Py-Feat: Python Facial Expression Analysis Toolbox

Jin Hyun Cheong[1], Tiankang Xie[1,2], Sophie Byrne[1], & Luke J. Chang[1]

[1]Computational Social and Affective Neuroscience Laboratory
Department of Psychological & Brain Sciences
Dartmouth College
Hanover, NH 03755

[2] Program in Quantitative Biomedical Sciences
Geisel School of Medicine
Dartmouth College
Hanover, NH 03755

Studying facial expressions is a notoriously difficult endeavor. Recent advances in the field of affective computing have yielded impressive progress in automatically detecting facial expressions from pictures and videos. However, much of this work has yet to be widely disseminated in social science domains such as psychology. Current state of the art models require considerable domain expertise that is not traditionally incorporated into social science training programs. Furthermore, there is a notable absence of user-friendly and open-source software that provides a comprehensive set of tools and functions that support facial expression research. In this paper, we introduce Py-Feat, an open-source Python toolbox that provides support for detecting, preprocessing, analyzing, and visualizing facial expression data. Py-Feat makes it easy for domain experts to disseminate and benchmark computer vision models and also for end users to quickly process, analyze, and visualize face expression data. We hope this platform will facilitate increased use of facial expression data in human behavior research.

Introduction

Facial expressions can reveal insights into an individual's internal mental state and provide nonverbal channels to aid in interpersonal and cross-species communication [1,2]. One of the main challenges to studying facial expressions has been arriving at a consensus understanding as to how to best represent and objectively measure expressions. The Facial Affect Coding System (FACS) [3] is one of the most popular systems to reliably quantify the intensity of groups of facial muscles referred to as action units (AUs). However, extracting facial expression information using FACS coding can be a laborious and time-intensive process. Becoming a certified FACS coder not only requires 100 hours of training, but a trained coder may require an hour to code a single minute of video [4] and is not without cultural biases and errors [5,6]. Facial electromyography (EMG) provides one method to objectively record from a finite number of facial muscles at a high temporal resolution [7,8], but requires specialized recording equipment that restricts data collection to the laboratory and can visually obscure the face making it less ideal for social contexts.

Automated methods using techniques from computer vision have emerged as a promising approach to extract representations of facial expressions from pictures, videos, and depth cameras both inside and outside the laboratory. Participants can be untethered from cumbersome wires and can naturally engage in tasks such as watching a movie or having a conversation [9–13]. In addition to AUs, computer vision techniques have provided alternative spaces to represent facial expressions such as facial landmarks [14] or lower dimensional latent representations [15]. These tools have a number of applications relevant to psychology such as predicting the intensity of emotions [16–19] and other affective states such as pain [20,21], distinguishing between genuine and fake expressions [22], detecting signs of depression [23], inferring traits such as personality [24–26] or political orientations [27], and predicting the development of interpersonal relationships [11,13]. Though facial expression research has seen rapid growth in affective computing facilitated by recent advances in machine learning, adoption in fields outside the domain of computer science such as psychology has been surprisingly slow.

In our view, there are at least two specific barriers contributing to the slow adoption of automated methods in social science fields such as psychology. First, there is a relatively high barrier to entry to training and accessing state of the art models capable of quantifying facial expressions. This requires knowledge of computer vision techniques, neural network architectures, and access to large labeled datasets and computational infrastructure that include Graphics Processing Units (GPUs). Though there are heroic efforts to share high quality datasets [28–34], there are still difficulties sharing this data involving participants' privacy, complicated end user agreements, expensive handling fees, contacting data curators, and finding affordable and stable long-term hosting solutions. Though hundreds of models have been developed to characterize facial expressions, no standards have emerged for disseminating these models to end users. These models are typically reported in conference proceedings, occasionally shared on open code repositories such as Github, and require considerable domain knowledge as they have been developed using a multitude of computer languages, rarely have documentation, and occasionally have restrictive licensing. Each model may require the data to be preprocessed in a specific way or rely on additional features (e.g., landmarks, predefined regions of interest). Because there are currently no generally agreed upon standards for training and benchmarking, each model is usually trained on different datasets, which makes it difficult to benchmark the models using the same dataset to aid in the model selection process [16,35]. Platforms such as paperswithcode.com are helping to standardize the dissemination and benchmarking of models, but sharing state of the art models has not yet become a norm in the field. Other domains such as natural language processing and reinforcement learning have begun to overcome this issue with a variety of high quality platforms such as Stanza [36], SpaCy, and OpenAI Gym [37].

Second, there is a notable lack of free open-source software to aid in detecting, preprocessing, analyzing, and visualizing facial expressions (see Table 1 for software comparison). Commercial software options such as Affdex (Affectiva Inc) available through iMotions [38] and Noldus FaceReader [39] can be expensive, have limited functionality, and typically do not employ state of the art models [40–42] (see Stöckli et al 2018 [16] and Dupré et al 2020 [19] for commercial software performance comparisons). Furthermore, due to strong interest from industry, there have been several free software packages such as the Computer Expression Recognition Toolbox [43], Intraface [14], and Affectiva API [44] (Affectiva Inc) that have turned into commercial products or been acquired by larger technology companies such as Apple Inc or Facebook and rendered unavailable to researchers. Currently, OpenFace [45] is the most widely used open-source software that allows users to extract facial landmarks and action units from face images and videos. However, OpenFace does not provide a full suite of tools for preprocessing, analyzing, and visualizing data, which would make these tools more accessible to non-domain experts. As an example, in other fields such as neuroscience, the rapid growth of neuroimaging research has been facilitated by the widespread use of free tools such as FSL [46], AFNI [47], SPM [48], and NiLearn [49] that enables end users to preprocess, analyze, and visualize complex brain imaging data. We believe the broader emotion research community would greatly benefit from additional software platforms dedicated to facial expression analysis with functions for extracting, preprocessing, analyzing, and visualizing facial expression data.

	Facial feature detection					Preprocessing	Analysis	Free
	Facial landmarks	Action units	Emotions	Headpose	Gaze			
iMotions*						X	X	
FACET	X	X	X	X				
AFFDEX	X	X	X	X				
Noldus FaceReader		X**	X			X	X**	
OpenFace	X	X		X	X			X
face-api.js	X		X					X
Py-Feat	[X]	[X]	[X]			[X]	[X]	[X]

Table 1. Software comparison on functionalities and affordability. *X indicates features provided by each package. Features from Py-Feat toolbox are shown in brackets. Facial landmarks are points pertaining to locations of key spatial positions of the face including the jaw, mouth, nose, eyes, and eyebrows. Action units are facial muscle groups defined by FACS [50]. Emotions refer to the detection of canonical emotional expressions. Headpose refers to the pitch, roll, and yaw orientations of the face. Gaze refers to the direction the eyes are looking. *iMotions is a*

*platform and it's feature extraction relies on the purchase of either the AFFDEX or FACET modules. **Detection of action units and analysis functionalities require a separate add-on purchase of The Action Unit Module and the Project Analysis Module for the Noldus FaceReader.*

To meet this need, we have created the Python Facial Expression Analysis Toolbox (Py-Feat) which is a free, open-source package dedicated to support the analysis of facial expression data. It provides tools to extract facial features like OpenFace [45], but additionally provides modules for preprocessing, analyzing, and visualizing facial expression data (see pipeline in Figure 1). Py-Feat is designed to meet the needs of two distinct types of users. Py-Feat benefits computer vision researchers who can use our platform to disseminate their state of the art models to a broader audience and easily compare their models with others. It also benefits social science researchers looking for free and easy to use tools that can both detect and analyze facial expressions. In this paper, we outline the key components of the Py-Feat toolbox including detailed descriptions of the facial feature detection models and analysis tools.

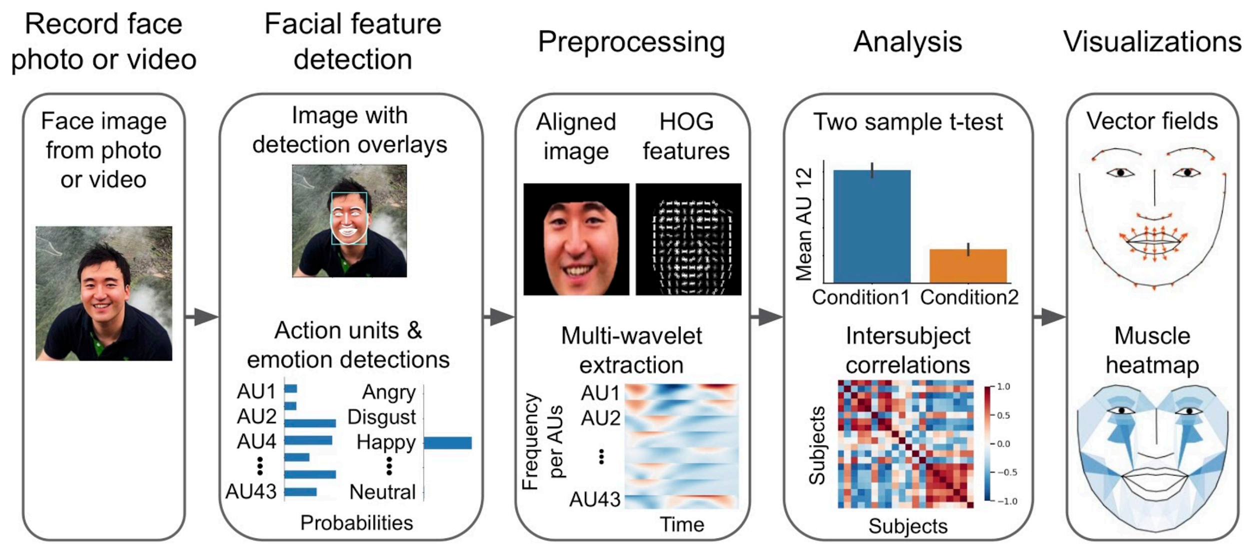

Figure 1. Facial expressions analysis pipeline. *Analysis of facial expressions begins with recording face photos or videos using a recording device such as webcams, camcorders, head mounted cameras, or 360 cameras. After capturing the face, researchers can use Py-Feat to detect facial features such as facial landmarks, action units, and emotions, and check the detection results with image overlays and bar graphs. The detection results can be preprocessed by extracting additional features such as Histogram of Oriented Gradients or multi-wavelet decomposition. Resulting data can then be analyzed within the toolbox using statistical methods such as t-tests, regressions, and intersubject correlations. Visualization functions can generate face images from models of action unit activations to show vector fields depicting landmark movements and heatmaps of facial muscle activations.*

Py-Feat Modules

In this section, we outline the key modules of Py-Feat and how they were designed, developed, and evaluated. Py-Feat includes a Detector module for detecting facial expression features (i.e., detecting faces, facial landmarks, AU activations, emotional expressions) from face photos and videos and a Fex data class that includes methods for preprocessing, analyzing, and visualizing facial expression data.

Detector Module

The Detector module offers several models for detecting each of the following facial features: (a) finding a face in an image or video frame, (b) locating facial landmarks, (c) detecting activations of facial muscle action units, and (d) detecting displays of canonical emotional expressions. These models are designed to be modular so users can decide which algorithms to use for each detection task based on their needs for accuracy and speed. Here we provide an overview of the models currently available in the Detector module and how they were benchmarked and selected to be included with Py-Feat. In general, we considered models with high reported accuracy, written in Python, easy to install (e.g., Pytorch [51] for neural network models and scikit-learn [52] for statistical models), and open to use for academic research.

Face detectors

One of the most basic steps in the facial feature detection process is to identify if there is a face in the image and where that face is located. Py-Feat includes three popular face detectors including Faceboxes [53], Multi-task Convolutional Neural Network (MTCNN) [54,55], and RetinaFace [56]. These detectors are widely used in other open-source software [45] and are known to achieve fast and accurate face detection results even for partially occluded or non-frontal faces. Face detection results are reported as a bounding box of the face including confidence scores that can also be used to initialize the facial landmark detector. We benchmarked the face detection models on the validation set of the WIDER FACE dataset [57]. The validation scores were calculated as an average precision score derived from detection accuracy using a Jaccard similarity over the ground truth and predicted bounding boxes implemented by WIDER FACE. Average precision scores of other models benchmarked on the same dataset are included for comparison [57].

Facial landmark detectors

Facial landmarks are points identified in the image space outlining the jaw, mouth, nose, eyes, and eyebrows of a face. The distance and angular relationships between the landmarks can be used to infer emotional states such as pain [20]. The 68 facial landmark scheme is used widely across datasets and software [45,58,59] and was chosen as the target output for our facial landmark detectors. Py-Feat offers three facial landmark detectors including the Practical Facial Landmark Detector (PFLD) [60], MobileNets [61], and MobileFaceNets [62] algorithms. These algorithms are designed to be compact (less than 50MB) and are able to detect facial landmarks quickly on a variety of platforms including mobile devices while maintaining high accuracy. Our implementation of facial landmark detectors receives the face bounding box from the face detector as inputs to estimate the x and y coordinates of 68 facial landmarks. We benchmarked these models on the 300 Faces in the Wild (300W) dataset [59,63] and compared the results with other models reported at paperswithcode.com. The accuracy was calculated as the average euclidean distance between the predicted and ground truth landmark coordinates normalized by the interocular distance.

Action unit detectors

Action units (AUs) are the building blocks of facial expressions where each AU number corresponds to a specific facial muscle movement [3]. The combination of action units can indicate an emotional expression. For example, the activations of AUs 6 (cheek raiser) and 12 (lip corner puller) comprise a display of a happy or joyful face, while activations of AUs 1 (inner brow raiser), 4 (brow lowerer), and 15 (lip corner depressor) create a sad facial expression. AUs can also be used as features for characterizing affective states such as contempt [64], confusion [65], and guilt [66], or for predicting the facial expressions associated with different experimental conditions. We include several different types of models using deep learning and statistical learning approaches.

We adapted and trained the JÂA-Net [67,68] neural network model to predict facial landmarks and action units. JÂA-Net is an end-to-end deep learning model that employs an adaptive attention mechanism, in which it uses facial landmarks to help localize areas within an image to perform action unit detection. JÂA-Net refines its attention maps for individual AUs with an adaptive regional attention layer. In the final layers, JÂA-Net combines all of the global, local attention, and landmark alignment features for AU detection [68]. In the original paper, JÂA-Net reported an impressive performance in accurately detecting AUs (average F1=78.4). We include JÂA-Net in Py-Feat after training the model using the BP4D (Binghamton-Pittsburgh 3D Dynamic Spontaneous Facial expression Database) [31] and BP4D+ [69] datasets.

Py-Feat provides three additional AU detectors which were trained on the BP4D [31], DISFA [30], CK+ [29], Shoulder Pain [70] and AFF-Wild2 [71–76] datasets using statistical learning algorithms, specifically a Random Forest classifier (Feat-RF), a linear Support Vector Machines classifier (Feat-SVM), and a logistic regression classifier (Feat-Logistic). The training of these algorithms closely followed the steps outlined in Baltrusaitis et al. (2015) [77] which used facial landmarks and Histogram of Oriented Gradients (HOGs) as features in predicting action unit activations. HOGs are feature descriptors that describe an image as a distribution of orientations such as edges and corners measured across the image and have been proven effective in identifying people in images as well as action units [77,78]. We first preprocessed each image by aligning the detected faces using the interocular distance to a neutral facial expression. We then detected the facial landmarks for the aligned faces and applied a convex hull to mask out the background irrelevant to the face. To include facial features of the forehead, a convex hull was applied with the eyebrows shifted upwards 1.5 times the distance between the eyebrows and the upper eye landmarks. We extracted HOGs using the scikit-image implementation [79] with 8 orientations, 8x8 pixels per cell, and 2x2 cells per block which led to a total of 5,408 HOG features. We then applied a principal component analysis (PCA) to retain 95% of the variance, which reduced the dimensionality of these features down to 1,195. The aligned facial landmarks and the HOGs were used to train a Random Forest model, Linear SVM model, and Logistic Regression model using scikit-learn [80] which were selected to compare the performance of modeling linear and non-linear relationships. Hyperparameters were tuned with a grid search during training using 3-fold cross validation. Additional details for the training procedure and parameters used are available in our package and online tutorials.

To evaluate and compare the performance of these models, we benchmarked our implementation of the JÂA-Net model and the three statistical learning models (Random Forest, SVM, & Logistic Regression) against the previously available FACET model in iMotions, and the OpenFace [45] software on the Extended DISFA Plus dataset [32] which was not included in training any of these models. We evaluated model performance across twelve AUs: AU1 (inner brow raiser), AU2 (outer brow raiser), AU4 (brow lowerer), AU5 (upper lid raiser), AU6 (cheek raiser), AU9 (nose wrinkler), AU12 (lip corner puller), AU15 (lip corner depressor), AU17 (chin raiser), AU20 (lip stretcher), AU25 (lips part), and AU26 (jaw drop) using F1 scores. F1 is an accuracy metric for binary classification, defined as:

$$F1 = 2 * \left(\frac{precision * recall}{precision + recall}\right) \quad \text{(eq1)}$$

where precision is the number of true positives divided by the total number of positive results:

$$precision = \frac{true\ positive}{true\ positive + false\ positive} \quad \text{(eq2)}$$

and recall is the proportion of true positives relative to the ground truth:

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \quad \text{(eq3)}$$

F1 scores range from 0 to a perfect precision and recall of 1.0. We report the F1 scores for each AU and the average F1 score across all AUs.

Emotion detectors

Emotion detectors are trained on manually posed or naturalistically elicited emotional facial expressions which allows detectors to classify new images based on how much a face resembles a canonical emotional facial expression. It is important to note that detecting a smiling face as happy does not necessarily imply that the individual is experiencing an internal subjective state of happiness [81]. However, labeling specific configurations of AUs with the semantic concepts of emotions can still be useful in emotion research to characterize the contexts in which people tend to display these facial expressions or how the display of certain emotion expressions accompanies changes in learning [82] and social behaviors [13].

We provide four emotion detectors capable of detecting seven categories of emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. First, we include the Residual Masking Network (ResMaskNet) [83] which is an end-to-end convolutional neural network model that combines deep residual networks with masking blocks. The masking blocks help focus the model's attention on local regions of interest to refine its feature map for more fine-grained predictions and the residual structure helps to maintain performances in deeper layers. ResMaskNet achieved state of the art performance on the facial expression recognition (FER)

2013 [84] dataset at the time of preparing this article. Despite its accuracy, ResMaskNet has a large memory footprint (500MB) due to the depth of the architecture.

We also provide FerNet, a more compact neural network model (80MB) that builds on multilayer convolutional neural networks (CNN). We augment our CNN layers model by adding a multi-scale attention layer before the CNN layers. Similar to JÂA-Net, this entails adding an additional regional learning layer, which divides the input image into different sized patches (88x88, 44x44 and 22x22 pixels) and concatenates the patches to extract more local texture information [67,68] from the input image. This model was trained using the ExpW [85], CK+ [29] and JAFFE [86] datasets.

Lastly, we provide two statistical learning models to perform emotion expression recognition (i.e., Random Forest, Linear SVM). These models are identical to our statistical learning AU models in that they perform face alignment, apply a convex hull, and extract HOG features, except that they detect emotions rather than AUs. Both models were trained using the scikit-learn [52] implementation on the ExpW [85], CK+ [29] and JAFFE [86] facial expressions datasets with a 3-fold cross validation for identifying the best hyperparameters.

We benchmarked ResMaskNet, FerNet, and the two statistical learning models against the previously available FACET-iMotions [38] using the AffectNet dataset [87] which includes one million images depicting facial expressions collected on the web (i.e., in the wild) allowing us to test the generalizability of the models. We calculated the F1 scores for each emotion category (i.e., angry, disgust, fear, happy, sad, surprise, neutral, average) and the average F1 score across categories.

Fex Module

The Fex data class offers key functionalities to facilitate facial expression data analysis. The Fex class is an extension of the widely used Pandas DataFrame [88] allowing users to leverage the existing functionalities provided by Pandas including slicing, grouping, sampling, and summarizing data. The key contribution of the Fex class is that it offers specialized functions to aid in manipulating facial expressions data such as selecting different types of data (i.e., faceboxes, landmarks, action units, emotions), preprocessing facial expression time series data, extracting additional features from time series data, analyzing aggregates of facial expressions data, and visualizing intermediary preprocessing steps. Here we describe the key functionalities for preprocessing, analyzing, and visualizing Fex data.

Preprocessing

Preprocessing is an important aspect of facial expression data analysis and includes steps such as cleaning or standardizing the data as well as extracting additional features. For example, individuals may express different neutral facial expressions at rest where some individuals might lean towards a smiling resting face while others might lean towards a frowning resting face. To adjust for these individual differences, software such as iMotions recommend researchers to record a baseline facial expression which can be subtracted from facial expressions during the experiment. Alternatively, subtracting the median facial expression can also be an effective normalization step because neutral facial expressions tend to dominate most interactions [89] and have been found to increase classification accuracy of action unit detections [77]. Both of these approaches are available with the Fex class.

When face videos are recorded over time, researchers may need to summarize the time series data of facial expressions separately by subjects, trials, or experimental conditions. We allow researchers to specify these groupings in the sessions attribute of the Fex class. For example, for experiments with structured trials designed to elicit evoked responses such as experiencing acute pain, it can be useful to summarize and describe facial expression signals within a trial using the peak, minimum, or average values per recordings [20,21]. Other times, it may not be known exactly when to expect a discrete evoked response, and instead it may be more useful to describe signals in the frequency domain (Figure 1). This might involve conducting time-frequency analysis by performing frequency decomposition on a time series using discrete wavelets, or computing the number of times that a signal within a specific frequency band exceeds a preset threshold. For example, the bag of temporal filters approach provides sensitivity to the temporal and intensity profiles of a signal while maintaining invariance to when those peaks occur and has been found to be effective in differentiating genuine from posed pain facial expressions [22]. All of these feature extraction techniques are available in the Fex module.

Analysis

Py-Feat provides several basic analysis functions including t-tests, regression, prediction, and intersubject correlations. Py-Feat's t-test functions allow users to conveniently select which experimental conditions and facial features to test by handling the data reshaping for the user before conducting the test with implementations from scipy [90]. The predict method uses the scikit-learn API [80], and can thus be used with any scikit-learn prediction or classification model. This approach can be used to identify a combination of AUs or emotional facial expressions that predict an experimental condition. On the other hand, researchers who aim to explain the variability of action units can use the regress function by passing a design matrix which is implemented via nltools [91]. Lastly, users can calculate an intersubject similarity matrix across time or across features to identify a common structure across individuals or between conditions. Beyond these basic analyses, the representation of the Fex data class as an extension of a Pandas DataFrame allows users to easily incorporate additional analytic tools from other packages in the Python scientific computing ecosystem.

Visualizations

We provide several plotting methods to help visualize the FEX data in each stage of the analysis pipeline. In the facial feature detection stage, we offer the plot_detections function that overlays the face, facial landmarks, action units, and emotion detection results in a single figure (Figure 1). This function can be used to validate the detection results at each video frame or image. Fex class automatically inherits the plotting functionalities of a Pandas DataFrame and thus allows users to plot time series graphs as well. This function can be useful for examining how detected action unit activities vary over time or if there are segments of missing data.

We also provide a visualization model which can be used to visualize how combinations of activated AUs will look like on a stylized anonymous face. We trained this action unit to landmark model on 20 action units (AUs 1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 15, 17, 18, 20, 23, 24, 25, 26, 28, 43) with a subset of images from the EmotioNet [92], BP4D[31], and Extended DISFA Plus [32] datasets to balance the representation of each AU. We used our toolbox with the Feat-RetinaFace face detector and MobileNets landmark detector to detect the landmarks on these images. We aligned these landmarks to a neutral face with an affine transformation using the facial landmarks and fit a Partial Least Squares Regression model with 20 components to predict these aligned landmarks from the ground truth action unit labels provided by the datasets. Using this model, users can visualize the action units and their accompanying 2D landmark deformation on a standard face from any combination of action unit activations identified from their analyses.

Experimental evaluation results

Facial feature detection benchmarking results

Face detection

We benchmarked the face detection algorithms implemented in Py-Feat (Feat-Faceboxes, Feat-MTCNN, Feat-RetinaFace) on the WIDER face dataset [57] and compared these results with the benchmarking results of other models (TinaFace, RetinaFace, ACF-WIDER) reported by the WIDER face dataset. Benchmarking results are summarized in Table 2. The Py-Feat implementation of Faceboxes, MTCNN, and RetinaFaces achieved acceptable average precision in the *Easy* subset of the WIDER Face dataset with Feat-RetinaFace reaching within 10% of the accuracy of TinaFace [93], which was the state of the art model at the time of preparing this article. Although we attempted to install and implement TinaFace which is freely available (https://github.com/Media-Smart/vedadet), we were unable to successfully implement the model due to complicated installation procedures. Feat-RetinaFace did not reach the accuracy stated in the original RetinaFace paper [56] potentially due to hyperparameter or training set differences. We also observed a degraded performance on the *Hard* subset of WIDER faces which include small, inverted, and highly occluded faces. Nevertheless, we believe that most researchers will be recording faces at close proximity which would most likely be classified as an easy detection task. Based on these results, we have set RetinaFace as our default face detection model.

	Model	Easy	Medium	Hard
Models not available on Py-Feat	TinaFace	**.97**	**.96**	**.93**
	RetinaFace	.97	.96	.92
	ACF-WIDER	.66	.54	.27
Models available on Py-Feat	Feat-Faceboxes	.78	.68	.31
	Feat-MTCNN	.54	.50	.28
	Feat-RetinaFace (default)	[.89]	[.85]	[.61]

Table 2. Benchmarking results for face bounding box detection. *Easy, Medium, Hard results retrieved from WIDER Face. Numbers are average precision scores with higher numbers indicating better detection accuracy. Bold numbers indicate best performance for each column and bracketed numbers indicate the performance of the model selected as the default for Py-Feat.*

Landmark detection

Benchmarking results for landmark detection are summarized in Table 3 as the average root mean squared error between the predicted and ground truth coordinates across the 68 landmark points normalized by the interocular distance. While the models we adapted performed better than some models such as Pose-Invariant [94] or 3D Dense Face Alignment (3DDFA) [95], it was not as accurate as the more recent regression tree models such as 3D Deeply-initialized Ensemble (3DDE) [96] or DCFE (Deeply-initialized Coarse to Fine Ensemble) [97] models. Unfortunately, these models were written in languages not yet compatible with our toolbox (e.g., C++ and Tensorflow). Based on these results, we selected the MobileNet model as the default landmark detection model.

	Model	300W
Models not available on Py-Feat	3DDE	**3.13**
	DCFE	3.24
	Pose-Invariant	6.30
	3DDFA	7.01
Models available on Py-Feat	Feat-MobileNet (default)	[5.23]
	Feat-MobileFaceNet	6.00
	Feat-PFLD	6.41

Table 3. Benchmarking results for face landmark detection. *Feat models were initialized with face bounding boxes using RetinaFace. Numbers are root mean squared errors of coordinates with lower numbers indicating better alignment. Bold bracketed numbers indicate best performance for each column and bracketed numbers indicate the performance of the model selected as the default for Py-Feat.*

Action unit detection

Action unit (AU) detection benchmarking was conducted on the Extended DISFA Plus dataset [32] using F1 scores for each of the twelve AUs. Benchmarking results are summarized in Table 4. The previously available FACET-iMotions achieved the best overall accuracy and was the best detector for AUs 2, 4, 9, 15, and 17. OpenFace and our Feat-SVM model achieved the second highest average F1 scores followed by the Feat-RF model. OpenFace was the most accurate in detecting AUs 1, 6, and 12, while our implementations were the best at detecting AUs 5, 20, 25, and 26. Surprisingly, our implementation of a neural network model JÂA-Net performed the worst compared to the statistical learning algorithms. This also suggests that the generalizability of the model may be questionable given that it requires precise alignment of landmarks to determine the attention regions. Based on these results and for the ease of providing a detection probability which can be used for further analyses, we set the Random Forest model as our default action unit detection model.

	Model	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	Average
Models not available on Py-Feat	FACET iMotions	.58	**.62**	**.74**	.56	.78	**.73**	.77	**.59**	**.47**	.15	.64	.43	**.59**
	OpenFace	**.71**	.53	.69	.49	**.81**	.54	**.83**	.34	.43	.14	.72	.67	.57
Models available on Py-Feat	Feat-RF (default)	[.61]	[.61]	[.59]	**[.58]**	[.65]	[.39]	[.72]	[.43]	[.38]	[.25]	[**.84**]	[**.70**]	[.56]
	Feat-SVM	.54	.49	.68	.51	.75	.54	.74	.32	.35	**.43**	.83	.62	.57
	Feat-Logistic	.49	.45	.61	.56	.70	.61	.62	.26	.32	.29	.79	.52	.52
	Feat-JÂA-Net	.31	.22	.15		.29		.30	.08	.18				.22

Table 4. Benchmarking results for AU models on DisfaPlus. *Numbers shown are F1 scores. Bold bracketed numbers indicate best performance for each column and bracketed numbers indicate the performance of the model selected as the default for Py-Feat.*

Emotion detection

Benchmarking of the emotion detection models was conducted on the AffectNet dataset [87] which contains unposed expressions of emotions as they naturally occur in the wild outside of a carefully curated laboratory environment. We selected a random subset of 500 images for each of the seven different emotions for benchmarking and calculated the F1 score for each emotion category. The Residual Masking Network model [83] achieved the highest F1 score, followed by the FACET-iMotions model and the statistical learning models trained on HOG features. Based on these results we set the Residual Masking Network model as our default emotion detection model

	Model	angry	disgust	fear	happy	sad	surprise	neutral	average
Models not available on Py-Feat	FACET iMotions	.33	.42	.35	.67	.24	.36	.43	.40
Models available on Py-Feat	Residual Masking Network (default)	**[.53]**	**[.53]**	**[.48]**	**[.77]**	**[.54]**	**[.55]**	**[.49]**	**[.55]**
	Feat-FerNet	.22	.06	.15	.67	.35	.34	.38	.31
	Feat-SVM	.39	.27	.37	.60	.33	.39	.33	.38
	Feat-RF	.37	.21	.37	.62	.30	.35	.33	.36

Table 5. Benchmarking results for motion models on AffectNet. *Numbers shown are F1 scores. Bold bracketed numbers indicate best performance for each column and bracketed numbers indicate the performance of the model selected as the default for Py-Feat.*

Evaluation of preprocessing, analysis, and visualizations

To validate our preprocessing and analysis methods, we analyzed a previously published dataset exploring how natural facial expressions vary while delivering good and bad news [98]. In this study, Watson and colleagues (2020) recorded short clips of facial expressions while three participants delivered 10 types of good news (e.g., “your application has been accepted”) and 10 types of bad news (e.g., “your application was denied”). When participants delivered good news, AU 12 (lip corner puller) and 17 (chin raiser) were more consistently active while AU 1 (inner brow raiser) was more active when participants delivered bad news. We analyzed a sub-sample of the dataset (10 good news clips and 10 bad news clips from a single subject) and expected to find similar effects in AUs 1, 12, and 17, using an independent t-test, regression, and prediction analyses.

We used our Feat-RF model to detect action units from the videos of the good news and bad news delivering dataset [98] and preprocessed the data by extracting the mean probability of AU activation for each video clip. We compared the mean activation probabilities of AUs 1, 12, and 17 between the good news and bad news conditions with an independent-samples t-test. We replicated the original finding that AUs 12, t(18)=17, p<.001, and 17, t(18)=4.4, p<.001, were significantly more active when the participant delivered good news (Figure 2a). However, in contrast to the original paper, the average difference between the mean activation of AU 1 trended significance, t(18)=1.8, p=.08, towards more activation when delivering good news.

We also trained a Logistic Regression classifier to discriminate the good news clips from the bad news clips using the mean activation of AUs in a leave-one-clip-out cross-validation scheme as done in the original paper [98]. We achieved a perfect accuracy score in this subsample of the data, which was not too surprising given that the original paper also achieved a near perfect classification accuracy of .97 for predicting good news clips and .99 for bad news clips across a larger number of clips repeating the same messages. To identify which action units contributed to the classification, we fit a Logistic Regression classifier with all clips and inspected the regression coefficients. Consistent with the original findings, greater activations of AU 12 and AU 17 increased the likelihood of the clip being classified as a good news clip, while greater activation of AU 1 increased the likelihood of the clip being classified as a bad news clip (Figure 2b). To visualize the model, we used the regression coefficients to create a facial representation of delivering good news and bad news using our landmark visualization model (Figure 2c).

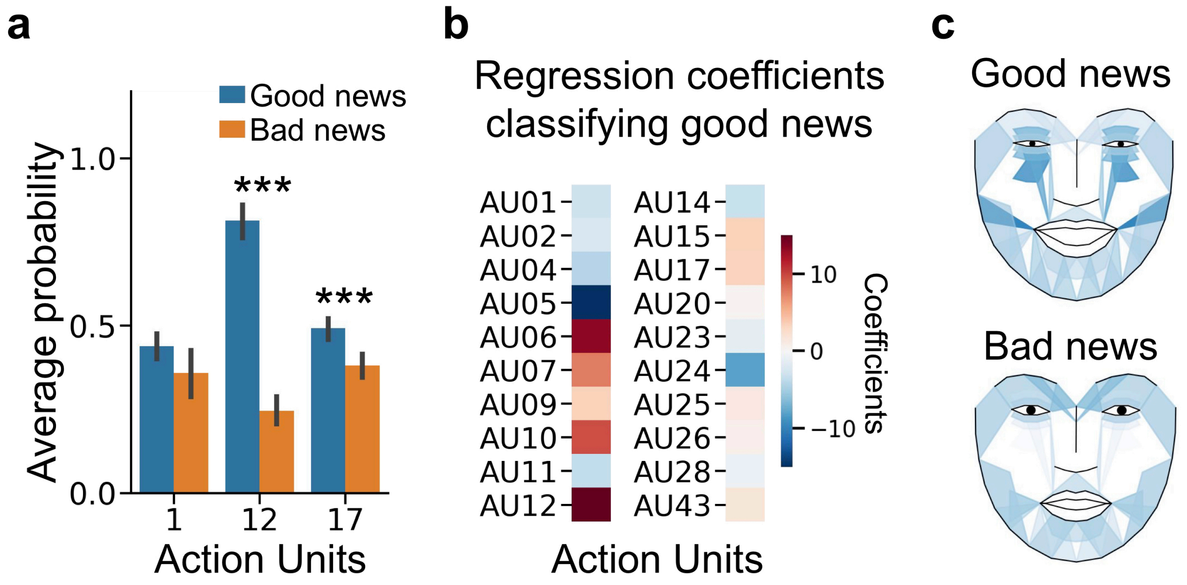

Figure 2. Conceptual replication results. *a) Average probability of action unit (AU) activation differences when delivering good news and bad news for AUs 12 and 17. *p<.05, **p<.01, ***p<.001. b) Regression coefficients on each AU from a logistic regression classifier for classifying good news and bad news clips. Activation of action units with positive coefficients (colored red) increases the probability of a clip being classified as a good news clip. Activation of action units with negative coefficients (colored blue) increases the likelihood of a clip to be classified as a bad news clip. c) Facial expressions predicting the delivery of good news and bad news generated using the regression coefficients from the good news and bad news classifier.*

The landmark visualization model trained with a PLS Regression achieved an r^2 of 0.15 on 10,000 sample images. To demonstrate face validity, we visualized the model fit by activating AUs 1 (inner brow raiser), 12 (lip corner puller), 15 (lip corner depressor), and 43 (eye closer) independently as shown in Figure 3a which produced the expected facial expressions for each muscle movement. The visualizations clearly indicated the movement of landmarks and muscles relating to the AUs that were activated. We also visualized emotional facial expressions based on detecting happy, sad, surprise, and anger expressions from single images representing each emotion label from the CK+ [29] using the Residual Masking Network implemented in Py-Feat (Figure 3b).

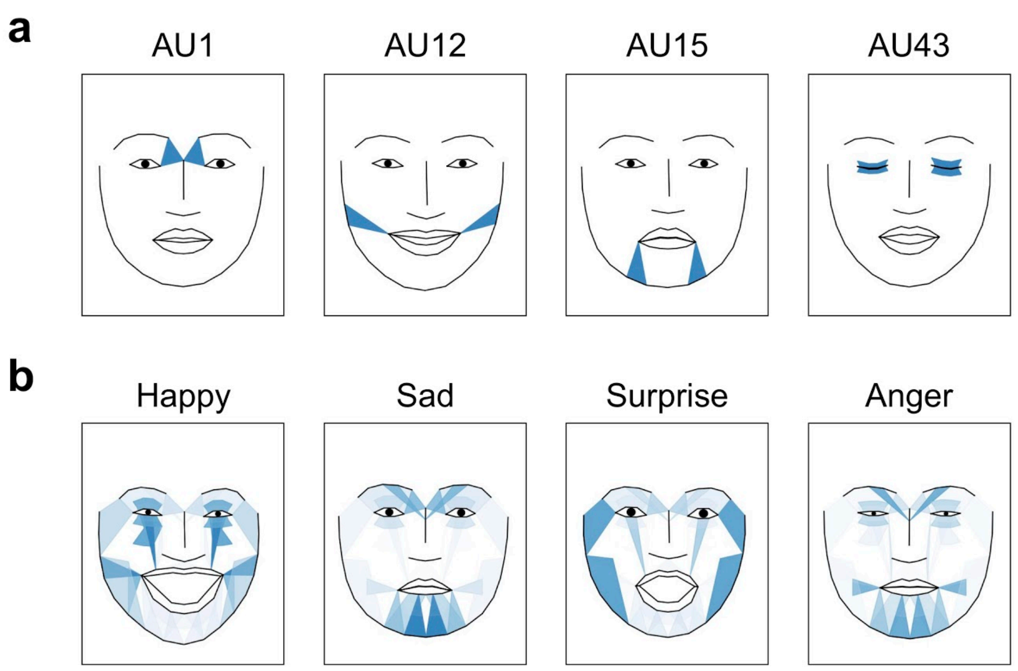

Figure 3. Demonstration of action unit to landmark visualization. *a) Facial expressions generated using our visualization model independently activating AUs 1 (inner brow raiser), 12 (lip corner puller), 15 (lip corner depressor), and 43 (eye closer). b) Facial expressions generated with a combination of AU activations extracted from images labeled as displaying each type of emotion including happy, sad, surprise, and anger.*

Discussion

In this paper, we describe the motivation, design principles, and core functionality of the open-source Python package Py-Feat. This package aims to bridge the gap between model developers creating new algorithms for detecting faces, facial landmarks, action units, and emotions with end users hoping to use these cutting edge models in their research. To achieve this, we designed an easy to use and open-source Python toolbox that allows researchers to quickly detect facial expressions from face images and videos and subsequently preprocess, analyze, and visualize the results. We hope this project will make facial expression analysis more accessible to researchers who may not have sufficient domain knowledge to implement these techniques themselves. In addition, Py-Feat provides a platform for model developers to disseminate their models to end-user researchers and compare the performance of their model with others included in the toolbox.

Automated detection of facial expressions has the potential to complement other techniques such as psychophysiology, brain imaging, and self-report [13,21,99] along with 3-D simulations [100] in improving our understanding of how emotions interact with perception, cognition, and social interactions and are impacted by our physical and mental health. Studying facial expressions is becoming increasingly more accessible to non-specialists. For example, recording participants has become more convenient with a number of affordable recording options such as webcams that can be used to record remote participants, open-source head mounted cameras allowing reliable face recordings in social settings [12], as well as 360 cameras that can be used to record multiple individuals simultaneously. The primary goal of Py-Feat is to make the preprocessing, analysis, and visualization of these results similarly accessible and free of charge to non-specialists. Open source software focused on the full analysis pipeline has been instrumental in contributing to the rapid progress of research in other domains such as neuroimaging with FSL [46], AFNI [47], SPM [48], and NiLearn [49] and natural language processing with Stanza [36] and SpaCy. We believe the broader emotion research community would greatly benefit from additional software platforms dedicated to facial expression analysis with functions for extracting, preprocessing, analyzing, and visualizing facial expression data.

Our toolbox is designed to be flexible and dynamic and includes models that are performing near state of the art. However, there are several limitations that are important to note. First, our current implementations of some of the models are not performing as well as the original versions. This could be attributed to nuances in hyperparameter optimization, variations in random seeds, and variations in the benchmarking datasets. We anticipate that these models will improve over time as more datasets become available and also plan to continually incorporate new models as they become available. Benchmarking of new models will be added to a living document on our project website to allow users to make informed choices in selecting models. Second, we have not yet attempted to optimize our toolbox for speed. For example, we did not benchmark our models on processing time because we believe most users will be applying these detectors on batches of pre-recorded videos rather than in real-time applications. Currently, our models are able to process a single image in about 300 milliseconds with a GPU and about 2 seconds on a CPU. For users who need faster processing times on videos, processing can be sped up by skipping n-number of frames. We hope to optimize our code and improve processing time in future versions of our toolbox. Third, our models likely contain some degree of bias with respect to gender and race. We have attempted to use as much high quality publicly available data as possible to train our models and selected challenging real world datasets for benchmarking when possible. This problem is inherent to the field and will only improve as datasets increase in diversity and representation and preprocessing pipelines improve (e.g., faces with darker pigmentation are often more difficult to detect) [101,102] . We plan to expand our benchmarking efforts to include race and gender to aid in model selection. Fourth, our toolbox currently only includes detection of core facial features (i.e., facial landmarks, action units, and emotions) but there are additional signals in the face that can be informative for social science researchers. Head pose can be used to detect nodding or a shaking of the head which can be signals of consent or denial in social interactions. Gaze extracted from face videos can be used to infer the attention of the recorded individual. Heart rate and respiration can also be extracted from face videos [103] which can be used to infer arousal or stress levels of the recorded individual. Models for detecting these facial features could be implemented in future versions of Py-Feat pending user interest.

In summary, we introduce Py-Feat, an open source full stack framework implemented in Python for performing facial expression analysis from detection, preprocessing, analysis, and visualization. This work leverages efforts from the broader affective computing community by relying on high quality datasets, state of the art models, and building on other open source efforts such as OpenFace. We hope others in the community may be interested in improving this toolbox by providing feedback and bug reports, and also contributing bug fixes, new models and features. We have outlined our contribution guidelines as well as the necessary code and tutorials on how to replicate our work on our main project website (https://py-feat.org). We look forward to the increasing synergy between the fields of computer science and social science and welcome feedback and suggestions from the broader community as we continue to refine and add features to the Py-Feat platform.

Code availability

All the code and data to reproduce the results are available at https://github.com/py-feat and https://py-feat.org.

Acknowledgments

The authors would like to acknowledge Filippo Rossi and Nathaniel Hanes for early discussions of this work. We thank Mark Thornton, Emma Templeton, and Wasita Mahaphanit for providing feedback on earlier drafts of this paper. This work was supported by funding from the National Institute of Mental Health R01MH116026, R56MH080716 and the National Science Foundation CAREER 1848370.

References

1. Darwin, C. *The Expression of the Emotions in Man and Animals*. (1886).
2. Ekman, P. Facial expression and emotion. *Am. Psychol.* **48**, 384 (1993).
3. Ekman, P. & Friesen, W. Facial action coding system: a technique for the measurement of facial movement. *Palo Alto: Consulting Psychologists* (1978).
4. Cohn, J. F., Ambadar, Z. & Ekman, P. Observer-based measurement of facial expression with the Facial Action Coding System. *The handbook of emotion elicitation and assessment* 203–221 (2007).
5. Kilbride, J. E. & Yarczower, M. Ethnic bias in the recognition of facial expressions. *J. Nonverbal Behav.* **8**, 27–41 (1983).
6. Graesser, A. C. *et al.* Detection of emotions during learning with AutoTutor. in *Proceedings of the 28th annual meetings of the cognitive science society* 285–290 (Citeseer, 2006).
7. Fridlund, A. J., Schwartz, G. E. & Fowler, S. C. Pattern recognition of self-reported emotional state from multiple-site facial EMG activity during affective imagery. *Psychophysiology* **21**, 622–637 (1984).
8. Larsen, J. T., Norris, C. J. & Cacioppo, J. T. Effects of positive and negative affect on electromyographic activity over zygomaticus major and corrugator supercilii. *Psychophysiology* **40**, 776–785 (2003).
9. Sayette, M. A. *et al.* Alcohol and group formation: a multimodal investigation of the effects of alcohol on emotion and social bonding. *Psychol. Sci.* **23**, 869–878 (2012).
10. Navarathna, R. *et al.* Predicting movie ratings from audience behaviors. in *IEEE Winter Conference on Applications of Computer Vision* 1058–1065 (2014).
11. Golland, Y., Mevorach, D. & Levit-Binnun, N. Affiliative zygomatic synchrony in co-present strangers. *Scientific Reports* vol. 9 (2019).
12. Cheong, J. H., Brooks, S. & Chang, L. J. FaceSync: Open source framework for recording facial expressions with head-mounted cameras. *F1000Res.* (2019).
13. Cheong, J. H., Molani, Z., Sadhukha, S. & Chang, L. J. Synchronized affect in shared experiences strengthens social connection. (2020) doi:10.31234/osf.io/bd9wn.
14. De la Torre, F. *et al.* IntraFace. *IEEE Int Conf Autom Face Gesture Recognit Workshops* **1**, (2015).
15. Vemulapalli, R. & Agarwala, A. A compact embedding for facial expression similarity. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* 5683–5692 (openaccess.thecvf.com, 2019).
16. Stöckli, S., Schulte-Mecklenbeck, M., Borer, S. & Samson, A. C. Facial expression analysis with AFFDEX and FACET: A validation study. *Behav. Res. Methods* **50**, 1446–1460 (2018).
17. Haines, N., Southward, M. W., Cheavens, J. S., Beauchaine, T. & Ahn, W.-Y. Using computer-vision and machine learning to automate facial coding of positive and negative affect intensity. *PLoS One* **14**, e0211735 (2019).
18. Höfling, T. T. A., Gerdes, A. B. M., Föhl, U. & Alpers, G. W. Read My Face: Automatic Facial Coding Versus Psychophysiological Indicators of Emotional Valence and Arousal. *Front. Psychol.* **11**, 1388 (2020).
19. Dupré, D., Krumhuber, E. G., Küster, D. & McKeown, G. J. A performance comparison of eight commercially available automatic classifiers for facial affect recognition. *PLoS One* **15**,

e0231968 (2020).
20. Werner, P. *et al.* Automatic Pain Assessment with Facial Activity Descriptors. *IEEE Transactions on Affective Computing* **8**, 286–299 (2017).
21. Chen, P.-H. A. *et al.* Socially transmitted placebo effects. *Nat Hum Behav* **3**, 1295–1305 (2019).
22. Littlewort, G. C., Bartlett, M. S. & Lee, K. Automatic coding of facial expressions displayed during posed and genuine pain. *Image Vis. Comput.* **27**, 1797–1803 (2009).
23. Wang, Y. *et al.* Automatic Depression Detection via Facial Expressions Using Multiple Instance Learning. in *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)* 1933–1936 (2020).
24. Penton-Voak, I. S., Pound, N., Little, A. C. & Perrett, D. I. Personality Judgments from Natural and Composite Facial Images: More Evidence For A 'Kernel Of Truth' In Social Perception. *Soc. Cogn.* **24**, 607–640 (2006).
25. Segalin, C. *et al.* What your Facebook Profile Picture Reveals about your Personality. *Proceedings of the 25th ACM international conference on Multimedia* (2017) doi:10.1145/3123266.3123331.
26. Kachur, A., Osin, E., Davydov, D., Shutilov, K. & Novokshonov, A. Assessing the Big Five personality traits using real-life static facial images. *Sci. Rep.* **10**, 8487 (2020).
27. Kosinski, M. Facial recognition technology can expose political orientation from naturalistic facial images. *Sci. Rep.* **11**, 100 (2021).
28. Kanade, T., Cohn, J. F. & Yingli Tian. Comprehensive database for facial expression analysis. in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)* 46–53 (2000).
29. Lucey, P. *et al.* The Extended Cohn-Kanade Dataset (CK): A complete dataset for action unit and emotion-specified expression. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops* (2010) doi:10.1109/cvprw.2010.5543262.
30. Mavadati, S. M., Mahoor, M. H., Bartlett, K., Trinh, P. & Cohn, J. F. DISFA: A Spontaneous Facial Action Intensity Database. *IEEE Transactions on Affective Computing* **4**, 151–160 (2013).
31. Zhang, X. *et al.* BP4D-Spontaneous: a high-resolution spontaneous 3D dynamic facial expression database. *Image Vis. Comput.* **32**, 692–706 (2014).
32. Mavadati, M., Sanger, P. & Mahoor, M. H. Extended disfa dataset: Investigating posed and spontaneous facial expressions. in *proceedings of the IEEE conference on computer vision and pattern recognition workshops* 1–8 (2016).
33. Zhang, Z. *et al.* Multimodal spontaneous emotion corpus for human behavior analysis. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 3438–3446 (2016).
34. Krumhuber, E. G., Skora, L., Küster, D. & Fou, L. A Review of Dynamic Datasets for Facial Expression Research. *Emot. Rev.* **9**, 280–292 (2017).
35. Dhall, A., Goecke, R., Joshi, J., Sikka, K. & Gedeon, T. Emotion Recognition In The Wild Challenge 2014: Baseline, Data and Protocol. in *Proceedings of the 16th International Conference on Multimodal Interaction* 461–466 (Association for Computing Machinery, 2014).

36. Qi, P., Zhang, Y., Zhang, Y., Bolton, J. & Manning, C. D. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *arXiv [cs.CL]* (2020).
37. Brockman, G. *et al.* OpenAI Gym. *arXiv [cs.LG]* (2016).
38. *iMotions Biometric Research Platform 6.0*. (iMotions A/S, Copenhagen, Denmark, 2016).
39. Van Kuilenburg, H., Den Uyl, M. J., Israël, M. L. & Ivan, P. Advances in face and gesture analysis. *Measuring Behavior 2008* **371**, (2008).
40. Yitzhak, N. *et al.* Gently does it: Humans outperform a software classifier in recognizing subtle, nonstereotypical facial expressions. *Emotion* **17**, 1187–1198 (2017).
41. Krumhuber, E. G., Küster, D., Namba, S. & Skora, L. Human and machine validation of 14 databases of dynamic facial expressions. *Behav. Res. Methods* (2020) doi:10.3758/s13428-020-01443-y.
42. Krumhuber, E. G., Küster, D., Namba, S., Shah, D. & Calvo, M. G. Emotion recognition from posed and spontaneous dynamic expressions: Human observers versus machine analysis. *Emotion* **21**, 447–451 (2021).
43. Littlewort, G. *et al.* The computer expression recognition toolbox (CERT). in *2011 IEEE International Conference on Automatic Face Gesture Recognition (FG)* 298–305 (ieeexplore.ieee.org, 2011).
44. McDuff, D. *et al.* AFFDEX SDK: A Cross-Platform Real-Time Multi-Face Expression Recognition Toolkit. in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* 3723–3726 (Association for Computing Machinery, 2016).
45. Baltrusaitis, T., Zadeh, A., Lim, Y. C. & Morency, L. OpenFace 2.0: Facial Behavior Analysis Toolkit. in *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)* 59–66 (2018).
46. Jenkinson, M., Beckmann, C. F., Behrens, T. E. J., Woolrich, M. W. & Smith, S. M. FSL. *Neuroimage* **62**, 782–790 (2012).
47. Cox, R. W. AFNI: software for analysis and visualization of functional magnetic resonance neuroimages. *Comput. Biomed. Res.* **29**, 162–173 (1996).
48. Friston, K. J., Frith, C. D., Liddle, P. F. & Frackowiak, R. S. Comparing functional (PET) images: the assessment of significant change. *J. Cereb. Blood Flow Metab.* **11**, 690–699 (1991).
49. Abraham, A. *et al.* Machine learning for neuroimaging with scikit-learn. *Front. Neuroinform.* **8**, 14 (2014).
50. Ekman, P. & Rosenberg, E. L. *What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System (FACS)*. (Oxford University Press, 1997).
51. Paszke, A. *et al.* PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv [cs.LG]* (2019).
52. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
53. Zhang, S. *et al.* FaceBoxes: A CPU real-time face detector with high accuracy. in *2017 IEEE International Joint Conference on Biometrics (IJCB)* 1–9 (2017).
54. Zhang, L. *et al.* Multi-Task Cascaded Convolutional Networks Based Intelligent Fruit Detection for Designing Automated Robot. *IEEE Access* **7**, 56028–56038 (2019).

55. Zhang, N., Luo, J. & Gao, W. Research on Face Detection Technology Based on MTCNN. in *2020 International Conference on Computer Network, Electronic and Automation (ICCNEA)* 154–158 (2020).
56. Deng, J. *et al.* RetinaFace: Single-stage Dense Face Localisation in the Wild. *arXiv [cs.CV]* (2019).
57. Yang, S., Luo, P., Loy, C.-C. & Tang, X. Wider face: A face detection benchmark. in *Proceedings of the IEEE conference on computer vision and pattern recognition* 5525–5533 (2016).
58. Shen, J. *et al.* The first facial landmark tracking in-the-wild challenge: Benchmark and results. in *Proceedings of the IEEE international conference on computer vision workshops* 50–58 (2015).
59. Sagonas, C., Antonakos, E., Tzimiropoulos, G., Zafeiriou, S. & Pantic, M. 300 Faces In-The-Wild Challenge: database and results. *Image Vis. Comput.* **47**, 3–18 (2016).
60. Guo, X. *et al.* PFLD: A Practical Facial Landmark Detector. *arXiv [cs.CV]* (2019).
61. Howard, A. G. *et al.* MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv [cs.CV]* (2017).
62. Chen, S., Liu, Y., Gao, X. & Han, Z. MobileFaceNets: Efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices. *arXiv [cs.CV]* (2018).
63. Sagonas, C., Tzimiropoulos, G., Zafeiriou, S. & Pantic, M. A semi-automatic methodology for facial landmark annotation. in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* 896–903 (2013).
64. Hyniewska, S., Sato, W., Kaiser, S. & Pelachaud, C. Naturalistic Emotion Decoding From Facial Action Sets. *Front. Psychol.* **9**, 2678 (2018).
65. Grafsgaard, J. F., Boyer, K. E. & Lester, J. C. Predicting Facial Indicators of Confusion with Hidden Markov Models. in *Affective Computing and Intelligent Interaction* 97–106 (Springer Berlin Heidelberg, 2011).
66. Julle-Danière, E. *et al.* Are there non-verbal signals of guilt? *PLoS One* **15**, e0231756 (2020).
67. Shao, Z., Liu, Z., Cai, J. & Ma, L. Deep adaptive attention for joint facial action unit detection and face alignment. in *Proceedings of the European Conference on Computer Vision (ECCV)* 705–720 (2018).
68. Shao, Z., Liu, Z., Cai, J. & Ma, L. JÂA-net: Joint facial action unit detection and face alignment via adaptive attention. *Int. J. Comput. Vis.* **129**, 321–340 (2021).
69. Li *et al.* An EEG-Based Multi-Modal Emotion Database with Both Posed and Authentic Facial Actions for Emotion Analysis. in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020) (FG)* vol. 0 336–343 (2020).
70. Lucey, P., Cohn, J. F., Prkachin, K. M., Solomon, P. E. & Matthews, I. Painful data: The UNBC-McMaster shoulder pain expression archive database. in *2011 IEEE International Conference on Automatic Face Gesture Recognition (FG)* 57–64 (2011).
71. Kollias, D., Nicolaou, M. A., Kotsia, I., Zhao, G. & Zafeiriou, S. Recognition of affect in the wild using deep neural networks. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* 26–33 (2017).
72. Zafeiriou, S. *et al.* Aff-Wild: Valence and Arousal 'In-the-Wild' Challenge. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2017)

doi:10.1109/cvprw.2017.248.

73. Kollias, D. *et al.* Deep Affect Prediction in-the-Wild: Aff-Wild Database and Challenge, Deep Architectures, and Beyond. *Int. J. Comput. Vis.* **127**, 907–929 (2019).
74. Kollias, D. & Zafeiriou, S. Expression, Affect, Action Unit Recognition: Aff-Wild2, Multi-Task Learning and ArcFace. *arXiv [cs.CV]* (2019).
75. Kollias, D., Sharmanska, V. & Zafeiriou, S. Face Behavior a la carte: Expressions, Affect and Action Units in a Single Network. *arXiv [cs.CV]* (2019).
76. Kollias, D., Schulc, A., Hajiyev, E. & Zafeiriou, S. Analysing Affective Behavior in the First ABAW 2020 Competition. *arXiv [cs.LG]* (2020).
77. Baltrušaitis, T., Mahmoud, M. & Robinson, P. Cross-dataset learning and person-specific normalisation for automatic Action Unit detection. in *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)* vol. 06 1–6 (2015).
78. Dalal, N. & Triggs, B. Histograms of oriented gradients for human detection. in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* vol. 1 886–893 vol. 1 (2005).
79. van der Walt, S. *et al.* scikit-image: image processing in Python. *PeerJ* **2**, e453 (2014).
80. Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* **12**, 2825–2830 (2011).
81. Barrett, L. F., Adolphs, R., Marsella, S., Martinez, A. M. & Pollak, S. D. Emotional Expressions Reconsidered: Challenges to Inferring Emotion From Human Facial Movements. *Psychological Science in the Public Interest* vol. 20 1–68 (2019).
82. Haines, N. *et al.* Regret Induces Rapid Learning from Experience-based Decisions: A Model-based Facial Expression Analysis Approach. *bioRxiv* 560011 (2019) doi:10.1101/560011.
83. Luan, P., Huynh, V. & Tuan Anh, T. Facial Expression Recognition using Residual Masking Network. in *IEEE 25th International Conference on Pattern Recognition* 4513–4519 (2020).
84. Goodfellow, I. J. *et al.* Challenges in representation learning: A report on three machine learning contests. *Neural Networks* vol. 64 59–63 (2015).
85. Zhang, Z., Luo, P., Loy, C. C. & Tang, X. From facial expression recognition to interpersonal relation prediction. *Int. J. Comput. Vis.* **126**, 550–569 (2018).
86. Lyons, M., Kamachi, M. & Gyoba, J. *The Japanese Female Facial Expression (JAFFE) Dataset*. (1998). doi:10.5281/zenodo.3451524.
87. Mollahosseini, A., Hasani, B. & Mahoor, M. H. AffectNet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Trans. Affect. Comput.* **10**, 18–31 (2019).
88. McKinney, W. & Others. pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing* **14**, 1–9 (2011).
89. Afzal, S. & Robinson, P. Natural affect data — Collection & annotation in a learning context. *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops* (2009) doi:10.1109/acii.2009.5349537.
90. Jones, E., Oliphant, T., Peterson, P. & Others. SciPy: Open source scientific tools for Python. (2001).
91. Chang, L. *et al.* *cosanlab/nltools: 0.3.14*. (Zenodo, 2019). doi:10.5281/ZENODO.2229812.

92. Fabian Benitez-Quiroz, C., Srinivasan, R. & Martinez, A. M. Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild. in *Proceedings of the IEEE conference on computer vision and pattern recognition* 5562–5570 (2016).

93. Zhu, Y., Cai, H., Zhang, S., Wang, C. & Xiong, Y. TinaFace: Strong but Simple Baseline for Face Detection. *arXiv [cs.CV]* (2020).

94. Jourabloo, A., Ye, M., Liu, X. & Ren, L. Pose-invariant face alignment with a single cnn. in *Proceedings of the IEEE International Conference on computer vision* 3200–3209 (2017).

95. Zhu, X., Lei, Z., Liu, X., Shi, H. & Li, S. Z. Face alignment across large poses: A 3d solution. in *Proceedings of the IEEE conference on computer vision and pattern recognition* 146–155 (2016).

96. Valle, R., Buenaposada, J. M., Valdés, A. & Baumela, L. Face alignment using a 3D deeply-initialized ensemble of regression trees. *Computer Vision and Image Understanding* vol. 189 102846 (2019).

97. Valle, R., Buenaposada, J. M., Valdes, A. & Baumela, L. A deeply-initialized coarse-to-fine ensemble of regression trees for face alignment. in *Proceedings of the European Conference on Computer Vision (ECCV)* 585–601 (2018).

98. Watson, D. M., Brown, B. B. & Johnston, A. A data-driven characterisation of natural facial expressions when giving good and bad news. *PLoS Comput. Biol.* **16**, e1008335 (2020).

99. Chang, L. J. *et al.* Endogenous variation in ventromedial prefrontal cortex state dynamics during naturalistic viewing reflects affective experience. *bioRxiv* (2018).

100. Jack, R. E., Garrod, O. G. B. & Schyns, P. G. Dynamic facial expressions of emotion transmit an evolving hierarchy of signals over time. *Curr. Biol.* **24**, 187–192 (2014).

101. Rhue, L. Racial Influence on Automated Perceptions of Emotions. (2018) doi:10.2139/ssrn.3281765.

102. Nagpal, S., Singh, M., Singh, R. & Vatsa, M. Deep Learning for Face Recognition: Pride or

Prejudiced? *arXiv [cs.CV]* (2019).

103. McDuff, D., Gontarek, S. & Picard, R. Remote measurement of cognitive stress via heart rate variability. *Conf. Proc. IEEE Eng. Med. Biol. Soc.* **2014**, 2957–2960 (2014).