Gaussian blur              | $\sigma$ | 1                    | 1.8                               | 2.6                               | 3.4                               | 4                                 |
|                            |           | $I_c(x, y, d) = I(x, y, d) \cdot G_\sigma(x, y, \forall d \in \{R, G, B\})$ where $G_\sigma(x, y) = e^{(x^2+y^2)/2\sigma^2}$ |
| Defocus blur               | $r$       | 1.5                  | 2                                 | 2                                 | 2.5                               | 3                                 |
|                            | $\sigma$ | 0.1                  | 0.2                               | 0.3                               | 0.4                               | 0.4                               |
|                            | $z$       | 1.11                 | 1.18                              | 1.26                              | 1.32                              | 1.4                               |
| Zoom blur                  | $\varepsilon$ | 0.01               | 0.01                              | 0.02                              | 0.02                              | 0.03                              |
|                            |           | $I_c(x, y, d) = I(x, y, d) \cdot (D_r(x, y) \cdot G_\sigma(x, y))$, being $D_r$ a disc shaped kernel with radius $r$. |
| Motion blur                | $r$       | 3.3                  | 5                                 | 5                                 | 5                                 | 6.7                               |
|                            | $\sigma$ | 1                    | 1.7                               | 2.7                               | 4                                 | 5                                 |
|                            | $\varepsilon$ | 0.01               | 0.12                              | 0.18                              | 0.24                              | 0.3                               |
| Gaussian noise             | $\sigma$ | 0.08                 | 0.12                              | 0.18                              | 0.24                              | 0.3                               |
| Shot noise                 | $q$       | 60                   | 29                                | 15                                | 8                                 | 5                                 |
| Contrast increase          | $q$       | 1.5                  | 1.9                               | 2.6                               | 3.3                               | 5                                 |

\(I_c(x, y, d) = I(x, y, d) + N(0, \sigma^2)\) 

\(I_c(x, y, d) = \text{Poisson}(I(x, y, d) \cdot q)/q\) 

\(I_c(x, y, d) = (I(x, y, d) - \mu_d) \cdot q + \mu_d\), where $\mu_d$ is the average value of the original image $I$ over the channel $d$. 

Implementation from the ImageMagick library, with random angle. 

Implementation from the ImageMagick library, with random angle. 

Implementation from the ImageMagick library, with random angle.
| Corruption                       | Parameter | Value (per severity) | Description                                                                 |
|---------------------------------|-----------|----------------------|-----------------------------------------------------------------------------|
|                                 |           | $s = 1$              | $s = 2$           | $s = 3$ | $s = 4$ | $s = 5$ |
|                                 |           |                     | $s = 1$  | $s = 2$ | $s = 3$ | $s = 4$ | $s = 5$ |
| Contrast decrease               | $q$       | 0.4                  | 0.33              | 0.24    | 0.16    | 0.1     | $I_c(x, y, v) = I(x, y, v) + q$, $\forall x, y$, where $v$ is the $v$ channel in hsv image representation. |
| Brightness increase             | $q$       | 0.1                  | 0.2               | 0.3     | 0.4     | 0.5     | |
| Brightness decrease             | $q$       | −0.1                 | −0.2              | −0.3    | −0.4    | −0.5    | |
| Spatter                         | $c_0$     | 0.65                 | 0.65              | 0.65    | 0.65    | 0.67    | Implementation by [22]. |
|                                 | $c_1$     | 0.3                  | 0.3               | 0.3     | 0.3     | 0.4     | |
|                                 | $c_2$     | 4                    | 3                 | 2       | 1       | 1       | |
|                                 | $c_3$     | 0.69                 | 0.68              | 0.68    | 0.65    | 0.65    | |
|                                 | $c_4$     | 0                    | 0                 | 0       | 1       | 1       | |
| JPEG compression                | quality   | 25                   | 18                | 15      | 10      | 7       | Implementation from the Pillow library. |
| Pixelation                      | $q$       | 0.6                  | 0.5               | 0.41    | 0.3     | 0.25    | $I_c(x, y, d) = I(\lfloor xf \rfloor/q, \lfloor yf \rfloor/q, d)$, $I_1 = BrightnessIncrease(I, s_b)I_c = ContrastDecrease(I_1, s_c)$ |
| Mixed 1 brightness increase +   | $s_b$     | 1                    | 2                 | 2       | 2       | 3       | |
| contrast decrease               | $s_c$     | 1                    | 1                 | 2       | 3       | 4       | |
| Mixed 2 brightness decrease +   | $s_b$     | 1                    | 2                 | 2       | 2       | 3       | |
| contrast decrease               | $s_c$     | 1                    | 1                 | 2       | 3       | 4       | |
| Mixed 3 gaussian noise +        | $s_g$     | 1                    | 2                 | 2       | 3       | 3       | |
| brightness decrease + contrast  | $s_b$     | 1                    | 2                 | 2       | 2       | 2       | |
| decrease                        | $s_c$     | 1                    | 1                 | 2       | 3       | 4       | |
| Corruption                                                                 | Parameter | Value (per severity) | Description                                                                                           |
|---------------------------------------------------------------------------|-----------|----------------------|-------------------------------------------------------------------------------------------------------|
| Mixed 4 motion blur + contrast decrease + brightness decrease              | $s_m$     | 2 3 4 5 5             | $I_1 = \text{MotionBlur}(I, s_m)$ $I_2 = \text{BrightnessDecrease}(I_1, s_b)$ $I_c = \text{ContrastDecrease}(I_2, s_c)$ |
|                                                                           | $s_b$     | 1 1 2 2 2             |                                                                                                       |
|                                                                           | $s_c$     | 1 1 2 1 3             |                                                                                                       |
| Mixed 5 pixelation + contrast decrease + brightness decrease               | $s_p$     | 1 2 3 4 4             | $I_1 = \text{Pixelation}(I, s_p)$ $I_2 = \text{BrightnessDecrease}(I_1, s_b)$ $I_c = \text{ContrastDecrease}(I_2, s_c)$ |
|                                                                           | $s_b$     | 1 2 2 2 3             |                                                                                                       |
|                                                                           | $s_c$     | 1 1 2 1 3             |                                                                                                       |

$I(x, y, d)$ refers at the original image and $I_c(x, y, d)$ at the corrupted image, while $d \in \{ R, G, B \}$ indicates the channel in the RGB space. Mixed corruptions are obtained as a combination of basic corruptions with different severity.

2 The $r$ and $\sigma$ parameter value are expressed in pixels and referred to a $48 \times 48$ image. The value of the parameter is scaled proportionally with larger images.

3 Implementation from the SciPy library: [https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.zoom.html](https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.zoom.html)

4 [https://imagemagick.org/api/effect.php#MotionBlurImage](https://imagemagick.org/api/effect.php#MotionBlurImage)

5 [https://pillow.readthedocs.io/en/3.1.x/handbook/image-file-formats.html#jpeg](https://pillow.readthedocs.io/en/3.1.x/handbook/image-file-formats.html#jpeg)
Fig. 1 Examples of corruptions. The first column contains images from the original RAF-DB test set, while the others depict the versions obtained by applying the considered corruptions with increasing value of severity (from 1 to 5).
server, acquiring the frames in motion JPEG (MJPEG) to reduce the required bandwidth. The compressed image may lose quality and details. Furthermore, the faces are rescaled for adaptation to the input size of neural networks for emotion recognition. We group all these corruptions in the digital category. They include contrast increase, contrast decrease, brightness increase, brightness decrease, spatter, JPEG compression and pixelation. Contrast and brightness are variations of the image due to the lighting conditions or to specific camera settings. Brightness variations occur outdoor with daylight, while in indoor environments it depends on the artificial illumination. Contrast corruptions occur when the difference between the brightest and the darkest pixels in the image (dynamic range) is high. The spatter consists of random patterns of obstructions on the camera lens. We simulate this effect by adding bright occlusions for low corruption severity and dark patterns for higher corruption severity. JPEG compression is often used to reduce the amount of data transferred on networks to an external server for the real-time processing of the images and can introduce compression artifacts. We reproduce its effects by gradually decreasing the quality of the compression. Image pixelation happens when an image is scaled from a lower resolution to a higher one. It is typical for face analysis in real environments, since the faces have a smaller resolution than the input size of the neural networks used for facial soft biometrics recognition.

The mixed corruptions, that we added to those proposed by [22], consist of combinations of single corruption types that simulate other challenging real-world scenarios. We create five mixed corruptions. We combine the contrast decrease with brightness increase and brightness decrease, which occur when the automatic exposure control of the cameras targets different elements in the scene and the resulting dynamic range of the images is compressed. In environments with low illumination, contrast and brightness decrease together: we combined it with added Gaussian noise, to simulate high gain on the camera sensor, and with motion blur, usually caused by long shutter times. We also combined contrast and brightness decrease with pixelation, to simulate faces at low resolution in dark environments.

For the evaluation, we adopted the experimental protocol proposed by [22]. Let \( E_d \) indicate the classification error on a data set \( d \), computed as the ratio between the number of wrongly classified images and the total number of images. Therefore, we use \( E_o \) to indicate the classification error on the original RAF-DB test set, and \( E_c \) for the classification error obtained on a set of images with corruption type \( c \in C \). Since the images with corruption \( c \) are provided with different levels of corruption severity, we compute the classification error \( E_c \) as the average of the errors obtained for each severity:

\[
E_c = \frac{1}{|S|} \sum_{s \in S} E_{c,s}
\]

where \( E_{c,s} \) is the classification error on samples with corruption type \( c \) and severity level \( s \).

Then, we compute the corruption error \( \tilde{E}_c \) as the classification error \( E_c \) normalized by the error \( E^b_c \) obtained by another classifier taken as the baseline:

\[
\tilde{E}_c = \frac{E_c}{E^b_c}
\]

Finally, we compute the mean corruption error \( \bar{E} \) as the average of the \( \tilde{E}_c \) over all the corruptions \( c \in C \):

\[
\bar{E} = \frac{1}{|C|} \sum_{c \in C} \tilde{E}_c
\]
where $|C|$ is the cardinality of the set $C$. The smaller the value of $E$, the better the robustness of the method with respect to the corruptions.

In addition to the absolute error, we compute the relative corruption error $\tilde{R}E_c$ as the gap between the classification error on the original test set $E_o$ and that on the corrupted sets $E_c$, normalized with respect to the error gap achieved by the baseline. For a specific corruption type $c$, it is defined as:

$$\tilde{R}E_c = \frac{E_c - E_o}{E^b_c - E^b_o}$$

Finally, we compute the mean relative corruption error $\overline{RE}$, i.e., the average of the $\tilde{R}E$ achieved for all the considered corruptions.

$$\overline{RE} = \frac{1}{|C|} \sum_{c \in C} \tilde{R}E_c$$

3.1.2 RAF-DB-P

We created the RAF-DB-P test set by applying 10 types of perturbation to the images of the RAF-DB test set. In contrast to the corruptions, a perturbation concerns a sequence of frames: it consists of a small corruption incrementally applied to subsequent frames. Its temporal character determines substantial appearance changes between the first and last frame of a sequence. The perturbations challenge the performance of the recognition methods when they are applied in real scenarios and have to perform analyses over time.

We selected a set $P$ of perturbations that typically occur when dealing with faces in real scenarios. For each perturbation $p \in P$, an image in the RAF-DB test set is replicated into a sequence of 30 frames, each with slight changes with respect to the previous one. In Fig. 2, we show few examples of perturbed image sequences.

We group the perturbations in four classes, namely blur, noise, digital and transformation. To implement the first category, a different noise pattern is applied to the original images, independent among different frames of the sequence. For all the other perturbations, the modifications are applied to each frame of the sequence in an incremental way.

The blur category includes Gaussian blur and motion blur. Perturbed sequences with Gaussian blur are generated by applying a Gaussian blur corruption incrementally frame after frame: the standard deviation of the Gaussian blur for the $j$-th frame is $\sigma_j = 0.25 + 0.035j$, where $j \in [0, 29]$. Similarly, sequences with motion blur are generated applying a motion blur corruption with $r = 10$ and $\sigma = 3$ and motion angle $\theta_j$ that increases for consecutive frames as $\theta_j = (4 \cdot j) ^{\circ}$, where $j \in [0, 29]$.

Noise perturbations include Gaussian noise and shot noise. The perturbed sequences are generated applying the corresponding Gaussian and shot noise corruption with the severity $s = 2$ repeatedly to the original image, as the noise has no inter-frame dependency.

The digital perturbations are spatter and brightness increase. For the spatter perturbation, a pattern of translucent water droplets is created and superimposed to the first frame causing occlusions. For subsequent frames, the droplet pattern is incrementally blurred and shifted downwards on the image. The implementation and the parameters are those used by Hendrycks and Dietterich [22]. In the case of brightness perturbations, subsequent frames of a sequence are modified by applying an incremental brightness increase corruption to the previous frame: at the $j$-th frame, the control parameter $q_j$ has value $q_j = \frac{j-15}{50}$, with $j \in [0, 29]$.

Finally, the transformation category includes translation, rotation, scale and shear perturbations. Translation consists in shifting the image for one pixel to the right with respect
to the previous frame. Rotation is implemented by rotating the face image one degree counterclockwise at every consecutive frame in the range $[-15; 15]$ degrees. The scale transformation is obtained as follows. We define a region of interest around the face in an image of the RAF-DB test set of size $w \times w$ and centered at location $(x, y)$. We change the size of the region in the $j$-th frame of the perturbed sequence to $w_j \times w_j$, where $w_j = w \cdot (0.79 + \frac{29-j}{39}0.51)$ with $j \in [0, 29]$, and keep its center location fixed in $(x, y)$. This results in a loose face crop in frame 0 and increasingly tighter crops in subsequent frames.

These three transformations are experienced in real-world applications due to the imprecision of face detection and face alignment algorithms. Typically, smoothing algorithms such as the Kalman filter are used to track a face producing a delay, which causes slight but continuous variations of position, rotation and scale. The shear transformation simulates a change of perspective of the face by bending the image according to the affine transformation $(x', y') = \begin{pmatrix} 1 & \alpha \\ \alpha & 1 \end{pmatrix}(x, y)$, where the parameter $\alpha \in [-0.15; 0.15]$ is increased in steps of 0.01.

![Fig. 2 Examples of perturbations. Each row contains a perturbation type, whose temporal sequence is represented from left to right (one every three frames is shown)](image-url)
To evaluate the stability of classification on perturbed image sequences, we compute the flip probability ($F$). It measures the likelihood that the predicted class changes across consecutive frames of a sequence of $N$ frames $x = \{x_1, x_2, \ldots, x_N\}$ $|x_i \in \mathcal{X}|$. Given a classification method $f : \mathcal{X} \rightarrow \{1, 2, \ldots, M\}$, which assigns one of $M$ classes to images $x_i \in \mathcal{X}$, the flip probability for the sequence $x$ is computed as the average of the number of changes of the classification output across consecutive frame. It is defined as:

$$F(x) = \frac{1}{N-1} \sum_{j=2}^{N} [1 - \delta(f(x_j), f(x_{j-1}))]$$

(6)

where $\delta(\cdot, \cdot)$ is the Kronecker delta function; note that the function $1 - \delta(f(x_j), f(x_{j-1}))$ assumes value equal to 0 if the method $f$ predicts the same class for the frames $x_{j-1}$ and $x_j$, 1 otherwise.

Let us consider a perturbation type $p$ from the set $P$. We compute the flip probability for the set of sequences with perturbation $p$ as the average of the flip probability computed for each sequence $x \in p$:

$$F_p = \frac{1}{|p|} \sum_{x \in p} F(x)$$

(7)

As the $F_p$ may assume values in different ranges for different perturbations $p \in P$, we normalize its value by the corresponding flip probability $F^b_p$ achieved by another classifier taken as the baseline. The normalized flip probability is computed as:

$$\tilde{F}_p = \frac{F_p}{F^b_p}$$

(8)

The overall measure of the flip rate for all perturbation types in the set $P$ is the average of the flip rate of the single perturbations $\tilde{F}_p$ and is called mean Flip Rate and is defined as:

$$\bar{F} = \frac{1}{|P|} \sum_{p \in P} \tilde{F}_p$$

(9)

The smaller this value, the better the stability of the method against perturbations.

### 3.2 Methods

We benchmarked the performance of different convolutional network models, namely VGG, SENet, Xception, DenseNet and ARM, as well as that of a method based on Local Binary Patterns (LBP) and an SVM classifier, that achieved the state-of-the-art performance prior to the development of deep-learning models for automatic feature learning. We chose different methods with peculiar architectures to evaluate the impact of the specific design choices on the overall network robustness.

**VGG** We selected the VGG-Face network as the baseline for our analysis [35]. It is one of the most widely used networks for facial soft biometrics analysis, based on the VGG-16 architecture [38], trained for face recognition on about 1M images. VGG-Face achieved high performance in the face recognition [35] and age estimation [6] problems. The VGG architecture was shown to have good generalization capabilities also for small data sets [11]. Its lightweight model has only 130k parameters.

**SENet** Designed by Hu et al. [23], it achieved the state-of-the-art performance on emotion recognition [2]. The version that we apply, namely SENet-50 (i.e. a SENet with 50
layers), is based on ResNet-50 [21]. SENet uses Squeeze-and-Excitation modules, adaptively weighting the input channels when computing the output feature maps. The intuition to explicitly model the inter-dependencies between the channels was demonstrated to be a winning strategy, reducing the error by 25% on the ImageNet benchmark over its plain ResNet counterpart [23]. The trained model has about 25M parameters.

**DenseNet** The Densely Connected Convolutional Network is designed to increase the representation capabilities while reducing the number of trainable parameters [24]. It adopts a dense connectivity scheme to improve the information flow between the layers, by forwarding and concatenating feature maps to subsequent layers and using a growth rate to establish how much each layer contributes to the global state. The architecture of DenseNet leverages transition layers, which do convolution and pooling between connected dense blocks, to normalize the size of the feature maps computed by different layers. We used the DenseNet-121-32 network, which has 121 layers and growth rate \( k = 32 \). It obtained a good trade-off of performance and size on the ImageNet classification challenge. It has 7M parameters, substantially less than the other architectures with comparable performance. To the best of our knowledge, the DenseNet architecture has not yet been benchmarked on the problem of facial expression recognition.

**Xception** Proposed by Chollet [7], this network architecture inherits the large use of identity connections of the ResNet architecture and combines it with inception modules and depth-wise separable convolutions. Its architecture holds the advantage of factoring convolutions into different branches and separates the convolutions in depth-wise and point-wise components, so retaining the performance of the network while reducing the number of parameters to 20M.

**LBP-SVM** LBP is the acronym for Local Binary Patterns [33], a method for dynamic texture description that achieved the state-of-the-art performance in facial expression recognition before CNNs became popular for image classification [36]. The LBP descriptor is invariant to translations and rotations, and robust to illumination changes. Each pixel is compared to its 8 neighbours: the 8-digit binary code will have a 1 or a 0 in each position according to the pixel being brighter than its neighbour or not. The extended operator, devised by Ojala et al. [34], considers neighbourhoods of different size. Following the method proposed by Shan et al. [36], we compute the \( LBP_{\mu}^{\alpha,2} \) by Ojala et al. [34], that means that we consider 8 neighbours, but with a distance of 2 from the central pixel. Furthermore, we do not consider all the possible 256 patterns as bins for the histogram, but only the 59 uniform ones, as they are shown to preserve large part of the information while reducing the dimensionality. We resample the image to 110 \( \times \) 150 pixels, then we divide it in 6 \( \times \) 7 blocks and compute the histogram on each of them. The feature vector is composed of the concatenation of the block-wise vectors. As suggested in the reference method, we use an SVM with a RBF kernel as classifier and we perform a grid search to select the optimal values for the parameters \( C \) and \( \gamma \) for the RBF-SVM; selected values are \( C = 4 \) and \( \gamma = 3 \cdot 10^{-6} \).

**ARM** Recently proposed by [37], it is a novel architecture that holds the best accuracy on the RAF-DB test set. The acronym name stands for Amending Representation Module. This module replaces the pooling layers of a ResNet-18 CNN. The authors demonstrated it allows to reduce the weight of eroded features and partially remove the side effects of padding on the output feature map, and to decompose the facial features and simplify the
emotion representation learning. We used the implementation made available together with the original paper.

3.3 Training procedure

The pre-training of a CNN on face recognition demonstrated its effectiveness for improving the classification performance on tasks like facial gender recognition and age estimation [3, 6]. Hence, we start from methods pre-trained on the VGG-Face2 data set. When available, we used the model weights released by the authors (e.g. for VGG and SENet). In the case of Xception and DenseNet we pre-trained the networks ourselves by following the protocol of Cao et al. [5].

Subsequently, we trained the networks on the RAF-DB data set for 220 epochs. We used the Stochastic Gradient Descent optimizer with a momentum equal to 0.9, a batch size of 64, and an initial learning rate equal to 0.002 that we halved every 40 epochs. We used a cross-entropy loss with a weight decay of 0.005. We resized the input images to native input size of each network, i.e. 299 × 299 for Xception, 224 × 224 for all the others, and zero-centered every channel by subtracting the mean computed on the VGGFace2 data set. Training on zero-centered data is very common and improves the convergence of the loss function.

We applied a standard augmentation commonly used for face analysis and emotion recognition [46]. The augmentation strategy, that we call hereinafter basic, includes random rotation, shear, cropping, horizontal flipping and change of brightness and contrast. The horizontal flipping is applied with a probability of 0.5, the random rotation is chosen in a range of ±10 degrees, while the shear matrix \( \begin{pmatrix} \alpha_1 & 1 \\ \alpha_2 & 1 \end{pmatrix} \) uses two independent values of \( \alpha_1 \) and \( \alpha_2 \) between 0 and 0.1. The contrast can be randomly increased or decreased by a factor up to 2, and brightness increases or decreases up to 20% of the maximum value.

3.4 Robustness and stability improvement

On one hand, the robustness of convolutional models to corruptions and perturbations is affected by the quality of the training data. On the other hand, certain architectural components of the networks may influence the performance when the input data undergoes specific types of transformation. For instance, the max-pooling (or strided convolution) layers introduce aliasing in the intermediate feature maps and the networks do not provide stable predictions on translated inputs.

We evaluated the impact that targeted expansion of the input data, using the AutoAugment data augmentation technique, and a modification of the CNN architecture with the use of an anti-aliasing filter before down-sampling, have on the robustness of the existing network models. We also evaluated the combined contribution of data- and architecture-related modifications on the robustness of SOTA methods.

3.4.1 AutoAugment

AutoAugment is an automated procedure for determining a set of data augmentation policies for a specific image classification problem [9]. It searches augmentation policies in a space with 16 basic operations, namely shear, translate, rotate, auto-contrast, invert, equalize, solarize, posterize, contrast, color balance, brightness, sharpness, cutout, sample pairing. The augmentations are applied with a certain probability and magnitude, for a total of 15\( k \) policies. The search algorithm, based on Reinforcement Learning, uses a Recurrent Neural
Network as controller and a Proximal Policy Optimization as training strategy. The result of the training is a set including the 25 best policies for training a network on the target data set. The authors demonstrated the improvement achieved with AutoAugment on four benchmark data sets, namely CIFAR-10, CIFAR-100, SVHN and ImageNet, and its superiority w.r.t. other augmentation techniques. A faster policy search algorithm called Fast-AutoAugment was developed by Lim et al. [30], based on density matching.

We evaluated the contribution that the AutoAugment data augmentation makes to improve the robustness of the considered methods to common corruptions and perturbations. We learned and made available the augmentation policies on the RAF-DB data set. For searching the augmentation policies, we applied the Fast-AutoAugment method. We used a reduced version of the RAF-DB data set that includes 20% of the training data and 40% of the validation data.

We append the suffix $|_a$ to the name of the methods that are trained with AutoAugment data augmentation, e.g. VGG$_{|_a}$ indicates the VGG method trained with AutoAugment. The need of learning a new set of AutoAugment sub-policies highlights the fact that data augmentation techniques are dataset-specific and do not generalize well to different input images.

### 3.4.2 Anti-aliasing filters

Although the convolution operator is translation-invariant, current CNN are not, as shown by Zhang [47]. This is caused by local pooling strategies, which introduce aliasing into intermediate representations computed inside the networks. Thus, small translations can cause dramatic performance drops. Zhang [47] demonstrated that a low-pass filter (LPF) before down-sampling reduces the aliasing in CNNs, according to the Nyquist-Shannon theorem of sampling.

We modified existing networks for face analysis and analyzed the impact that an LPF has on the robustness to several types of corruption and perturbation. We considered the three LPFs of different size: $2 \times 2$ is a rectangular filter $[1, 1]$, the $3 \times 3$ triangle filter is given by the convolution of two box filters $[1, 2, 1]$, and the $5 \times 5$ binomial filter is given by the repeated convolution of rectangular filters $[1, 4, 6, 4, 1]$.

We append the suffix $|_r$, $|_t$ and $|_b$ to the name of the methods that use the $2 \times 2$, $3 \times 3$ and $5 \times 5$ LPF, respectively.

### 4 Results and discussion

We carried out experiments to evaluate the robustness of SOTA methods for facial emotion recognition to corruptions and perturbations of the input data. In the following of the section, we report the results that we achieved on the RAF-DB, RAF-DB-C and RAF-DB-P data sets with the considered existing methods. We discuss the impact that the AutoAugment data augmentation and the insertion of anti-aliasing filters within their architecture have on the performance on corrupted and perturbed data in terms of robustness, generalization abilities and stability of the classification output.

---

2The optimized policies are public at https://github.com/MiviaLab/emotion-robustness.
4.1 Baseline results

We trained the VGG, SENet, DenseNet, Xception, LBP-SVM and ARM methods on the RAF-DB original training set and tested on the RAF-DB, RAF-DB-C and RAF-DB-P test sets. These experiments evaluate the ground capabilities of the considered methods for facial emotion recognition when input images are subjected to corruptions and perturbations. Here, we take VGG as the baseline to compute the values of $E$, $RE$ and $F$ for other methods. We report the results in Table 2.

ARM achieves the lowest classification error ($E = 8.05\%$) on the original RAF-DB test set. According to the typical benchmark evaluations of classification algorithms, ARM would be selected as the best performing method on these data. However, the results obtained on the RAF-DB-C and RAF-DB-P test sets gave contrasting insights. We observed a degradation by more than 10\% of the error for all the considered methods, as seen in Table 2.

On the RAF-DB-C test set, ARM achieved the lowest corruption error ($Ec = 23.93$), followed by VGG ($Ec = 26.38$). However, ARM performed substantially worse than VGG in terms of relative corruption error, as indicated by the $RE = 1.435$. We observed that, although ARM holds the best accuracy on the RAF-DB-C dataset, its performance degradation due to corruptions is larger than VGG, showing poor generalization capabilities. This indicates that the current approaches to benchmark recognition algorithms do not evaluate the generalization and robustness properties of these methods, hindering a proper characterization of the performance of such algorithms to be deployed in real-world scenario.

SENet performed worse than VGG: $E = 1.033$ and $RE = 1.089$ indicate that the corruption error and the relative corruption error are 3.3\% and 8.9\% higher than that of VGG. DenseNet, instead, achieved a lower relative corruption error ($RE = 0.863$). As the $RE$ measures the degradation of the performance of a given method on corrupted input with respect to its performance on the original input, we interpret this result as an intrinsic capability of the DenseNet architecture to generalize well with respect to corruptions. We conjecture that the forward connections within a dense block of the DenseNet architecture allow to repeatedly compute feature maps that catch highly complex characteristics of the input and make the network more robust with respect to local changes in the images.

The baseline VGG has the highest stability when dealing with perturbations, i.e. it achieved the lowest probability of flipping its prediction between consecutive perturbed frames. The normalized flip probability $F$ achieved by the other methods is higher than that

| Network  | $E_{RAF-DB}$ | $E_{RAF-DB-C}$ | $E$ | $RE$ | $F$ |
|----------|--------------|----------------|-----|------|----|
| VGG      | 14.28        | 26.38          | 1.000 | 1.000 | 1.000 |
| SENet    | 13.69        | 27.93          | 1.033 | 1.089 | 1.162 |
| DenseNet | 15.91        | 27.43          | 1.027 | 0.863 | 1.355 |
| Xception | 17.35        | 31.94          | 1.208 | 1.179 | 1.789 |
| LBP-SVM  | 24.92        | 43.14          | 1.364 | 1.341 | 2.452 |
| ARM      | 8.05         | 23.93          | 0.903 | 1.435 | 1.345 |

We report the classification error on the original test set ($E_{RAF-DB}$) and on the corrupted one ($E_{RAF-DB-C}$), while $E$, $RE$ and $F$ are normalized with respect to the results of VGG.
of VGG by 16.2% for SENet, 34.5% for ARM, 35.5% for DenseNet and 78.9% for Xception. Finally, we noted that the method based on LBP and SVM achieved performance not comparable with the CNNs.

The controversial results of this first analysis show that corruptions and perturbations are not negligible aspects, even for the method achieving the best performance on the original dataset, and must be treated carefully when deploying a network for emotion recognition.

4.2 Results with AutoAugment

We analyzed the impact of the AutoAugment data augmentation strategy on the robustness and stability of the considered methods. We trained them using a set of data augmentation policies that we determined for the RAF-DB data set, according to the procedure proposed by Lim et al. [30]. We compare the results of the considered methods with those obtained by the corresponding variants trained with AutoAugment, which we specify by adding the suffix $|a$ to the method name. We computed the $E$, $RE$ and $F$ using the original methods as the baseline.

As shown in Fig. 3, the AutoAugment policies are effective in improving the robustness of the networks to image corruptions. All the considered methods achieved a value of $E$ and $RE$ lower than that of the corresponding baseline. The improvement registered for VGG$_{|a}$ is relevant. It achieved $RE = 0.784$, which measures a reduction of the relative corruption error with respect to the baseline by 21.6%. SENet$_{|a}$ benefits the most from the use of AutoAugment as it achieved $E = 0.894$ and $RE = 0.8$. DenseNet$_{|a}$ and Xception$_{|a}$ achieved lower results, namely $E = 0.910$ and $RE = 0.921$ by the former and $E = 0.946$ and $RE = 0.936$ by the latter. AutoAugment is effective to reduce the impact of corruptions on the performance of the existing methods; this is confirmed by the fact that by applying this strategy on VGG, SENet and DenseNet we are able to obtain a corruption error very close to the one achieved by a more complex method like ARM (for detailed results, see Table 3 in the Appendix A).

On the RAF-DB-P data set, only SENet$_{|a}$ benefits from the use of AutoAugment ($F = 0.938$), while the other methods do not improve their stability.

![Fig. 3 Results achieved by the considered methods trained with AutoAugment. For each method, the mean error $E$ (left plot), the relative error $RE$ (middle plot) and the flip rate $F$ (right plot) are computed using the corresponding method without AutoAugment as the baseline, represented as the 1.0 horizontal line](image-url)
4.3 Results with anti-aliasing filters

We evaluated the impact that using anti-aliasing filters before the down-sampling layers has on the robustness of the considered methods. We computed the $E$, $RE$ and $F$ for each method using as the baseline the corresponding original network.

The results that we achieved (see Fig. 4) show different effects on different architectures. DenseNet$_r$ substantially benefits from the use of anti-aliasing filters. Its robustness to corruptions improved, with a reduction of the $E$ to 0.934 and of the $RE$ to 0.89. The stability to perturbations is also enhanced, as the $F = 0.85$ indicates a reduction of the flip probability by 15% with respect to the baseline. VGG$_r$ achieved $F = 0.9$, while the results on corruptions are negligible. For SENet, the impact of the anti-aliasing filters is limited, while for Xception it is pejorative. It is worth noting that the size of the filter that contributes to the best improvement is not the same for all the methods. The use of the anti-aliasing filter is an effective strategy for improving the performance of DenseNet against corruptions and perturbations, making its corruption error close to the one achieved by ARM, and it is also an effective technique for increasing the stability of VGG.

4.4 Results with combined anti-aliasing filters and AutoAugment

The results obtained applying AutoAugment and the anti-aliasing filters showed that these modifications allow to improve the robustness of the considered methods. AutoAugment improves the robustness to corruptions, while the anti-aliasing filter is more effective to achieve better stability against perturbations.

We applied these two techniques together, and computed the $E$, $RE$ and $F$ for each method using as the baseline the corresponding method without modifications. In Fig. 5, we report the obtained results.

DenseNet$_{r,a}$ achieved the best overall performance in terms of classification error ($E = 13.59$), as well as robustness and generalization to the corruptions in the RAF-DB-C data set ($E = 0.887$ and $RE = 0.676$). Its stability to perturbations also improved ($F = 0.894$).

![Results of the methods that use anti-aliasing filters](image)

**Fig. 4** Results achieved by the considered architectures when enhanced with anti-aliasing filters. The three bars from each group bar represent the use of rectangular, triangular and binomial filters respectively; lower is better. For each method the mean error $E$ (left plot), the relative error $RE$ (middle plot) and the flip rate $F$ (right plot) are computed using the corresponding method without any anti-aliasing filters as baseline, represented as the 1.0 horizontal line.
Fig. 5 Results achieved by the considered architectures when enhanced with anti-aliasing filters and trained with AutoAugment. The three bars from each group bar represent the use of rectangular, triangular and binomial filters respectively; lower is better. For each method the mean error $\bar{E}$ (left plot), the relative error $\bar{RE}$ (middle plot) and the flip rate $\bar{F}$ (right plot) are computed using as baseline the corresponding method trained without AutoAugment and without any anti-aliasing filters, and represented as the 1.0 horizontal line.

Xception$_{t,a}$ achieved a better classification error ($\bar{E} = 15.88$) and robustness to corruptions ($\bar{E} = 1.062$ and $\bar{RE} = 0.848$) with respect to the corresponding baseline. On the perturbations in the RAF-DB-P data set, instead, the modifications to the architecture and training data worsen the stability of the method ($\bar{F} = 1.576$). This is attributable to the fact that this network requires larger input images, leading to sub-optimal performance for the problem at hand, possibly caused by the need to scale up the small images from the RAF-DB data set. For SENet, we measured an improvement of robustness to corruptions ($\bar{RE} = 0.782$ and $\bar{E} = 0.932$ achieved by SENet$_{t,a}$), but not on the perturbations. Conversely, VGG benefited from these modifications and achieved better stability ($\bar{F} = 0.783$).

The combined use of AutoAugment and the anti-aliasing filters allows to improve the robustness of the considered models. It has a substantial impact on DenseNet, which achieved the lowest error on corrupted images. It performs better than ARM on RAF-DB-C, although the latter has higher accuracy (more than 5% better) on the original test images. VGG with AutoAugment and anti-aliasing filters achieved the best stability with respect to perturbations. These results are further evidence that ARM, although being the state-of-the-art method on RAF-DB, lacks proper robustness to real-world scenarios.

### 4.5 Robustness, generalization and stability

In Fig. 6a, we show a scatter plot of the mean corruption error $\bar{E}$ (x-axis) and the relative corruption error $\bar{RE}$ (y-axis) achieved by the considered methods. The axis direction is inverted for visualization purposes. In order to directly compare the performance of different networks, we normalized the results reported in Fig. 6a using as common baseline the results of VGG. The methods in the top-right quadrant (green region) perform better than the baseline in terms of robustness (lower $\bar{E}$) and generalization (lower $\bar{RE}$) to corruptions. It is worth pointing out that the generalization is intended as the capability of keeping the gap between the classification error on original and corrupted data very small. The points in the top-left and bottom-right quadrants (yellow regions) correspond to the methods achieving an improvement either of the $\bar{E}$ or the $\bar{RE}$ with respect to the baseline. The bottom-left
Fig. 6  a Results of the evaluation of the robustness and generalization of the considered methods with respect to corruptions, in terms of mean error $\overline{E}$ and relative error $\overline{RE}$. b Results of the evaluation of the robustness to corruptions and the stability to perturbations of the considered methods, in terms of mean error $\overline{E}$ and flip rate $\overline{F}$. The direction of the axes is inverted so that the points in the top right quadrant (green region) correspond to the results of methods that improve their performance w.r.t. the baseline, namely VGG. Different colors represent different network architectures: red is VGG, blue is SENet, green is DenseNet and yellow is Xception. The $\bullet$ marker refers at the original methods. The $\blacksquare$, $\triangle$ and $\star$ markers indicate the methods with anti-aliasing filters of type rectangular, triangular and binomial, respectively. The empty markers represent methods trained with the AutoAugment strategy. The $\times$ marker refers to ARM.

The methods based on DenseNet (green markers) achieved the best performance on the RAF-DB-C data set. SENet (blue markers) benefits from the use of the AutoAugment augmentation policies and anti-aliasing filters in its architecture, and improves on the performance of its original version. The Xception-based methods (dark-yellow points) show low robustness to corruptions: their results are all located in the left quadrants of Fig. 6a. VGG$_{16}$ is the only VGG-based method (red points) that achieved better robustness and generalization to corruptions with respect the baseline.

We compared the performance of the considered methods by jointly evaluating their robustness to corruptions (i.e. the mean corruption error $\overline{E}$) and stability with respect to perturbations (i.e. the flip rate $\overline{F}$) and show a scatter plot of the results in Fig. 6b. The points in the upper quadrants (top-left yellow and top-right green quadrants) correspond to the methods that achieved a better stability against perturbations than the baseline. The methods based on VGG (red markers) achieved the best stability against perturbations, as they are located in the upper quadrants of the plot; however, they are less robust to corruptions than the other methods. The methods based on SENet (blue markers) and DenseNet (green markers) achieved good robustness ($\overline{E} = 0.932$ and $\overline{E} = 0.887$) to corruptions but are less stable to perturbations ($\overline{F} = 1.055$ and $\overline{F} = 1.144$, respectively). Xception-based methods (dark-yellow markers) achieved results worse than those of the other methods. Their results indeed place in the bottom-left quadrant of the plot in Fig. 6b.

The modifications to the training data and the use of anti-aliasing filters in the network architectures generally contributed to an increase of robustness to corruptions and stability.
to perturbations of the methods with respect to their original implementation. However, none of the modified methods achieved at the same time higher robustness and stability than the VGG, i.e. there are no points in the green quadrant in Fig. 6b. VGG methods perform more stable classification over perturbed frame sequences and also exploit the data- and architecture-related improvements better than other existing methods.

ARM achieves impressive accuracy on the original RAF-DB test set and the second best absolute performance on RAF-DB-C. However, the very high values of $R_E$ and $F$ (placing in the bottom-right quadrants in Fig. 6a and b) demonstrate a poor generalization capability with respect to corruptions and a limited stability to perturbations.

### 4.6 Robustness to categories of corruption and perturbation

We carried out an analysis of the performance of the considered methods on specific categories of corruptions and perturbations. In the Appendix B, we report the $E$, $R_E$ and $F$ values achieved for each corruption and perturbation category in Tables 5, 6 and 7, respectively.

The combination of AutoAugment and anti-aliasing filters improved the robustness of the considered methods to all the categories of corruptions, except for Xception that obtained results comparable with those of the VGG baseline. The methods based on DenseNet achieved a considerable improvement of the corruption error $E$ and of the relative corruption error $R_E$ with respect to the baseline on all the categories of corruptions. The improvement is, however, less evident on the corruptions of type blur; we observed that these corruptions are the most challenging for all the considered methods. This is attributable to the fact that convolutional networks tend to learn distinctive and discriminant features at high spatial frequencies, while blur preserves lower frequency patterns which are less relevant for classification [45].

Furthermore, DenseNet methods achieved the highest results in terms of stability against noise perturbations, obtaining a reduction of the flip rate $F$ up to more than 40% with respect to the baseline. Most of the improvement is attributable to the use of the anti-aliasing filters, while training the methods with AutoAugment has lower impact. In scenarios in which the recorded images are affected by noise, the DenseNet network architecture with anti-aliasing filters can be deployed to ensure robust recognition performance.

The impact of the anti-aliasing filters on the stability of the VGG methods against perturbations is positive and evident for the blur, noise and transformation types of perturbation. These methods achieved a reduction of the flip rate $F$ between about 10% and 25% with respect to the baseline. Digital perturbations, instead, are more challenging. The VGG-based methods also achieved good robustness with respect to noise, digital and mixed corruptions.

The SENet methods benefited the most from the use of AutoAugment and the anti-aliasing filters to improve the performance of the original network model with respect to corruptions of the type noise. The values of the errors $E$ and $R_E$ improved respectively by about 30% and 45% with respect to the baseline. However, the results of SENet methods are negatively influenced by the other types of perturbations, while showing slight improvements of robustness with respect to the other categories of corruption.

For the methods based on Xception, instead, the use of AutoAugment and the anti-aliasing filters did not contribute to substantially improve their robustness to corruptions and perturbations, apart a small reduction of the relative corruption error $R_E$ on the corruptions of type digital.
5 Conclusions and future works

The classification performance of the state-of-the-art methods based on convolutional networks for emotion recognition considerably drops when the input images are subjected to corruptions and perturbations. We showed that more comprehensive experiments are necessary to assess the performance of emotion recognition methods to be deployed in real-world scenarios. We thus constructed a benchmark data set on top of the RAF-DB test set that includes images with corruptions and perturbations that typically occur when the recognition systems are deployed in real scenarios.

We analyzed the impact that the AutoAugment data augmentation policies and the use of anti-aliasing filters before the down-sampling operations (e.g. max-pooling or strided convolutions) in convolutional networks have on the robustness of existing methods. Their combined use substantially contributed to the improvement of the robustness to corruptions, especially to those of the noise and digital type, of SENet and DenseNet. The methods based on VGG and use anti-aliasing filters, instead, showed the highest classification stability with respect to perturbations that affect subsequent frames of a sequence. The Xception methods are not suitable for facial emotion analysis in the wild.

We demonstrated that the common corruptions and perturbations are important aspects to take into account when evaluating methods to be deployed in real scenarios. ARM, which achieved the best accuracy on the RAF-DB test set, demonstrated limited generalization and robustness to corruptions and perturbations. None of the existing methods, which we modified with anti-aliasing filters and trained with extensive data augmentation, showed robustness to all the considered corruptions and perturbations, but most of them showed a better robustness than ARM. DenseNet obtained the best corruption error and VGG obtained the best flip rate.

We performed an analysis of the processing time required by the various methods on an NVIDIA Tesla V100 GPU. In the worst case, VGG required less than 10ms per image, Xception and SENet 21ms and DenseNet 35ms, thus all the proposed methods can process more than 25 fps. This result demonstrates their suitability for real-time emotion recognition applications.

Possible future research directions may include the design of new architectural components or training strategies that take into account the occurrence of perturbations between subsequent frames to reduce the decrease of the performance of the existing methods when dealing with corruptions and perturbations. Furthermore, the investigation of methods for balancing the accuracy over the emotion categories is an interesting future direction. We performed a preliminary analysis of the categorical error achieved by the various methods over the RAF-DB test set and reported the results in Table 4. We can note that also ARM, which obtains the best accuracy on RAF-DB, achieves impressive performance on neutral and happiness emotions, average accuracy on faces expressing surprise and sadness, values below the average on anger, fear and disgust samples. From the results, we can observe that AutoAugment allows to reduce the imbalance, but more advanced and tailored augmentation and training strategies would have a more tangible impact on the balanced accuracy.

To encourage further investigations, we made the data sets and AutoAugment policies, the experimental framework and the trained network models publicly available at: https://github.com/MiviaLab/emotion-robustness.
Appendix A: Overall results

Table 3 reports a complete overview of the error results that we achieved on the RAF-DB, RAF-BD-C and RAF-DB-P test sets, taking the performance of the non-modified VGG method as the baseline for the normalization of the $E$, $R E$ and $F$. The lower the values, the better the performance.

Table 4 reports a complete overview of the categorical error that we achieved on the RAF-DB test set. The lower the values, the better the performance.

Table 3 Results on the RAF-DB, RAF-DB-C and RAF-DB-P data sets. The best performance is highlighted in bold.

| Network | Type  | E   | $E$  | $RE$ | $F$  |
|---------|-------|-----|------|------|------|
| VGG     | VGG   | 14.28 | 1.000 | 1.000 | 1.000 |
|         | $VGG_a$ | 14.54 | 0.942 | 0.784 | 1.187 |
|         | $VGG_r$ | 15.13 | 1.062 | 1.011 | 0.810 |
|         | $VGG_l$ | 14.21 | 0.998 | 0.957 | 0.828 |
|         | $VGG_{l,a}$ | 14.37 | 1.003 | 0.976 | 0.857 |
|         | $VGG_{r,a}$ | 17.09 | 1.121 | 0.954 | 0.961 |
|         | $VGG_{l,a}$ | 17.35 | 1.058 | 0.800 | 0.881 |
|         | $VGG_{b,a}$ | 16.21 | 1.011 | 0.803 | 0.998 |
| SENet   | SENet | 13.69 | 1.033 | 1.089 | 1.087 |
|         | $SENet_{l,a}$ | 14.05 | 0.911 | 0.832 | 1.242 |
|         | $SENet_{r}$ | 16.93 | 1.152 | 1.061 | 1.041 |
|         | $SENet_{l}$ | 14.28 | 1.031 | 1.026 | 1.128 |
|         | $SENet_{b}$ | 18.43 | 1.097 | 0.802 | 1.431 |
|         | $SENet_{r,a}$ | 14.87 | 0.932 | 0.782 | 1.243 |
|         | $SENet_{l,a}$ | 14.54 | 0.940 | 0.824 | 1.308 |
|         | $SENet_{b,a}$ | 15.49 | 0.962 | 0.785 | 1.609 |
| DenseNet| DenseNet | 15.91 | 1.027 | 0.863 | 1.456 |
|         | $DenseNet_{l,a}$ | 14.77 | 0.927 | 0.759 | 1.572 |
|         | $DenseNet_{r}$ | 15.29 | 0.953 | 0.749 | 1.221 |
|         | $DenseNet_{l}$ | 14.64 | 0.952 | 0.816 | 1.233 |
|         | $DenseNet_{b}$ | 16.54 | 1.015 | 0.774 | 1.356 |
|         | $DenseNet_{r,a}$ | 16.57 | 0.978 | 0.700 | 1.276 |
|         | $DenseNet_{l,a}$ | 13.59 | 0.887 | 0.786 | 1.276 |
|         | $DenseNet_{b,a}$ | 16.31 | 0.956 | 0.676 | 1.476 |
| Xception| Xception | 17.35 | 1.208 | 1.179 | 1.937 |
|         | $Xception_{l,a}$ | 17.19 | 1.138 | 1.058 | 2.328 |
|         | $Xception_{r}$ | 16.44 | 1.232 | 1.331 | 1.541 |
|         | $Xception_{l}$ | 16.67 | 1.192 | 1.209 | 1.758 |
|         | $Xception_{b}$ | 19.19 | 1.263 | 1.092 | 1.704 |
|         | $Xception_{r,a}$ | 16.57 | 1.062 | 0.913 | 1.759 |
|         | $Xception_{l,a}$ | 15.88 | 1.066 | 0.980 | 1.842 |
|         | $Xception_{b,a}$ | 17.62 | 1.095 | 0.848 | 1.984 |
| ARM     | [37]  | 8.05 | 0.903 | 1.435 | 1.345 |
### Table 4: Error on the RAF-DB test set (SU=surprise, F=fear, D=disgust, H=happiness, SA=sadness, A=anger, N=neutral). The best result is highlighted in bold

| Network | Type | SU    | F    | D    | H    | SA   | A    | N    |
|---------|------|-------|------|------|------|------|------|------|
| VGG     | VGG  | 16.82 | 44.44| 63.64| 9.38 | 17.65| 16.77| 8.09 |
|         | VGG\_ia | 12.54 | 38.89| 39.61| 5.07 | 14.71| 33.54| 16.91 |
|         | VGG\_r  | 16.82 | 36.11| 52.60| 7.18 | 18.91| **16.15**| 10.44 |
|         | VGG\_t  | 12.54 | 61.11| 61.69| 6.08 | **11.13**| 23.60| 14.12 |
|         | VGG\_b  | 19.57 | 43.06| 59.74| 5.07 | 14.08| 25.47| 11.91 |
|         | VGG\_r,a | 19.27 | 47.22| 56.49| 7.43 | 19.75| 34.16| 7.65 |
| SENet   | SENet | 15.29 | 44.44| 55.19| 7.94 | 16.18| 19.25| 7.21 |
|         | SENet\_ia | 16.21 | 44.44| 53.25| 7.26 | 26.47| 20.50| 4.71 |
|         | SENet\_r  | 19.57 | 45.83| 55.84| 11.82| 22.90| 21.74| 7.35 |
|         | SENet\_t  | 17.43 | 41.67| 52.60| 8.45 | 16.18| 21.12| 8.38 |
|         | SENet\_b  | 19.27 | 50.00| 71.43| 11.40| 23.11| 32.92| 8.24 |
|         | SENet\_r,a | 22.63 | 38.89| 72.08| 9.46 | 20.59| 21.12| 9.56 |
|         | SENet\_t,a | 19.27 | 54.17| 69.48| 8.11 | 27.31| 28.57| 7.21 |
|         | SENet\_b,a | **9.79**| 58.33| 69.48| 8.11 | 19.54| 27.95| 11.76 |
| DenseNet| DenseNet | 21.71 | 33.33| 48.05| 7.69 | 11.76| 20.50| 20.15 |
|         | DenseNet\_ia | 21.10 | 37.50| 50.00| 5.66 | 19.33| 17.39| 13.38 |
|         | DenseNet\_r  | 18.04 | 43.06| 57.14| 8.28 | 13.24| **16.15**| 15.00 |
|         | DenseNet\_t  | 17.13 | 45.83| 58.44| 7.94 | 17.23| 17.39| 9.41 |
|         | DenseNet\_b  | 13.76 | 58.33| 65.58| 8.11 | 24.16| 19.25| 11.03 |
|         | DenseNet\_r,a | 23.55 | 29.17| 68.83| 8.70 | 15.34| 20.50| 13.53 |
|         | DenseNet\_t,a | 21.71 | 38.89| 51.30| 5.07 | 21.85| 20.50| 5.88 |
|         | DenseNet\_b,a | 16.21 | 66.67| 55.84| 4.05 | 19.33| 27.33| 18.68 |
| Xception| Xception | 19.27 | 47.22| 72.73| 8.70 | 21.64| 22.98| 11.47 |
|         | Xception\_ia | 19.27 | 41.67| 62.34| 9.80 | 16.60| 18.01| 16.47 |
|         | Xception\_r  | 20.49 | 36.11| 63.64| 10.73| 15.97| 29.81| 8.82 |
|         | Xception\_t  | 22.32 | 48.61| 66.23| 9.04 | 21.85| 23.60| 7.35 |
|         | Xception\_b  | 18.04 | 54.17| 69.48| 12.16| 31.09| 26.09| 6.91 |
|         | Xception\_r,a | 25.99 | 31.94| 57.79| 8.02 | 22.48| 26.71| 9.41 |
|         | Xception\_t,a | 16.82 | 41.67| 62.34| 6.17 | 21.64| 34.16| 10.74 |
|         | Xception\_b,a | 18.04 | 50.00| 68.18| 8.70 | 27.94| 26.71| 8.68 |
| ARM     | [37]  | 11.75 | **23.90**| **35.78**| **4.01**| 11.51| 18.64| **1.40**|

### Appendix B: Results on corruption and perturbation categories

In Tables 5 and 6 we report an overview of the $\overline{E}$ and the $\overline{RE}$ achieved by all the methods on the four categories of image corruption, namely blur, noise, digital and mixed corruptions. In Table 7 we show the resulting stability against perturbations in terms of $\overline{F}$. The values are all normalized by the baseline VGG results. The lower the numbers, the better the performance. The cells of the tables are filled in yellow color when the results are to
the baseline, while gradients towards green and red are used to indicate improvements or worsening with respect to the baseline, respectively.

### Table 5  Results on the RAF-DB-C data set in terms of $\bar{E}$

| net    | Models | Corruption | net| | net| | net| | net| | net| | net| |
|--------|--------|------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|        |        | blur       | 1.000 | 1.049 | 0.998 | 0.901 | 1.092 | 1.248 | 1.196 | 1.133 |
|        |        | noise      | 1.000 | 1.254 | 1.188 | 1.189 | 0.943 | 1.143 | 0.938 | 0.889 |
|        |        | digital    | 1.000 | 0.999 | 0.951 | 1.000 | 0.875 | 1.039 | 1.019 | 1.015 |
|        |        | mixed      | 1.000 | 1.083 | 0.990 | 1.014 | 0.916 | 1.125 | 1.050 | 0.956 |
| VGG    |        | blur       | 1.078 | 1.151 | 1.019 | 1.122 | 0.932 | 0.966 | 1.036 | 0.951 |
|        |        | noise      | 1.218 | 1.290 | 1.273 | 0.921 | 0.720 | 0.725 | 0.729 | 0.903 |
|        |        | digital    | 0.967 | 1.103 | 0.993 | 1.178 | 0.933 | 0.957 | 0.929 | 0.971 |
|        |        | mixed      | 1.017 | 1.166 | 0.997 | 1.033 | 0.940 | 0.952 | 0.963 | 0.981 |
| SENet  |        | blur       | 0.933 | 0.939 | 0.984 | 1.012 | 0.973 | 1.036 | 0.996 | 0.979 |
|        |        | noise      | 1.227 | 0.975 | 0.965 | 1.032 | 0.909 | 0.821 | 0.800 | 0.867 |
|        |        | digital    | 1.038 | 0.987 | 0.942 | 1.045 | 0.932 | 1.017 | 0.868 | 0.990 |
|        |        | mixed      | 1.007 | 0.909 | 0.936 | 0.966 | 0.890 | 0.940 | 0.862 | 0.927 |
| DenseNet | blur | 1.134 | 1.157 | 1.120 | 1.177 | 1.076 | 1.112 | 1.052 | 1.140 |
|        |        | noise      | 1.203 | 1.404 | 1.234 | 1.477 | 1.105 | 0.962 | 1.157 | 1.149 |
|        |        | digital    | 1.173 | 1.183 | 1.155 | 1.250 | 1.140 | 1.026 | 1.032 | 1.052 |
|        |        | mixed      | 1.318 | 1.293 | 1.284 | 1.264 | 1.197 | 1.110 | 1.089 | 1.097 |

Green color gradients indicate an improvement w.r.t. the baseline, while yellow and red color gradients indicate comparable or lower results than the baseline.
Table 6  Results on the RAF-DB-C data set in terms of $\overline{RE}$

| net    | Corruption | Models | net|\_| net|\_| net|\_| net|\_| net|\_| net|\_| net|\_| net|\_ |
|--------|------------|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|        |            | net|\_| net|\_| net|\_| net|\_| net|\_| net\_\_\_\_\_\_ | net\_\_\_\_\_\_ |
| VGG    | blur       | 1.000 | 1.035 | 1.004 | 0.795 | 1.169 | 1.293 | 1.186 | 1.126 |
|        | noise      | 1.000 | 1.352 | 1.285 | 1.280 | 0.906 | 1.116 | 0.798 | 0.766 |
|        | digital    | 1.000 | 0.840 | 0.841 | 0.982 | 0.577 | 0.702 | 0.607 | 0.747 |
|        | mixed      | 1.000 | 1.094 | 0.950 | 0.991 | 0.717 | 0.970 | 0.761 | 0.637 |
| SENet  | blur       | 1.203 | 1.105 | 1.040 | 0.947 | 0.885 | 0.890 | 1.056 | 0.819 |
|        | noise      | 1.348 | 1.343 | 1.409 | 0.735 | 0.588 | 0.567 | 0.584 | 0.812 |
|        | digital    | 0.954 | 0.892 | 0.945 | 0.878 | 0.837 | 0.760 | 0.727 | 0.731 |
|        | mixed      | 1.085 | 1.149 | 0.975 | 0.606 | 0.882 | 0.811 | 0.869 | 0.822 |
| DenseNet| blur       | 0.724 | 0.785 | 0.938 | 0.850 | 0.904 | 0.893 | 1.059 | 0.808 |
|        | noise      | 1.283 | 0.927 | 0.935 | 0.969 | 0.846 | 0.650 | 0.724 | 0.729 |
|        | digital    | 0.844 | 0.748 | 0.730 | 0.757 | 0.728 | 0.681 | 0.705 | 0.656 |
|        | mixed      | 0.832 | 0.651 | 0.791 | 0.658 | 0.651 | 0.592 | 0.708 | 0.579 |
| Xception| blur       | 1.036 | 1.165 | 1.061 | 0.988 | 0.943 | 1.068 | 0.992 | 1.046 |
|        | noise      | 1.197 | 1.531 | 1.267 | 1.543 | 1.056 | 0.863 | 1.180 | 1.107 |
|        | digital    | 1.068 | 1.244 | 1.129 | 1.015 | 1.041 | 0.765 | 0.881 | 0.672 |
|        | mixed      | 1.444 | 1.507 | 1.414 | 1.101 | 1.174 | 1.017 | 1.028 | 0.834 |

Green color gradients indicate an improvement w.r.t. the baseline, while yellow and red color gradients indicate comparable or lower results than the baseline.
Table 7. Results on the RAF-DB-P data set in terms of $F$.

| net       | Models | Perturbation | net | net|r | net|p | net|a | net|a,r | net|a,r,a | net|a,r,a |
|-----------|--------|--------------|-----|------|-----|-----|-----|-----|------|--------|--------|
| VGG       |        | blur         | 1.00| 0.845| 0.881| 0.917| 1.107| 0.881| 0.845| 0.881 |
|           |        | noise        | 1.00| 1.247| 1.186| 1.200| 0.691| 0.595| 0.530| 0.778 |
|           |        | digital      | 1.00| 0.917| 0.958| 1.000| 1.197| 1.087| 0.981| 0.955 |
|           |        | transformation| 1.00| 0.746| 0.762| 0.806| 1.150| 0.845| 0.779| 0.802 |
| SENet     |        | blur         | 1.036| 1.000| 1.238| 1.310| 1.036| 1.000| 1.238| 1.310 |
|           |        | noise        | 1.241| 1.381| 1.180| 1.040| 0.854| 1.017| 0.932| 0.938 |
|           |        | digital      | 1.023| 1.068| 1.155| 1.481| 1.023| 1.068| 1.155| 1.481 |
|           |        | transformation| 1.255| 1.094| 1.177| 1.728| 1.255| 1.094| 1.177| 1.728 |
| DenseNet  |        | blur         | 1.500| 1.226| 1.226| 1.345| 1.500| 1.226| 1.226| 1.345 |
|           |        | noise        | 0.577| 0.569| 0.589| 0.572| 0.865| 0.688| 0.714| 0.878 |
|           |        | digital      | 1.500| 1.330| 1.284| 1.417| 1.500| 1.330| 1.284| 1.417 |
|           |        | transformation| 1.598| 1.252| 1.248| 1.468| 1.598| 1.252| 1.248| 1.468 |
| Xception  |        | blur         | 1.810| 1.500| 1.738| 1.845| 1.810| 1.655| 1.571| 1.810 |
|           |        | noise        | 0.796| 1.687| 1.706| 1.767| 1.345| 0.992| 1.308| 1.224 |
|           |        | digital      | 1.674| 1.523| 1.477| 1.610| 1.777| 1.496| 1.432| 1.561 |
|           |        | transformation| 2.333| 1.699| 1.775| 1.968| 2.734| 1.868| 1.843| 2.098 |

Green color gradients indicate an improvement w.r.t. the baseline, while yellow and red color gradients indicate comparable or lower results than the baseline.

**Funding** Open access funding provided by Università degli Studi di Salerno within the CRUI-CARE Agreement. This research was partially supported by the Italian MIUR within PRIN 2017 grants, Projects Grant 20172BH297 002CUP D44I17000200005.

**Availability of data and material** The RAF-DB-C and the RAF-DB-P data sets are available at: https://github.com/MiviaLab/emotion-robustness. The original RAF-DB dataset is available from the website of the authors: http://www.whdeng.cn/RAF/model1.html.

**Code Availability** The code, the optimized augmentation policies and the models trained in this work are available at: https://github.com/MiviaLab/emotion-robustness

**Declarations**

**Conflict of Interests** The authors declare no conflicts of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.
References

1. Acharya D, Huang Z, Pani Paudel D, Van Gool L (2018) Covariance pooling for facial expression recognition. In: IEEE CVPR workshops, pp 367–374
2. Albanie S, Nagrani A, Vedaldi A, Zisserman A (2018) Emotion recognition in speech using cross-modal transfer in the wild, arXiv:1808.05561
3. Antipov G, Baccouche M, Berrani S-A, Dugelay J-L (2017) Effective training of convolutional neural networks for face-based gender and age prediction. Pattern Recogn 72:15–26
4. Barsoum E, Zhang C, Ferrer CC, Zhang Z (2016) Training deep networks for facial expression recognition with crowd-sourced label distribution. In: ACM ICM, pp 279–283
5. Cao Q, Shen L, Xie W, Parkhi OM, Zisserman A (2018) Vggface2: a dataset for recognising faces across pose and age. In: IEEE FG, pp 67–74
6. Carletti V, Greco A, Percannella G, Vento M (2019) Age from faces in the deep learning revolution. IEEE Trans Pattern Anal Mach Intell
7. Chollet F (2017) Xception: deep learning with depthwise separable convolutions. In: IEEE CVPR, pp 1251–1258
8. Corneanu CA, Simon MO, Cohn JF, Guerrero SE (2016) Survey on rgb, 3d, thermal, and multimodal approaches for facial expression recognition: history, trends, and affect-related applications. IEEE Trans Pattern Anal Mach Intell 38(8):1548–1568
9. Cubuk ED, Zoph B, Mane D, Vasudevan V, Le QV (2019) Autoaugment: learning augmentation strategies from data. In: IEEE CVPR, pp 113–123
10. Dhall A, Ramana Murthy O, Goecke R, Joshi J, Gedeon T (2015) Video and image based emotion recognition challenges in the wild: Emotiw 2015. In: ACM ICM, pp 423–426
11. Ding H, Zhou SK, Chellappa R. (2017) Facenet2expnet: regularizing a deep face recognition net for expression recognition. In: FG 2017, pp 118–126. IEEE
12. Ekman P, Sorenson ER, Friesen WV (1969) Pan-cultural elements in facial displays of emotion. Science 164(3875):86–88
13. Fan Y, Lam JC, Li VO (2018) Multi-region ensemble convolutional neural network for facial expression recognition. In: ICANN, pp 84–94
14. Farzaneh AH, Qi X (2021) Facial expression recognition in the wild via deep attentive center loss. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 2402–2411
15. Gad R, Talha M, Abd El-Latif AA, Zorkany M, Ayman E-S, Nawal E-F, Muhammad G (2018) Iris recognition using multi-algorithmic approaches for cognitive internet of things (ciot) framework. Futur Gener Comput Syst 89:178–191
16. Geirhos R, Temme CR, Rauber J, Schütt HH, Bethge M, Wichmann FA (2018) Generalisation in humans and deep neural networks. In: NeurIPS, pp 7538–7550
17. Goodfellow I, Erhan D, Carrier PL, Courville A, Mirza M, Hamner B, Cukierski W, Tang Y, Thaler D, Lee D-H et al (2013) Challenges in representation learning: a report on three machine learning contests. In: NeurIPS, pp 117–124
18. Goodfellow I, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples, arXiv:1412.6572
19. Greco A, Roberto A, Saggese A, Vento M, Vigilante V (2019) Emotion analysis from faces for social robotics. In: IEEE SMC, pp 358–364
20. Gunes H, Hung H (2016) Is automatic facial expression recognition of emotions coming to a dead end? The rise of the new kids on the block. Image Vis Comput 55:6–8
21. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: IEEE CVPR, pp 770–778
22. Hendrycks D, Dietterich T (2019) Benchmarking neural network robustness to common corruptions and perturbations. ICLR
23. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: IEEE CVPR, pp 7132–7141
24. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: IEEE CVPR, pp 4700–4708
25. Kim Y, Yoo B, Kwak Y, Choi C, Kim J (2017) Deep generative-contrastive networks for facial expression recognition, arXiv:1703.07140
26. Kollia D, Tzirakis P, Nicolaou MA, Papaioannou A, Zhao G, Schuller B, Kotsis I, Zafeiriou S (2019) Deep affect prediction in-the-wild: aff-wild database and challenge, deep architectures, and beyond. Int J Comput Vis 127(6–7):907–929
27. Kollia D, Cheng S, Ververas E, Kotsis I, Zafeiriou S (2020) Deep neural network augmentation: generating faces for affect analysis. Int J Comput Vis, 1–30
28. Li S, Deng W, Du J (2017) Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In: IEEE CVPR, pp 2584–2593
29. Li Y, Zeng J, Shan S, Chen X (2018) Occlusion aware facial expression recognition using cnn with attention mechanism. IEEE Trans Image Process 28(5):2439–2450
30. Lim S, Kim I, Kim T, Kim C, Kim S (2019) Fast autoaugment, arXiv:1905.00397
31. Ly TS, Do N-T, Kim S-H, Yang H-J, Lee G-S (2019) A novel 2d and 3d multimodal approach for in-the-wild facial expression recognition. Image Vis Comput 92:103817
32. Mollahosseini A, Hasani B, Mahoor MH (2017) Affectnet: a database for facial expression, valence, and arousal computing in the wild. IEEE Trans Affect Comput 10(1):18–31
33. Ojala T, Pietikainen M, Harwood D (1994) Performance evaluation of texture measures with classification based on kullback discrimination of distributions. IEEE ICPR 1:582–585
34. Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell 24(7):971–987
35. Parkhi OM, Vedaldi A, Zisserman A et al (2015) Deep face recognition. BMVC 1:6
36. Shan C, Gong S, McOwan PW (2009) Facial expression recognition based on local binary patterns: a comprehensive study. Image Vis Comput 27(6):803–816
37. Shi J, Zhu S, Liang Z (2021) Learning to amend facial expression representation via de-albino and affinity, arXiv:2103.10189
38. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556
39. Strisciuglio N, Lopez-Antequera M, Petkov N (2020) Enhanced robustness of convolutional networks with a push–pull inhibition layer. Neural Comput Appl, 1–15
40. Vasiljevic I, Chakrabarti A, Shakhnarovich G (2016) Examining the impact of blur on recognition by convolutional networks, arXiv:1611.05760
41. Vo T-H, Lee G-S, Yang H-J, Kim S-H (2020) Pyramid with super resolution for in-the-wild facial expression recognition. IEEE Access 8:131988–132001
42. Wang N, Li Q, Abd El-Latif AA, Yan X, Niu X (2013) A novel hybrid multibiometrics based on the fusion of dual iris, visible and thermal face images. In: 2013 International symposium on biometrics and security technologies. IEEE, pp 217–223
43. Wang N, Li Q, Abd El-Latif AA, Peng J, Niu X (2014) An enhanced thermal face recognition method based on multiscale complex fusion for gabor coefficients. Multimed Tools Appl 72(3):2339–2358
44. Wen Z, Lin W, Wang T, Xu G (2021) Distract your attention: multi-head cross attention network for facial expression recognition, arXiv:2109.07270
45. Yin D, Lopes RG, Shlens J, Cubuk ED, Gilmer J (2019) A fourier perspective on model robustness in computer vision, arXiv:1906.08988
46. Yu Z, Zhang C (2015) Image based static facial expression recognition with multiple deep network learning. In: ACM ICMI, pp 435–442
47. Zhang R (2019) Making convolutional networks shift-invariant again. ICML
48. Zheng S, Song Y, Leung T, Goodfellow I (2016) Improving the robustness of deep neural networks via stability training. In: IEEE CVPR, pp 4480–4488
49. Zhou H, Meng D, Zhang Y, Peng X, Du J, Wang K, Qiao Y (2019) Exploring emotion features and fusion strategies for audio-video emotion recognition. In: 2019 International conference on multimodal interaction, pp 562–566

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
Affiliations

Antonio Greco\textsuperscript{1} \(
\bullet \)
\textsuperscript{1} · Nicola Strisciuglio\textsuperscript{2} · Mario Vento\textsuperscript{1} · Vincenzo Vigilante\textsuperscript{1}

Nicola Strisciuglio
n.strisciuglio@utwente.nl

Mario Vento
mvento@unisa.it

Vincenzo Vigilante
info@vvigilante.com

\textsuperscript{1} Department of Computer Engineering, Electrical Engineering and Applied Mathematics (DIEM), University of Salerno, Fisciano, Italy

\textsuperscript{2} Faculty of Electrical Engineering, Mathematics & Computer Science, University of Twente, Enschede, The Netherlands