

# **Name this expression**

## **Final report**

# Table of contents

Table of contents	2
Summary	3
Defining the problem & client	3
Background on Computer vision	3
Context for project	3
Goal of the project	3
Data & Sources	4
Data exploration	4
Additional datasets of facial expressions	6
Data Wrangling	7
Data_exploration.ipynb	7
Image_processing.ipynb	7
Data_wrangling.ipynb	11
Classification Model	12
Baseline	12
KNN Model	12
Convolutional neural network (CNN)	13
Conclusions	22
References	22
Appendix	23
Appendix 1: CNN Architectures	23
Appendix 2: Confusion Matrices for classifiers 1-11	26

# Summary

This report guides the reader through the development of a facial expressions (limited to neutral, happy, sad, contempt, surprise, anger, and disgust) classifier. A practical application of the classifier is its use in games on mobile or web applications to assist in developing practicing social skills of children with autism in environments with less pressure.

This reports the development of a proof of concept model that has achieved an accuracy of ~86%. It covers aspects of data acquisition from the CK+ dataset, facial image processing and the training and evaluation of KNN and CNN classification models.

The project uses OpenCV for the image processing and Keras' deep learning for the model.

## Defining the problem & client

### Background on Computer vision

Data science has experienced tremendous growth in the past few years, helping society to better understand the world around us.

Can a computer be taught to see? One area that has received significant attention is computer vision. Computer vision involves training computers to make sense of pictures and videos in order to extract a high-level understanding.

Computer vision's recent growth has been fuelled by the interest in drones, robots and self-driving cars. Computer vision has also made great progress due to advancements in the implementations of neural networks and deep learning.

Growth has not been limited to frontier applications of technology, computer vision is being used in existing industries. One common application of computer vision is facial recognition. This includes auto focusing in cameras, facial recognition technology in surveillance and border control, as well as facial recognition technology for unlocking smart phone devices.

### Context for project

Children on the autism spectrum have difficulties recognising facial expressions, body language and tonality of voice. Practicing social skills, such as identifying facial expressions, in environments with less pressure allows these children to build confidence and assists with the application of the skills in daily life. <http://luxai.com/2017/11/29/improve-emotion-recognition-understanding-children-autism/>

Games on mobile and web applications are a useful and scalable approach to developing such skills in children with autism. However to automate the classification of facial expressions in these applications, computers must first learn to recognise facial expressions.

### Goal of the project

The goal of this project is to build a tool that correctly identifies facial expressions (limited to neutral, happy, sad, contempt, surprise, anger, and disgust) from photographs.

# Data & Sources

The data was sourced from the Cohn-Kanade AU-Coded Facial Expression (CK+) [database](#) by P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews. The database includes posed and spontaneous expressions with metadata. The expressions have discrete labels (neutral, sadness, surprise, happiness, fear, anger, contempt and disgust) and are mostly grey.

## Data exploration

The full CK+ dataset is loaded and explored in notebook: 01\_data\_exploration.ipynb

The full dataset has 10,708 images in 593 sequences across 123 subjects. Each sequence has an average of 18 images but only 1 facial expression label that corresponds to the last (peak) frame. The emotions have labels ranging from 0 to 7. (where 0=neutral, 1=anger, 2=contempt, 3=disgust, 4=fear, 5=happy, 6=sadness, 7=surprise).

Each sequence begins with a neutral expression in the first frame and progresses to the peak frame. Figure 1 below is an example of 1 sequence (subject S111, sequence 007) for disgust:



Figure 1: Image sequence 1

The images show that for the first 4-6 images in this sequence, the facial expression is fairly neutral. Later in the sequence (image 7 / 8), the expression of disgust becomes more pronounced.

A similar progression is seen in other sequences such as figure 2 below.



Figure 2: Image sequence 2

In addition, examining figure 1 and figure 2 reveals that the subject occupies different proportions of the images and that the angles of head tilt varies.

Only 327 sequences of the 593 sequences, are labelled. The labelled sequences have a total of 5,876 images. If we assume the first 4 frames are a neutral expression we have the following distribution of expressions (note that neutral expression sequences are a double count):

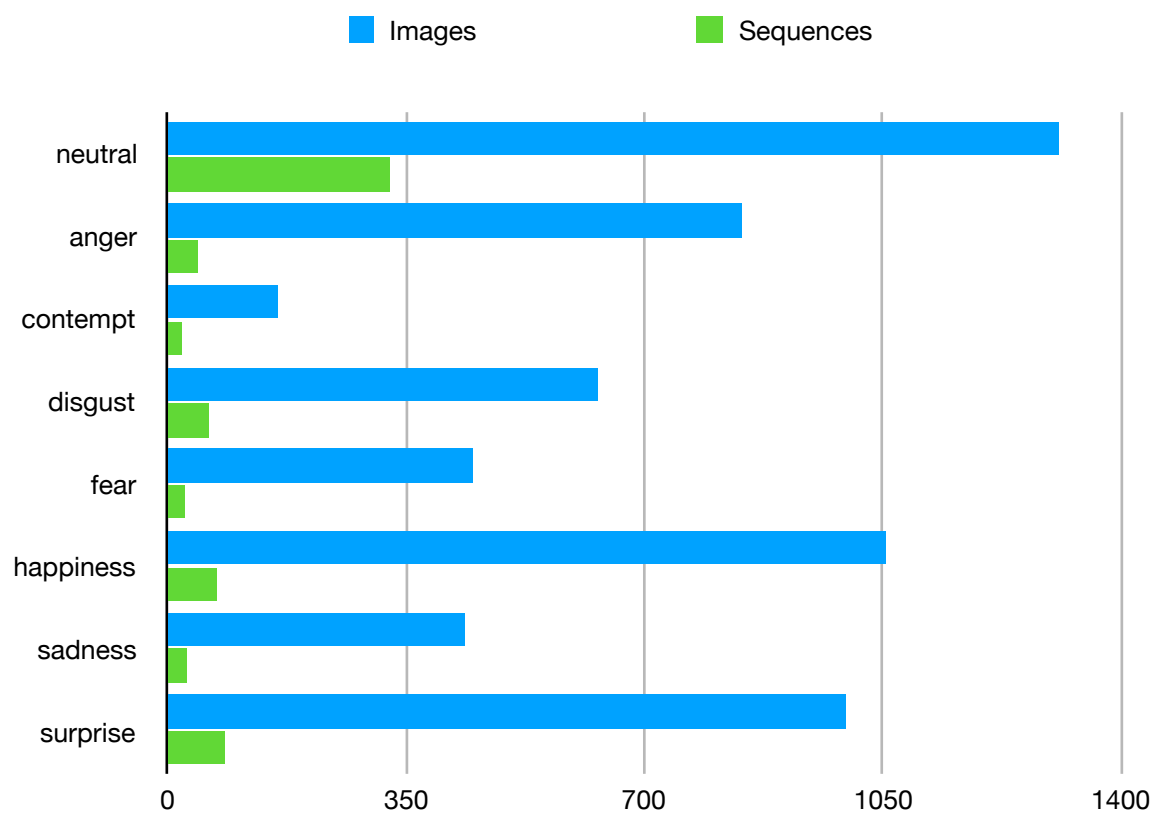


Figure 2: Distribution of raw data

The distribution of images and sequences for each emotion are very uneven. The number of images per sequence for various emotions vary. For example, contempt has an average of 9 (excluding the first 4) images per sequence, whereas anger has an average of 19 images per sequence.

## **Additional datasets of facial expressions**

The Japanese Female Facial Expressions (JAFPE) database are annotated images of 60 Japanese female subjects. The emotions are mean semantic ratings for 6 expressions (sadness, surprise, happiness, fear, anger, and disgust) rated using 1-5 scale.

The CMU Face Images Dataset consist of grey images of 20 individuals with 4 head positions (straight, left, right, up), 4 facial expressions (neutral, happy, sad, angry), 2 eye states (open, sunglasses). A good description of the dataset can be found [here](#).

# Data Wrangling

For details regarding how the data was joined, processed and wrangled, please refer to the following notebooks:

01\_data\_exploration.ipynb

02\_image\_processing.ipynb

03\_data\_wrangling.ipynb

## Data\_exploration.ipynb

The source data came as a zip file of images and separate text files with emotion labels. In the data exploration notebook, a data frame of image information was created. Leveraging the glob module to navigate the CK+'s folder structure, a data frame with the following columns was constructed:

- image : file name of image file (e.g. S111\_007\_00000001.png)
- image\_path: path for the raw image file (e.g. ../01\_raw\_data/CK+/cohn-kanade-images/S111/007/S111\_007\_00000001.png)
- sequence: sequence number (e.g. S111\_007)
- image\_number: order of the image in the sequence (e.g. 7)

Next the emotion labels assigned to the peak frame needed to be matched to each image. For this, a dictionary was created with sequence number as the key and the emotion label as the value. Using the dictionary, the emotion label was then mapped onto the sequence found in the data frame. The emotion label was stored in a new column:

- emotion: label assigned to peak frame in sequence (where 0=neutral, 1=anger, 2=contempt, 3=disgust, 4=fear, 5=happy, 6=sadness, 7=surprise)

The data frame was then cleaned up:

1. the images with no corresponding emotion labels were dropped, reducing the number of observations from 10,708 images to 5,876 images
2. emotions were set to integers
3. emotion of first 4 images in each sequence were labelled neutral (i.e. category 0) .

Once this was done, the data was saved to a CSV and verified.

## Image\_processing.ipynb

The image is processed to help improve the outcome of the classification (training and application) by making the images more consistent. This section explains the steps used in the image processing stage. The following reference was helpful for more information regarding image processing: <https://github.com/PacktPublishing/OpenCV-Computer-Vision-Projects-with-Python>

### Face recognition

The image is scanned for faces using OpenCV's `haarcascade_frontalface` (<https://github.com/opencv/opencv/tree/master/data/haarcascades>) classifier. Since there is only 1 face per image in our dataset, the first face recognised is used.

Figure 3 visualises this process by drawing a green box around the identified face. The region highlighted is then extracted from the image.



Figure 3: Face recognition example

### Normalising the facial image

This step aims to achieve 3 things in order to normalise the image:

1. Adjust the tilt of the face such that the eyes lie along the same horizontal line
2. Scale the image such that the faces are approximately the same size
3. Center the face around the same point

In image processing terms, the image's tilt, translation and scale must be homogenised across the data set. The accuracy of image classification models are said to benefit from these steps. (<https://www.pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/>)

An example of the desired processed image is shown in figure 4 below.

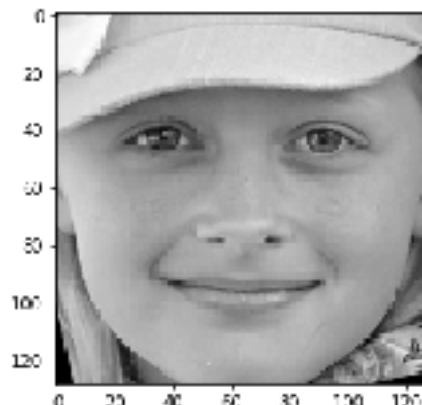


Figure 4: Desired processed image



To achieve this, the following steps are taken:

- The left and right eyes are detected using OpenCV's `haarcascade_lefteye` and `haarcascade_righteye` classifiers, as illustrated by the blue boxes in figure 5 below.

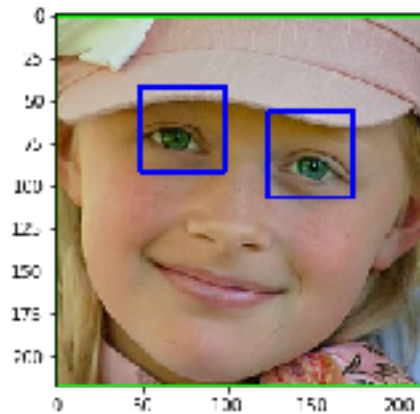


Figure 5: Eye recognition example

- Next the center point of these two regions are taken and a line is extended between them, as illustrated by the red dots in figure 6 below.

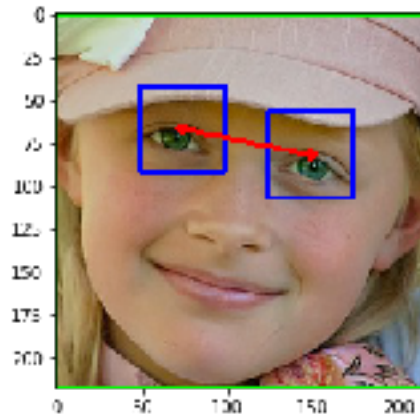


Figure 6: Taking the centre of each eye region

- The focal point for the image transformation is the middle of the red line. This is illustrated by the green dot in figure 7 below. It will be the centreline of the adjusted image and the focal point of the image rotation.

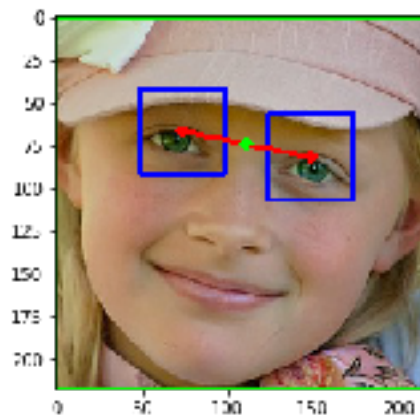


Figure 7: Focal point for image transformation

- The image is then transformed around the green dot such that it is:
  1. Rotated to make the red line horizontal
  2. Resized to make the distance between the centre of the 2 eye regions (i.e. the red line) take 40% of the image width of 128px. The width is equal to the image height of 128px.

- Translated such that the middle of the 2 eye regions (ie. the green dot) is in the middle of the image's width and 30% of the image's height.

The resulting transformed image is shown in Figure 8 below.

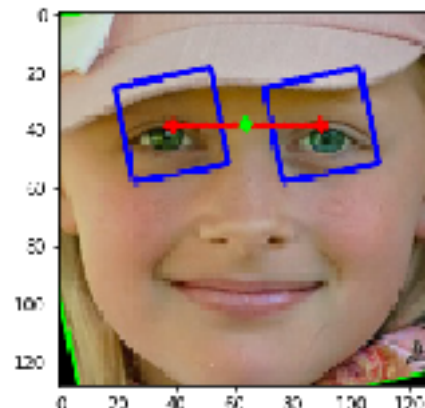


Figure 8: Transformed image

- Finally, the transformed image is converted to greyscale and saved in a folder of processed images. The path for the processed image is then saved in the data frame from the data exploration stage and the saved data frame is validated. Figure 9 below shows the processed images for the first sequence.



Figure 9: Processed images from image sequence 1

## Data\_wrangling.ipynb

This stage of the process focuses on:

- splitting the images into training and test sets
- importing the images as data
- flattening the image
- saving out the data

Unlike a typical data set of independent images, the images are organised into sequences. Hence, instead of splitting by images, the data set was first split by sequences. This helps to ensure that the test set contained images from sequences that the model was not trained on.

The training set comprised of 90% of the total sequences, this corresponds to 5297 images. Conversely, the test set comprised of 10% of the total sequences and had 579 images. Table 1 below shows the split between training and test images by emotion.

Table 1: Training and test image split

	# Training images	# Test images
<b>neutral</b>	1176	132
<b>anger</b>	739	103
<b>contempt</b>	151	10
<b>disgust</b>	568	64
<b>fear</b>	413	33
<b>happiness</b>	949	106
<b>sadness</b>	378	57
<b>surprise</b>	923	74

# Classification Model

## Baseline

The baseline performance of a classification model would be equivalent to randomly assigning a class. Since there are a total of 8 classes, this would correspond to a baseline accuracy of 12.5%.

## KNN Model

For further details regarding the model, please refer to the Jupyter notebook: 04\_knn\_classification.ipynb

The simplest classifier for this problem is a K-nearest Neighbour (KNN) model. The KNN model was constructed using sklearn's KNeighborsClassifier. The X value inputs for the model were the grey scale flattened image data described in the section above split into training and test sets. The corresponding y value inputs for the model were the integer representation of emotion labels. This "raw pixel" model was neither scaled (i.e. divided by 255) nor was it normalised (i.e. adjusted histogram).

The results of the model are as follows for the training and test data:

Raw pixel training accuracy: 83.6%

Table 2: Confusion matrix for KNN training data

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	1025	37	21	20	11	31	20	11
anger	90	631	1	4	4	4	5	0
contempt	15	1	133	0	0	0	2	0
disgust	122	24	2	406	4	6	3	1
fear	49	2	0	0	358	3	1	0
happiness	120	4	1	5	0	818	0	1
sadness	62	17	0	5	2	0	292	0
surprise	123	5	0	5	4	10	11	765

Raw pixel test accuracy: 23.0%

Table 3: Confusion matrix for KNN test data

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	111	8	4	0	2	1	4	2
anger	61	5	4	5	7	4	17	0
contempt	5	3	0	0	2	0	0	0
disgust	35	2	3	1	2	21	0	0
fear	23	0	0	0	0	0	0	10
happiness	76	2	0	17	0	3	5	3
sadness	33	10	7	2	0	0	0	5
surprise	45	2	0	10	2	2	0	13

This model's performance is an improvement over the baseline accuracy of 12.5% but not accurate enough for practical application.

## Convolutional neural network (CNN)

In order to improve the performance of the KNN model, a CNN deep learning model was used.

---

### General architecture

The architecture of CNNs used in image classification may be generalised by figure 10.

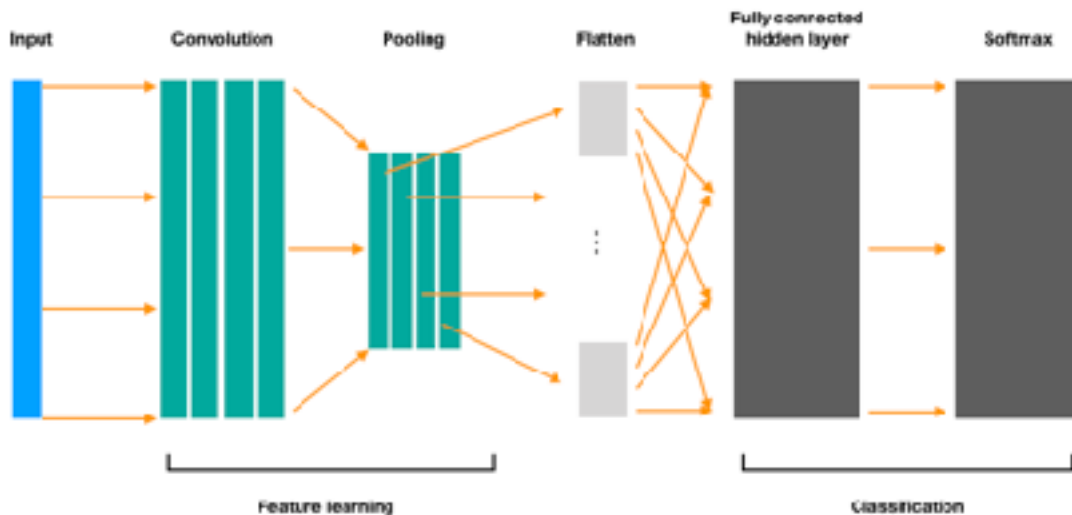


Figure 10: general form of CNN architecture for computer vision

**Input layer:** To feed the images into a 2D convolution layer of the CNN, the data needs to have the following dimensions: (?, a, b, c). Since the image is a 128 by 128 pixels greyscale, the data has to be reshaped into (?, 128, 128, 1) dimensions. This applies to both X\_train and X\_test data.

**Feature learning:** Feature extraction is done in the 2D convolutional layers of the CNN. The convolutional layer applies a filter over a receptive field to create feature maps. Pooling generalises the pixels of the image and reduces its size. By repeating convolution and pooling layers, the network may extract features from images.

**Flattening:** The original image data is multi-dimensional. Inputs into the hidden layers are drawn from single dimensional (i.e. flat) data. This flattening stage simply reshapes the data.

**Classification:** The interpreting of features and their combinations in relation to classes is done in the fully connected hidden layers. The number of hidden layers and their nodes may vary. In addition, dropout layers may be added to reduce the risk of overfitting. The dropout layer randomly ignores neurons during the training phase to prevent an over reliance on individual nodes. For more information, please refer to this article (<https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/>).

**Output:** The classification problem has mutually exclusive classes. A softmax function is used in the output layer and it ensures that the summation of all output is 1.

## Training the model

The classifiers were trained using a maximum of 20 epochs, with a batch size of 128, a validation split of 0.1 and an early stopping function with a patience of 10 epochs based on validation accuracy.

Trial and error was used to find a suitable CNN architecture. The architectures of classifiers 1-11 are shown in Appendix 1.

Once the models were trained, the test data was used to predict the emotion classification. These predictions were then compared with the actual labels using a confusion matrix. The confusion matrices for architectures 1-11 are shown in Appendix 2.

Classifier architectures 1-5 had 2 convolutional 2D layers. Changing the dense layers in the classification layers did not help improve model performance. The classifiers mostly predicted a single class with a poor overall performance.

Classifier 6 had 4 convolutional 2D layers followed by a dense layer with few nodes. This architecture was inspired by a similar project by Gábor Vecsei (<https://github.com/gaborvecsei/Emotion-Recognition>). Architectures 7 to 11 were based off classifier 6 but with varying number of nodes in the hidden layers. It was found that only classifiers 6, 8 and 11 had respectable performance. Classifiers 7, 9 and 10, predicted single classes resulting in poor overall accuracy.

The best performing model was classifier 6. The architecture and prediction results of this model are discussed in the subsequent sections.

It should be noted that although the maximum number of epochs for training was set to 20, the models that performed poorly (e.g. Classifier 10) failed to perform any better even when the number of epochs was increased. For example, classifier 10 was trained on 200 epochs and yielded the same results (implying that the model was unable to learn). Since the model wasn't learning at all, it failed to converge. More experimentation would be required to debug the model's inability to learn.

---

## Exploring outputs from convolutional layers

There were 4 layers of 2D convolution (32, 32, 64, 128) and 4 pooling layers. The outputs from the 4 convolution layers are shown below. These help to illustrate the features extracted in each of the convolution layers. This was done with reference to this source <https://github.com/dipanjanS/practical-machine-learning-with-python>

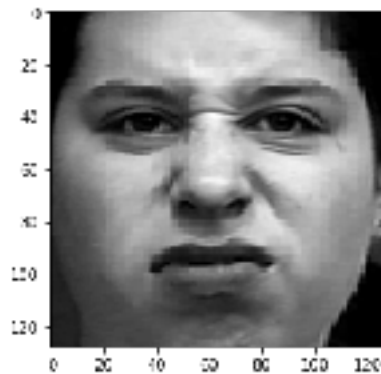


Figure 11: Image input

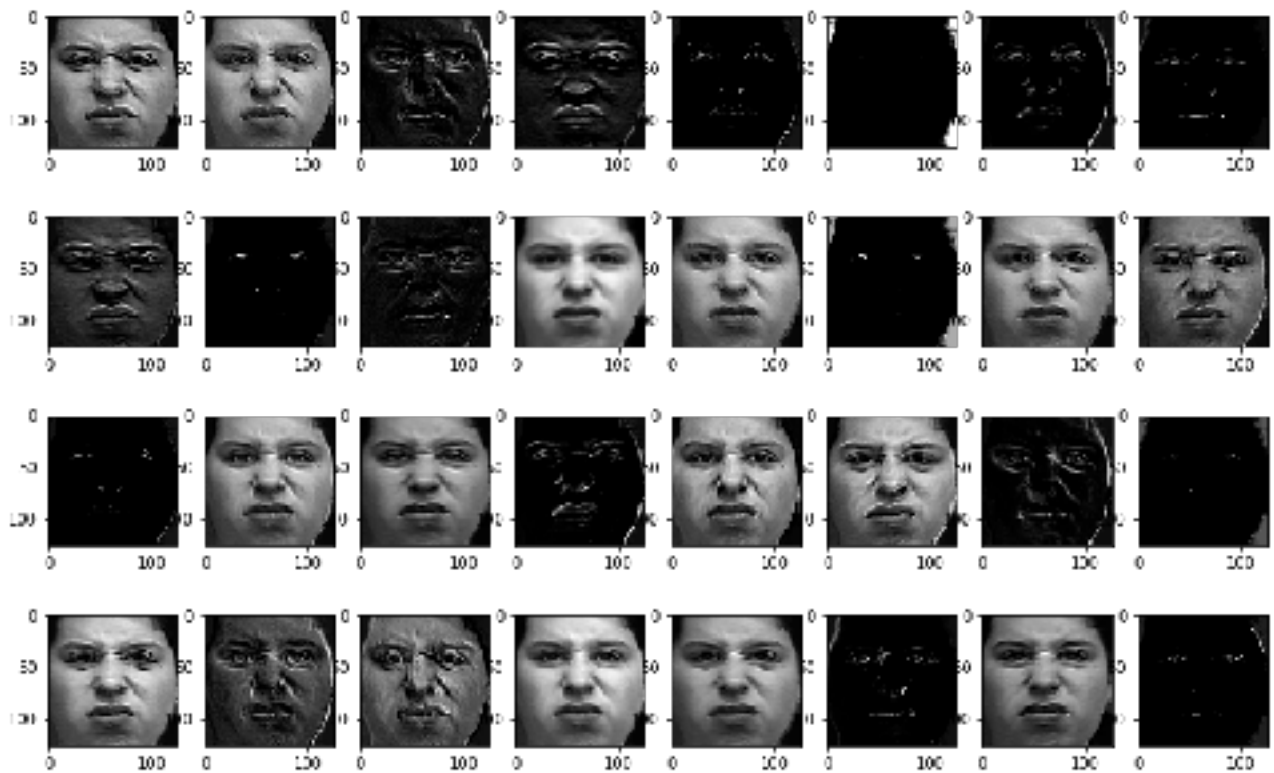


Figure 12: Output of first 2D Convolution layer

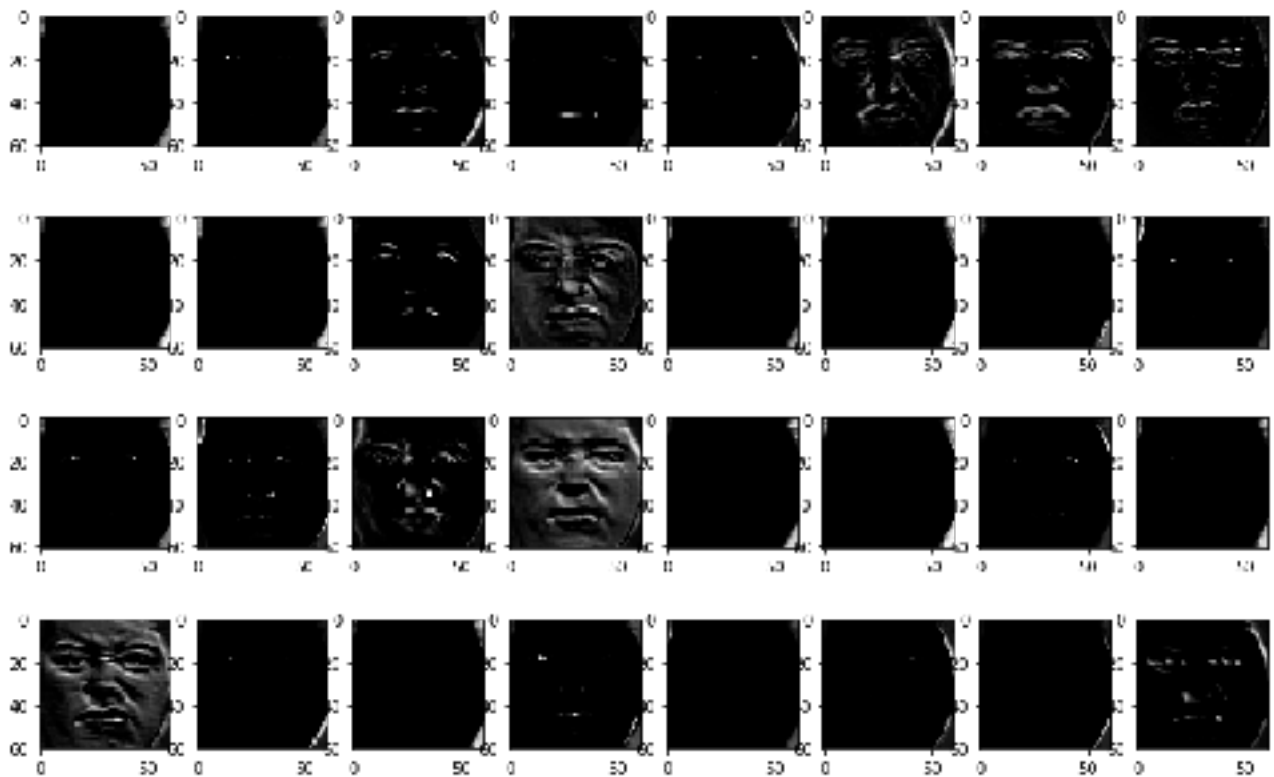


Figure 13: Output of second 2D Convolution layer

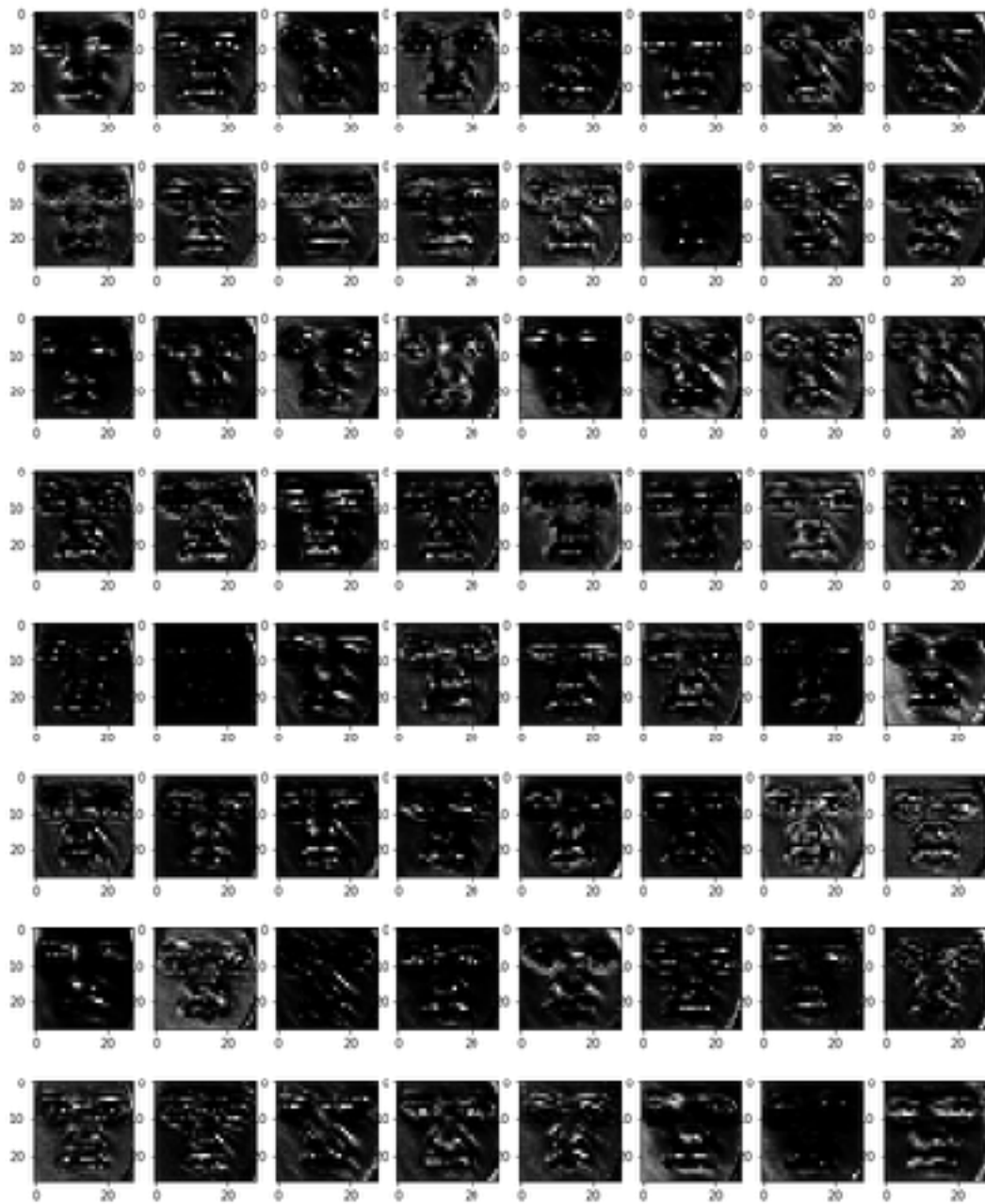


Figure 14: Output of third 2D Convolution layer



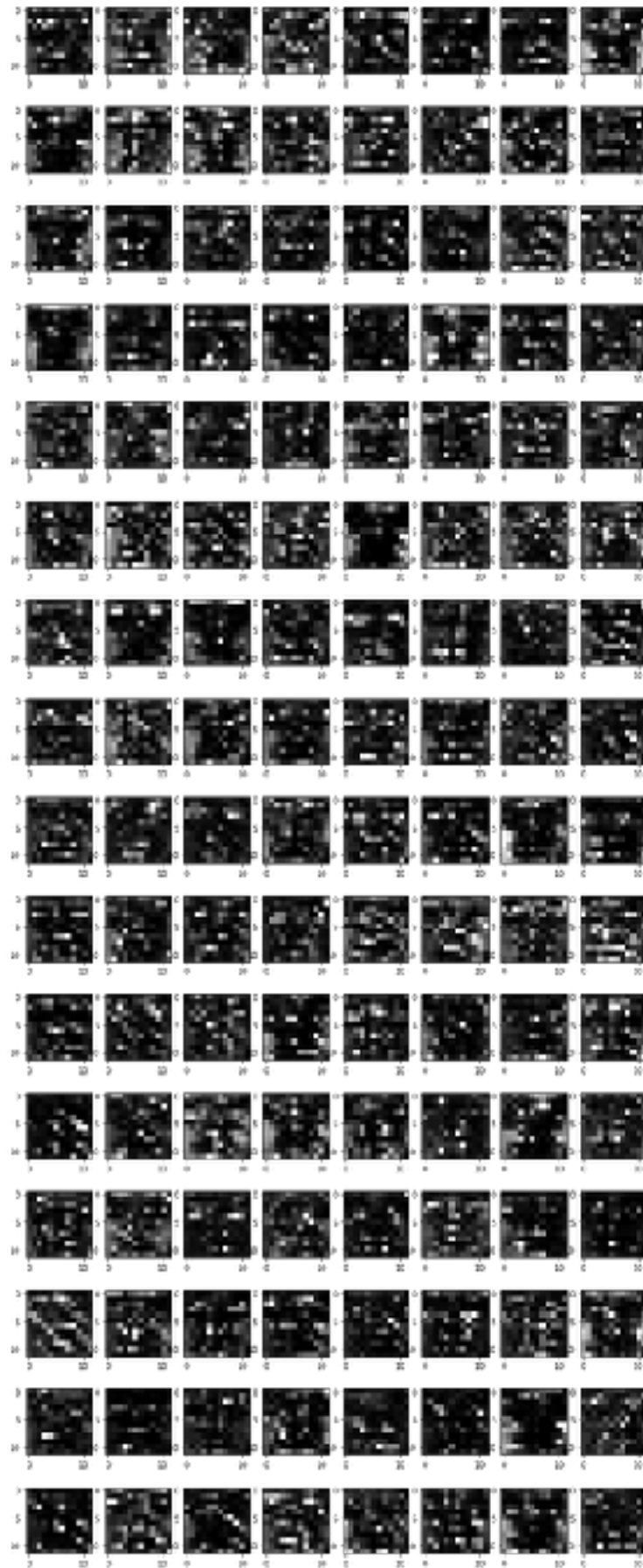


Figure 15: Output of fourth 2D Convolution layer

## Evaluating performance

Classification model 6 had the best performance. Its architecture is as follows:

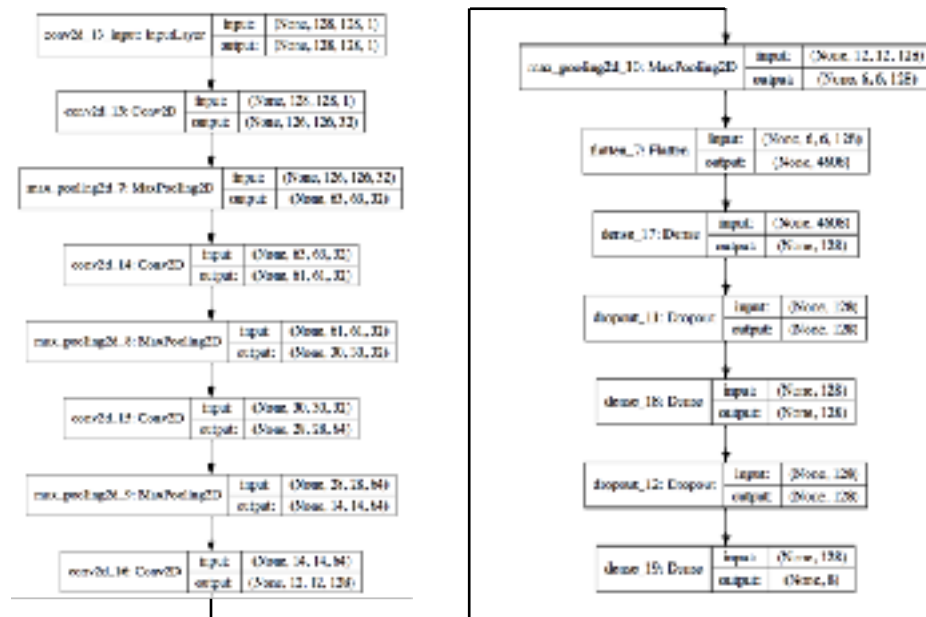


Figure 16: CNN architecture of classifier 6

The confusion matrix that arose from predicting the test set using classifier 6 and the actual labels is shown in table 4 below:

Table 4: Confusion matrix for CNN test data

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	116	4	0	1	0	1	4	6
anger	37	50	0	7	0	0	7	2
contempt	4	0	1	0	3	2	0	0
disgust	7	0	0	57	0	0	0	0
fear	7	0	0	0	12	0	3	11
happiness	19	0	0	2	4	80	0	1
sadness	32	1	0	0	5	0	19	0
surprise	6	0	0	0	0	0	0	68

The overall Accuracy is 69.6%, the precision 74.9% and the recall is 69.6%.

The precision and recall for class in classifier 6 are shown in table 5 below:

Table 5: Precision and recall per class for classifier 6

	Precision	Recall
neutral	50.9%	87.9%
anger	90.9%	48.5%
contempt	100%	10.0%
disgust	85.1%	89.1%
fear	50.0%	36.4%
happiness	96.4%	75.5%
sadness	57.6%	33.3%
surprise	77.3%	91.9%

To model's accuracy of ~70% can be partially attributed to the raw data and the processing that was applied. In particular, for every sequence, the first four frames were labelled neutral, the remaining frames were labelled with the emotion of the peak frame. This approach also does not account for the gradations in expression, mixed expressions or ambiguous expressions.

Figure 17 uses the sequence of expressions previously discussed: frames 5 and 6 would be labelled as disgust. Most humans would struggle to label frames 5 and 6 as disgust. Frame 7 would be labelled as disgust in the same way that frame 14 would be. This expression (disgust) could also have overlapping similarities with contempt and anger.



Figure 17: Processed images from image sequence 1

To attempt to address the issue of mislabeling, an additional dataset of only first two frames (neutral) and last two frames (peak expression) was created. (Note, at least 2 frames from each end was required as it is difficult to train a neural net with under 1000 observations). More details about this can be found in the notebook: 06\_data\_wrangling\_miniset.ipynb

This new dataset is a much smaller version of the original CK+ dataset that contains a total of 1176 training images and 132 test images. These comprise of the following expressions shown in table 6:

Table 6: Training and test image split for mini dataset

	# Training images	# Test images
<b>neutral</b>	588	66
<b>anger</b>	80	10
<b>contempt</b>	32	4
<b>disgust</b>	106	12
<b>fear</b>	44	6
<b>happiness</b>	124	14
<b>sadness</b>	48	8
<b>surprise</b>	154	12

Training using the same architecture and the mini dataset improved the results. The confusion matrix that arose from predicting the test mini set using classifier 6 trained on the mini training set is shown in table 7 below:

Table 7: Confusion matrix for classifier 6 trained on the mini dataset

	<b>neutral</b>	<b>anger</b>	<b>contempt</b>	<b>disgust</b>	<b>fear</b>	<b>happiness</b>	<b>sadness</b>	<b>surprise</b>
<b>neutral</b>	66	0	0	0	0	0	0	0
<b>anger</b>	0	9	0	0	0	0	1	0
<b>contempt</b>	1	0	3	0	0	0	0	0
<b>disgust</b>	0	1	1	10	0	0	0	0
<b>fear</b>	0	0	0	0	3	0	1	2
<b>happiness</b>	2	0	0	2	1	9	0	0
<b>sadness</b>	4	2	0	0	0	0	2	0
<b>surprise</b>	0	0	0	0	0	0	0	12

The overall Accuracy is 86.4%, the precision 85.6% and the recall is 86.4%.

The precision and recall for each class are shown in the table 8 below:

Table 8: Precision and recall for classes for classifier 6 trained on mini dataset

	<b>Precision</b>	<b>Recall</b>
<b>neutral</b>	90.4%	100%
<b>anger</b>	75.0%	90.0%
<b>contempt</b>	75.0%	75.0%
<b>disgust</b>	83.3%	83.3%
<b>fear</b>	75.0%	50.0%
<b>happiness</b>	100%	64.3%
<b>sadness</b>	50.0%	25.0%
<b>surprise</b>	85.7%	100%

An attempt to further improve the model by normalizing the data (i.e. dividing pixel intensity by 255), however the model trained on the normalised mini dataset failed to converge.

It should also be noted that the mini dataset had different training and test datasets. Therefore, direct comparisons of accuracy between datasets should be made with caution.

---

## Understanding mis-classifications

To better understand the images that were misclassified, the misclassified images were filtered out then labeled with the predicted and actual classes. Images from subjects with consent for publication are shown in figure 12 below. Note that some images might seem repeated but they are in fact slightly different - they are sequential frames of the same sequence.


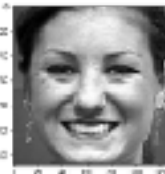
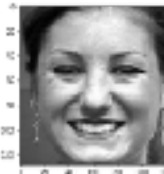
					
Predicted	Surprise	Surprise	Sadness	Disgust	Disgust
Actual	Fear	Fear	Fear	Happiness	Happiness

Figure 12: Selection of misclassified images from mini-dataset

The first 3 images demonstrate how facial expressions can be challenging to classify for humans. This is likely because emotions and facial expressions are complex and often mixed. This would be compounded by issues caused by posed expressions, which could be less 'pure' and focus on certain facial muscles that the subject is able to control for the pose.

However the final 2 images suggest that the classifier still lacks the ability to perform classifications that most humans are able to do.

---

## Potential improvements

To further improve the accuracy of classification can be taken:

1. Train the model on more images. In particular classes with fewer samples
2. Alter the problem from a strict classification model to one with semantic ratings - see the JAFFE dataset for an example
3. Produce a model that outputs the probability of a class so ambiguous results can be manually verified
4. Pre-process the image by adjusting the image's histogram
5. Use more advanced preprocessing with more facial landmarks identified to avoid cropping off parts of the face
6. Have a staged classification approach where things like gender and race are determined before applying a more sensitive emotional classifier (for example: <https://www.wired.com/story/how-coders-are-fighting-bias-in-facial-recognition-software/>)

# Conclusions

The CK+ dataset was used to train and test an emotion classifier. Images are first processed by identifying the face and adjusting the image to be corrected for tilt, resized and centred on the face.

Though the use of a subset with more accurately labeled images, a classifier of 86% accuracy was developed. Whilst this level of accuracy might be good for a proof of concept, the use of automated classifiers for teaching children skills for daily living require an even greater level of accuracy. Suggestions such as flagging ambiguous images for manual labelling could help overcome the shortfalls of this classifier.

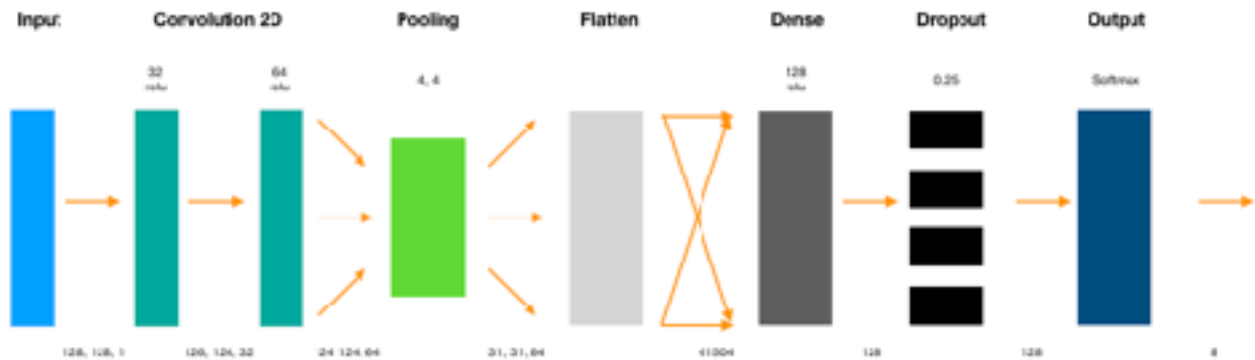
# References

- <http://luxai.com/2017/11/29/improve-emotion-recognition-understanding-children-autism/>
- Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I. (2010). The Extended Cohn-Kanade Dataset (CK+): A complete expression dataset for action unit and emotion-specified expression. Proceedings of the Third International Workshop on CVPR for Human Communicative Behavior Analysis (CVPR4HB 2010), San Francisco, USA, 94-101.
- [Japanese Female Facial Expressions \(JAFPE\) database](#)
- [CMU Face Images Dataset](#)
- <https://www.pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/>
- <https://github.com/gaborvecsei/Emotion-Recognition>
- <https://github.com/dipanjanS/practical-machine-learning-with-python>
- <https://github.com/PacktPublishing/OpenCV-Computer-Vision-Projects-with-Python>
- <https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/>
- <https://github.com/opencv/opencv/tree/master/data/haarcascades>
- <https://keras.io/>
- <https://www.wired.com/story/how-coders-are-fighting-bias-in-facial-recognition-software/>

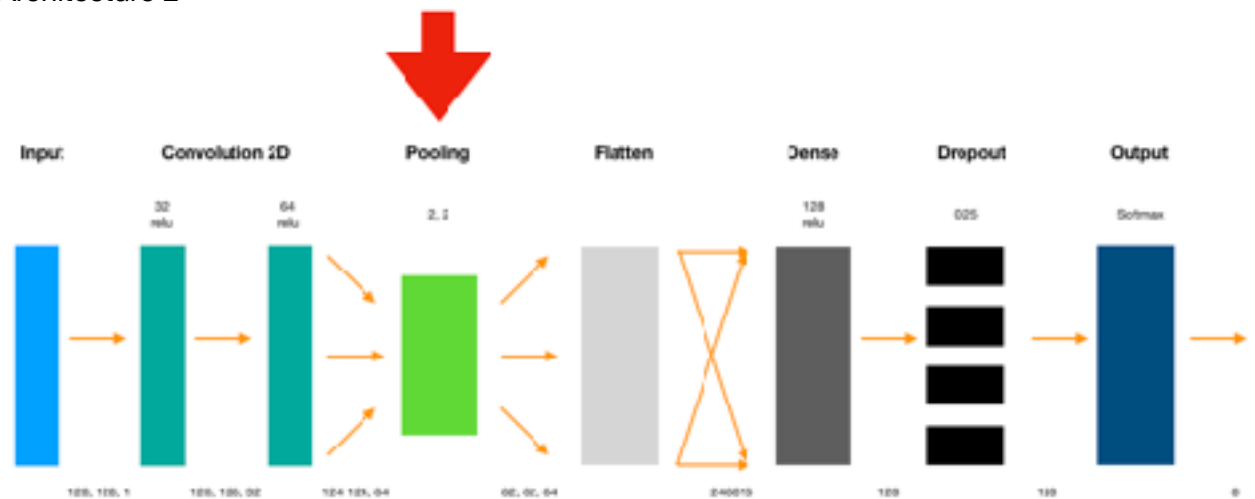
# Appendix

## Appendix 1: CNN Architectures

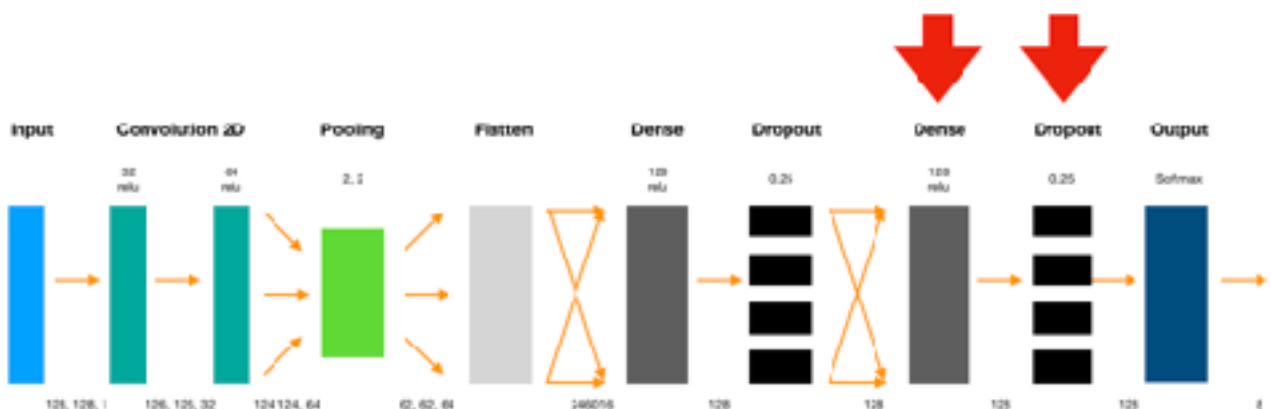
Architecture 1



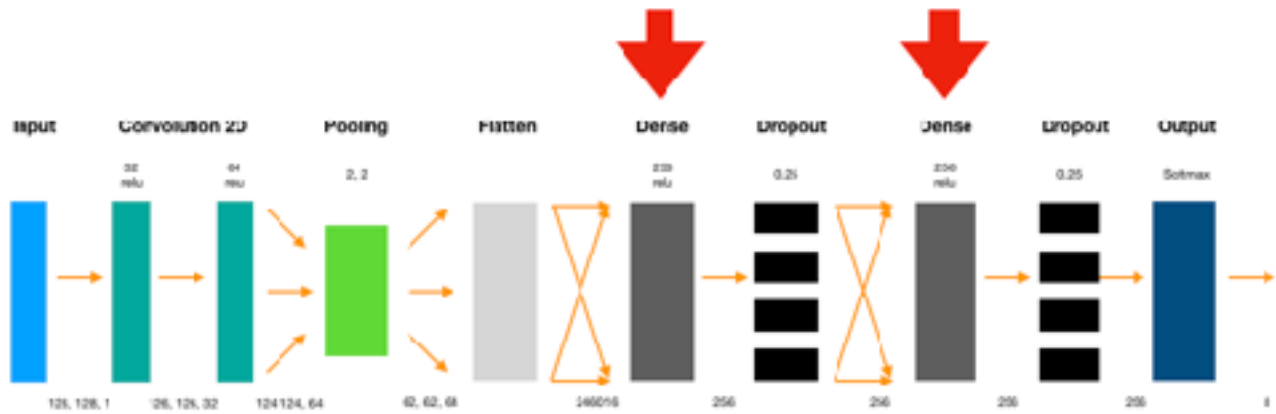
Architecture 2



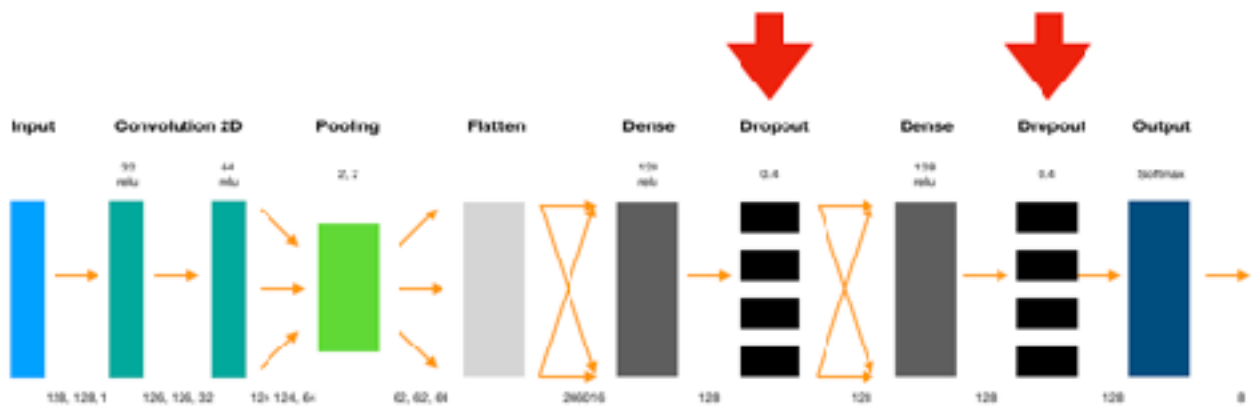
Architecture 3



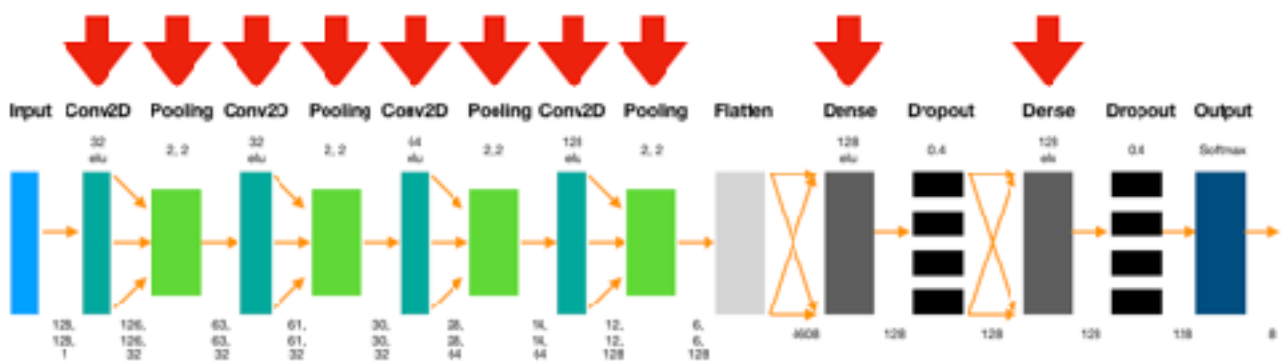
#### Architecture 4



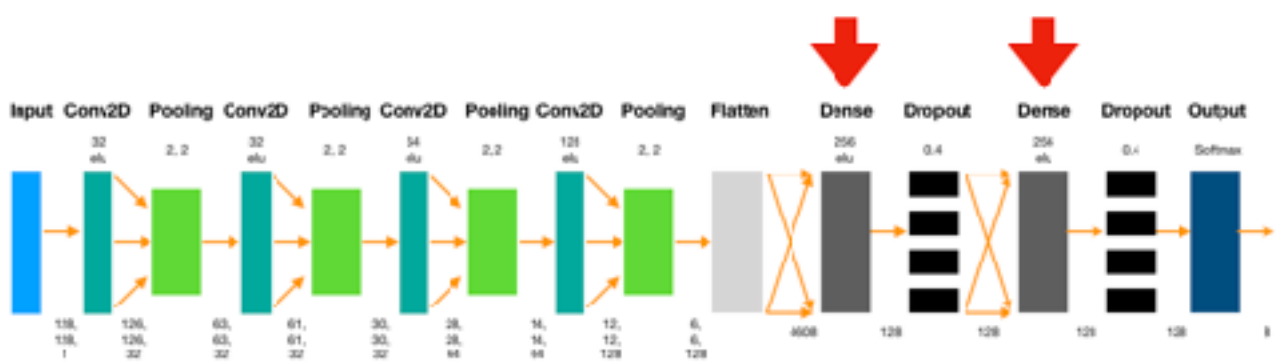
#### Architecture 5



#### Architecture 6

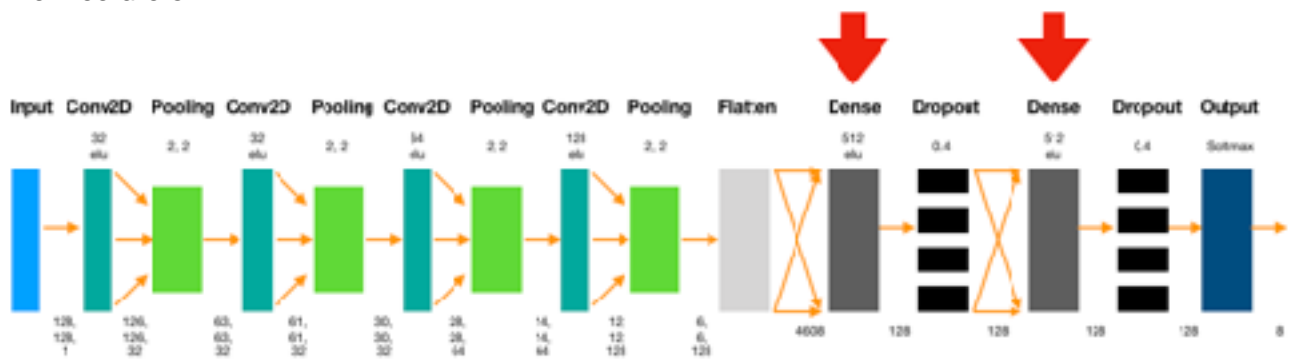


#### Architecture 7

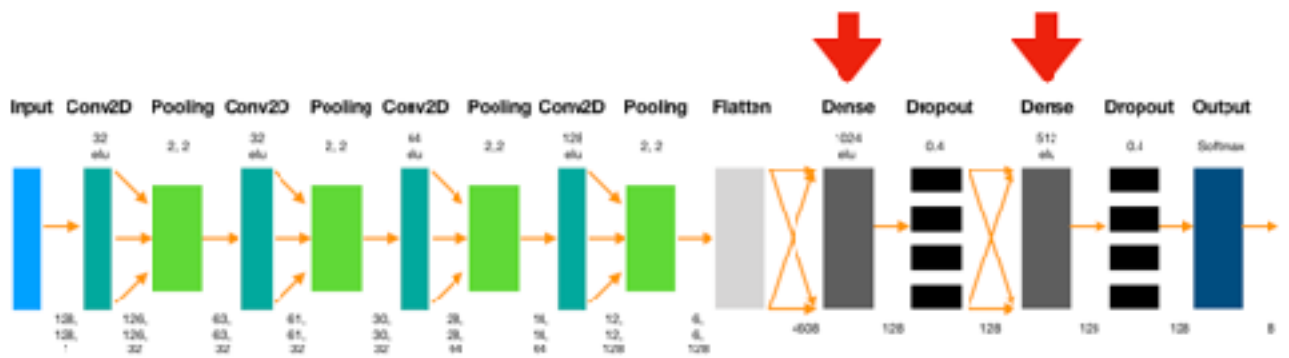




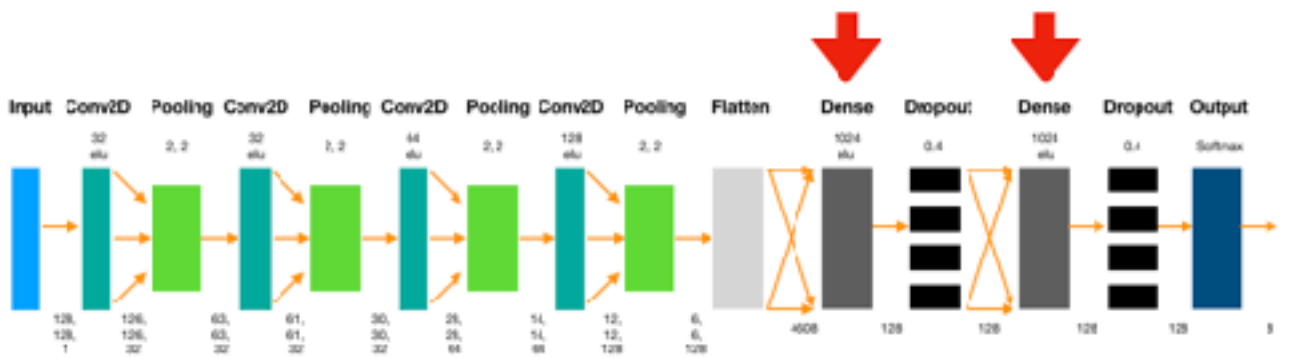
## Architecture 8



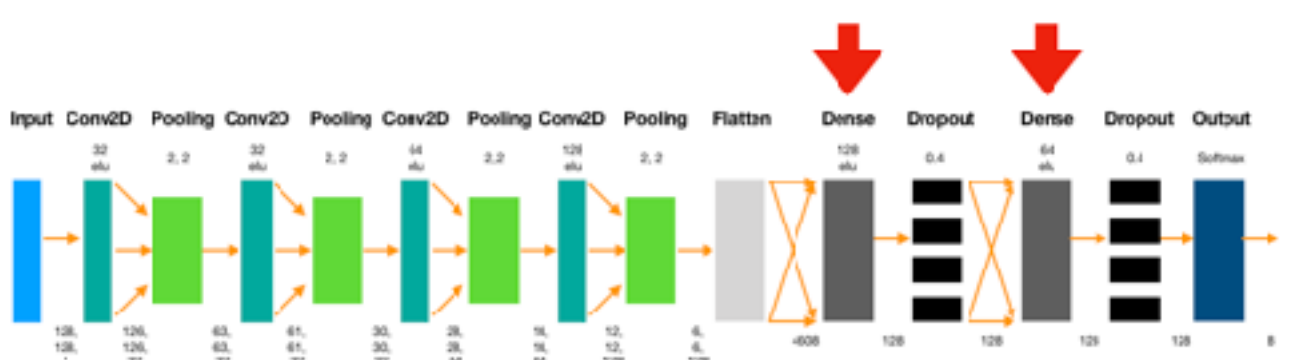
## Architecture 9



## Architecture 10



## Architecture 11



## Appendix 2: Confusion Matrices for classifiers 1-11

### Classifier 1

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	0	107	25	0	0	0	0
anger	0	0	81	22	0	0	0	0
contempt	0	0	5	5	0	0	0	0
disgust	0	0	47	17	0	0	0	0
fear	0	0	25	8	0	0	0	0
happiness	0	0	99	7	0	0	0	0
sadness	0	0	44	11	2	0	0	0
surprise	0	0	47	27	0	0	0	0

### Classifier 2

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	132	0	0	0	0	0	0	0
anger	103	0	0	0	0	0	0	0
contempt	10	0	0	0	0	0	0	0
disgust	64	0	0	0	0	0	0	0
fear	33	0	0	0	0	0	0	0
happiness	106	0	0	0	0	0	0	0
sadness	57	0	0	0	0	0	0	0
surprise	74	0	0	0	0	0	0	0

### Classifier 3

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	0	0	0	0	0	0	132
anger	0	0	0	0	0	0	0	103
contempt	0	0	0	0	0	0	0	10
disgust	0	0	0	0	0	0	0	64
fear	0	0	0	0	0	0	0	33
happiness	0	0	0	0	0	0	0	106
sadness	0	0	0	0	0	0	0	57
surprise	0	0	0	0	0	0	0	74

### Classifier 4

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	132	0	0	0	0	0	0
anger	0	103	0	0	0	0	0	0
contempt	0	10	0	0	0	0	0	0
disgust	0	64	0	0	0	0	0	0
fear	0	33	0	0	0	0	0	0
happiness	0	106	0	0	0	0	0	0
sadness	0	57	0	0	0	0	0	0
surprise	0	74	0	0	0	0	0	0

## Classifier 5

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	132	0	0	0	0	0	0
anger	0	103	0	0	0	0	0	0
contempt	0	10	0	0	0	0	0	0
disgust	0	64	0	0	0	0	0	0
fear	0	33	0	0	0	0	0	0
happiness	0	106	0	0	0	0	0	0
sadness	0	57	0	0	0	0	0	0
surprise	0	74	0	0	0	0	0	0

## Classifier 6

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	116	4	0	1	0	1	4	6
anger	37	50	0	7	0	0	7	2
contempt	4	0	1	0	3	2	0	0
disgust	7	0	0	57	0	0	0	0
fear	7	0	0	0	12	0	3	11
happiness	19	0	0	2	4	80	0	1
sadness	32	1	0	0	5	0	19	0
surprise	6	0	0	0	0	0	0	68

## Classifier 7

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	0	0	0	0	132	0	0
anger	0	0	0	0	0	103	0	0
contempt	0	0	0	0	0	10	0	0
disgust	0	0	0	0	0	64	0	0
fear	0	0	0	0	0	33	0	0
happiness	0	0	0	0	0	106	0	0
sadness	0	0	0	0	0	57	0	0
surprise	0	0	0	0	0	74	0	0

## Classifier 8

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	109	7	0	9	3	2	0	2
anger	29	62	0	10	0	0	2	0
contempt	8	1	1	0	0	0	0	0
disgust	10	0	0	54	0	0	0	0
fear	9	0	0	0	10	0	1	13
happiness	19	0	0	3	2	82	0	0
sadness	35	9	0	0	1	1	11	0
surprise	10	0	0	0	0	0	0	64

## Classifier 9

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	0	0	0	0	0	132	0	0
anger	0	0	0	0	0	103	0	0
contempt	0	0	0	0	0	10	0	0
disgust	0	0	0	0	0	64	0	0
fear	0	0	0	0	0	33	0	0
happiness	0	0	0	0	0	106	0	0
sadness	0	0	0	0	0	57	0	0
surprise	0	0	0	0	0	74	0	0

## Classifier 10

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	132	0	0	0	0	0	0	0
anger	103	0	0	0	0	0	0	0
contempt	10	0	0	0	0	0	0	0
disgust	64	0	0	0	0	0	0	0
fear	33	0	0	0	0	0	0	0
happiness	106	0	0	0	0	0	0	0
sadness	57	0	0	0	0	0	0	0
surprise	74	0	0	0	0	0	0	0

## Classifier 11

	neutral	anger	contempt	disgust	fear	happiness	sadness	surprise
neutral	121	2	0	2	1	4	0	2
anger	38	48	0	3	1	0	11	2
contempt	9	0	0	0	1	0	0	0
disgust	14	0	0	50	0	0	0	0
fear	8	0	0	0	6	0	1	18
happiness	20	0	0	0	0	86	0	0
sadness	23	4	0	3	8	0	15	4
surprise	4	0	0	1	0	0	0	69