המעבדה לעיבוד גיאומטרי של תמונות
Geometric Image Processing Laboratory

המרכז לדימות מוחי בילדים
Educational NeuroImaging Center

CHILD FACIAL EXPRESSION DETECTION

Eden Benhamou

Deborah Wolhandler

Supervisors:

Alon Zvirin

Michal Zivan

Spring 2018

Abstract

We present an examination of facial expression detection of children in two different study environments, joint dialogic reading and yoga.
We analyze videos of preschool children from the ENIC lab filmed during 6 months.
Our data analysis combines face detection algorithms, artificial neural networks designed for emotion recognition, face recognition algorithm and image processing tools for tracking.
We present results of child facial expressions during the recorded video sessions.
This project was made in collaboration of ENIC-GIP labs.

Table of Contents

Introduction

This project is part of a collaboration between the Geometric Image Processing lab (GIP) at the Computer Science Faculty and the Educational Neuroimaging Center lab (ENIC) at the Faculty of Education in Science and Technology, both in the Technion.

The aim of this project is to analyse children facial expressions in different study environments like yoga and Dialogic Reading. The facial expression of a child provides information about his emotional state during a lesson. The ENIC lab studies the relationships between emotional state and learning capacity.

We analyzed videos of children from the ENIC lab filmed in different study environments. The project goal is to create an algorithm that is able to detect emotion of children in a video, track the child and display the results as a function of time.

We use an emotion recognition network trained with adults dataset, then we trained it with other databases and compare the different models obtained.
We use those models on the videos from the ENIC Lab and use the temporal redundancy to create relevant output. We track children along the video using face recognition and image processing tools.
We provide quantitative measurements of children emotions, and present statistics for each child as recorded in the videos of the study sessions.

We've faced a lot of challenges during our work.
One of them was to deal with the differences of emotion classification between children and adults. For example, the "open mouth" symptom : most of the children open their mouths during neutral state. However, adults usually open their mouth only when they are surprised.
Another one was that children tend to move a lot and to touch their faces more often than adults.
Besides, some emotions are very similar, and it is difficult to differentiate them. For example, "angry" and "disgust" emotions were shown as hard to discern. In general, humans recognize emotions with 65% accuracy and use gesture, position, context.
Additionally, the children emotion databases are relatively rare and small compared to the adults ones.
Finally, the videos were filmed before our project was considered, and although intended for studying child behavior, were not designed for our analysis. Thus, we get videos in a low quality that hard to deal with.

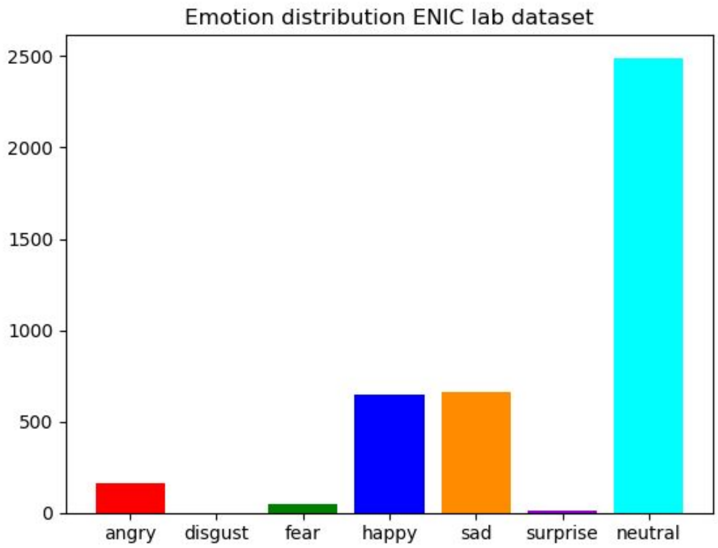
During our work, we've faced these challenges and tried to find the best solutions to achieve our goals.

Project data

Children Dataset

Original videos data from ENIC lab - Technion

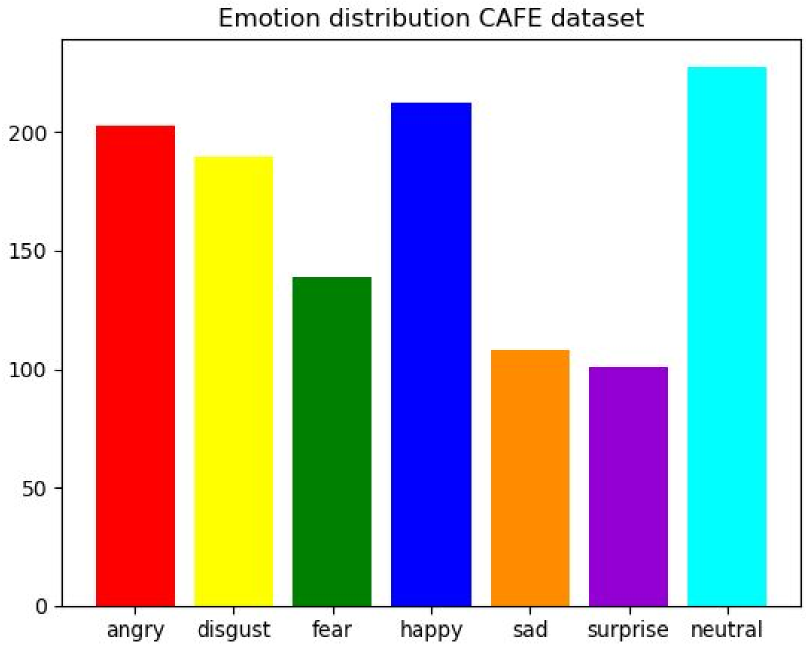
The data consists of 255 videos of preschool children filmed in two different study modes: joint Dialogic Reading (DR) and Yoga. The videos record groups of approximately 12 children each and lasts ~12 minutes.
We used an open-source algorithm of face detection. The resolution of the bounding boxes detected are between 48*48 and 100*100 pixels (approximately).
We classified the facial expressions into seven emotions categories (Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral).
A part of the detected faces was labelled by the ENIC team members. We use it to train our model in emotion detection (See more information in the training part).
The emotion distribution in the labelled dataset is:



The Child Affective Facial (CAFE) Set - Databrary:

This dataset consists in photos taken of 2- to 8-year-old children posing for 6 emotional facial expressions - sadness, happiness, surprise, anger, disgust, and fear - plus a neutral face.
The full set features 90 female models and 64 male models (27 African American, 16 Asian, 77 Caucasian/European American, 23 Latino, and 11 South Asian). With the exception of surprise, children were asked to pose for each expression with their mouths open and with their mouths closed. Surprised faces were only posed with their mouths open. Open mouth disgusted faces generally included a tongue protrusion. In total, we had 154 child-models pose each of these 7 expressions. Not all children were able to successfully pose for all 7 expressions, so all unsuccessful attempts were eliminated from the set.
The result is **1192** total photographs.
But why the open mouth is so interesting in emotion recognition ?
Most of the children are opening mouth in neutral state. However, adults usually open mouth when they are surprised only. Thus, this dataset is a huge improve against dataset of adults emotions.

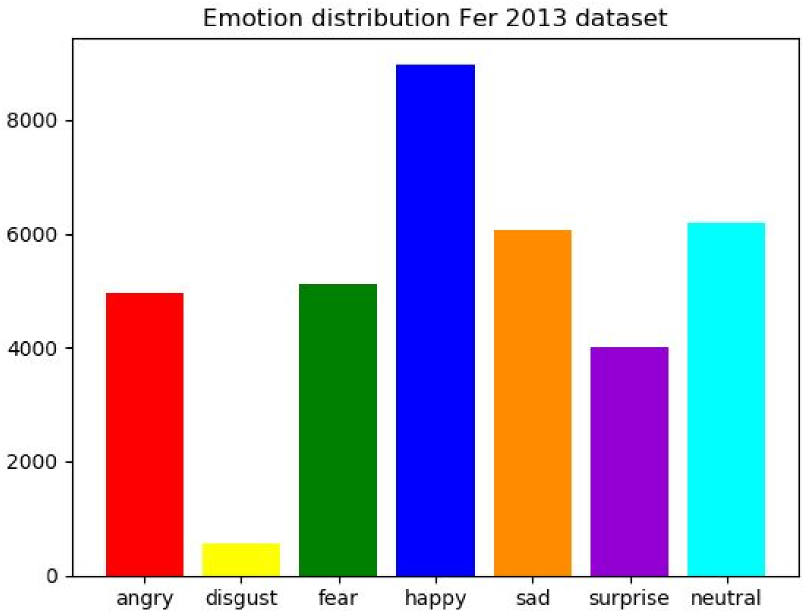
We’ve trained our model with this dataset. The model we get after training was worse than the previous model trained with fer2013 only.
We can explain those results by the fact that the pictures of CAFE show forced emotions and don’t reflect perfectly the reality (See more information in the training part).



Adults Dataset

FER2013

The data consists of 48x48 pixel grayscale images of faces.
The face images are categorized in one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).
The training set consists of 28,709 examples.
The public test set used for the leaderboard consists of 3,589 examples.
The final test set, which was used to determine the winner of the competition, consists of another 3,589 examples.
Due to label noise, the human accuracy on this data is 68%.



This dataset was originally used by the detection algorithm we found for training (See more information in the model description). The results we found with the model trained only with fer2013 were quite satisfying. But, we decide to try to improve them by training with others datasets.

Work steps

Theoretical research

When we start working on our project, we had to learn a lot of concepts about image processing and deep learning. We also had to learn about the main tools we would need to work with: opencv, haarcascades, tensorflow, keras… Additionally, we had to get familiar with our working environnements: we choose to use Pycharm as IDE for our python project and work with Anaconda3 as virtual environment. After getting familiar with the concepts and the tools, we tried some algorithms in opencv to recognize faces in pictures and also in videos. This was our first result of face recognition in video:

CNN research

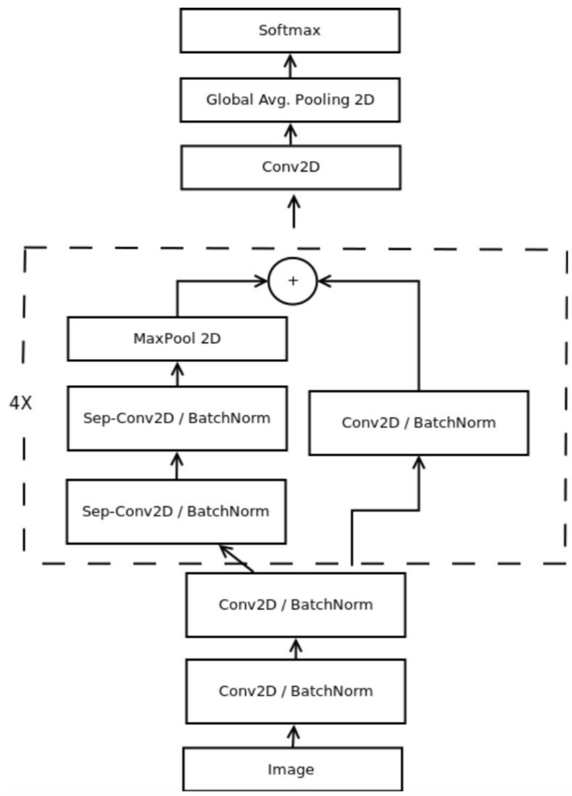
We spend a long time on looking for an existing network of emotion recognition. Our plan was to find a network for testing results in a first time, without training. We found a lot of non trained networks. Occasionally, we found trained networks and tested them with labelled pictures of children that we found on internet. The experimental results were not enough satisfying, thus we continued our research. Finally, we found a performing network that detect faces and recognize emotions in videos: the face detection works with a model of haarcascades and the emotion detection works with a model trained on fer2013 database. We tried it on several examples of videos and it recognized pretty good the faces but not always the right emotion. Usually, happy emotion is recognized as neutral or sad as neutral, not surprising errors. We also tried it on a dataset of children pictures in good resolution and obtained pretty good results: 98% for happy, 55% for sad, 78% for neutral, 42% for angry, 35% for fear.

Emotion recognition

Model

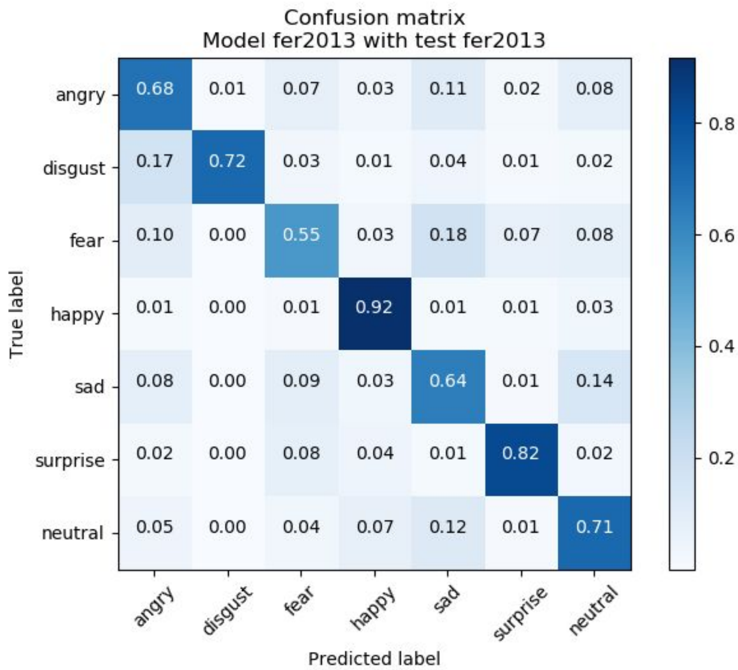
For our emotion recognition algorithm, we used a CNN model named "*mini-Xception*", inspired by the Xception architecture of Francois Chollet, which is available on keras, with weights trained on ImageNet.
Xception proposes a deep with depthwise separable convolutions architecture.

Depthwise separable convolutions perform first a convolution over each channel of an input layer, and then a 1 by 1 convolution which combines the different output channels to a single one. This method allows significant reduction of parameter numbers.
The Xception Architecture is made of 36 convolutional layers, structured into 14 components.
Those residual modules make this architecture very easy to adapt.
Moreover, there is no fully connected layer, unlike most of the CNN architectures. This specificity also contributes to the relatively small number of parameters.
According to the paper related to mini-Xception architecture *"Real-time Convolutional Neural Networks for Emotion and Gender Classification"*, this cut in the number of parameters helps to speed up the algorithm running time, and also provides a better generalization (too high number of parameters compare to the training set size can induce overfitting).
Mini-Xception was trained with Adam optimizer.



Mini-Xception model schema

The input of the network is an image with the following format: 64*64*1.
It outputs a probability for each of the 7 following emotions : anger, sad, happy, disgust, fear, surprise and neutral.
The model was pretrained with the FER2013 training dataset.
The results on the training set are the following:

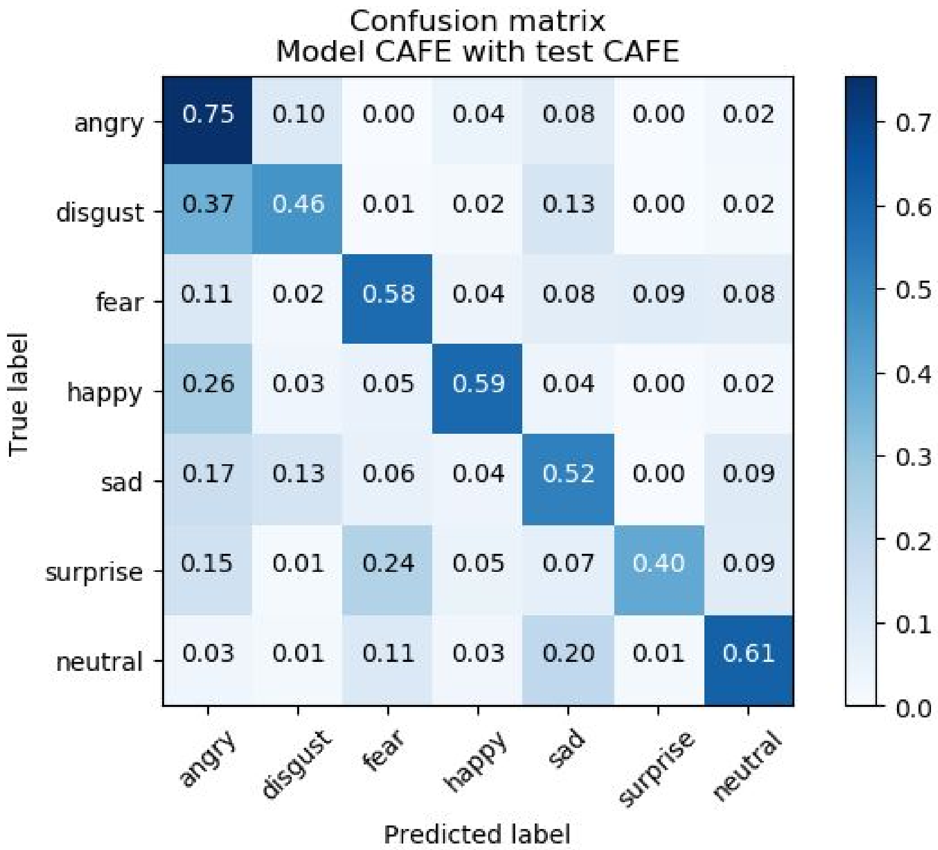


We can observe several common misclassifications such as predicting *“sad”* instead of *“fear”* and predicting *“angry”* instead *“disgust”*.

Training for Emotion recognition

CAFE

We found this dataset on Internet. It contains children pictures posing for 6 emotional facial expressions - sadness, happiness, surprise, anger, disgust, and fear - plus a neutral face (Please see more information at CAFE).
We’ve created a load function for the dataset and use it as input for training our model. We decided to train an existing model previously trained with fer2013, then with CAFE.
The results on the training set can be found in the following confusion matrix:



We notice some recurrent mistakes, such as predict angry instead of disgust, or sad instead if neutral. We can see in the following examples of prediction mistakes examples, that those emotions are indeed hard to differentiate.

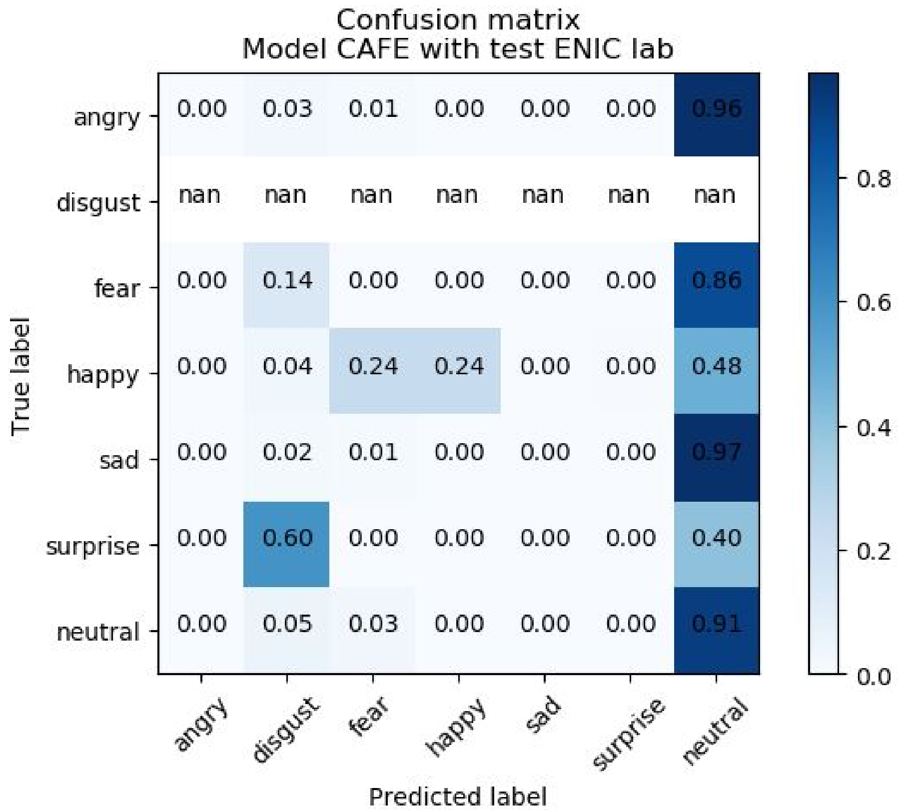
True label: Disgust
Predicted: Angry

True label: Happy
Predicted: Angry

True label: Surprise
Predicted: Fear

True label: Neutral
Predicted: Sad

We tested the model on the dataset created from the labelled data of ENIC lab, and observed that the results were worse than those with the previous model (trained with fer2013 only).



We can explain those results by the fact that the pictures of CAFE show forced emotions and don't reflect the reality. Moreover, the pictures of the CAFE dataset are only frontal ones, and do not contain any faces images with obstructions.

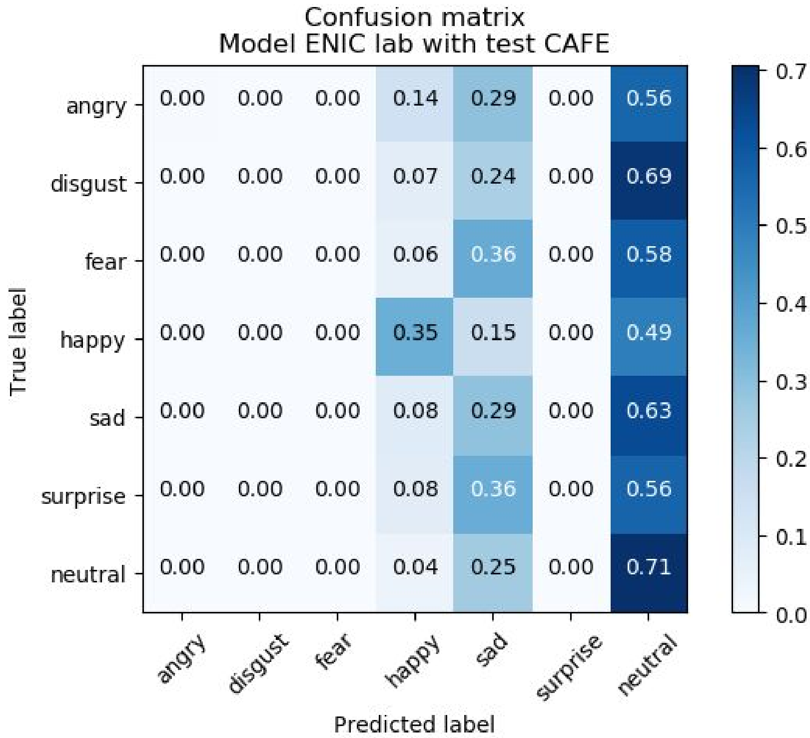
Our data

During our work, we've prepared bounding boxes pictures for emotion labelling by the ENIC lab. After we get the labels, we've created a load function for the dataset. We've used this dataset for testing our different models and compare results (for example, when we compare the model trained with CAFE and the model trained with fer2013 only, see CAFE for more information).

Later, we used the data for training. We can find below a confusion matrix of the model created and tested on CAFE. We can see that all the pictures in the test set has been classified as happy, sad or neutral. These results are due to the fact that the training database has been classified in these three categories mostly.

Because of these bad results, we decided to not use this prediction model in our final algorithm.

We display in the following confusion matrix the results of the emotion recognition model trained with the ENIC dataset and tested on CAFE



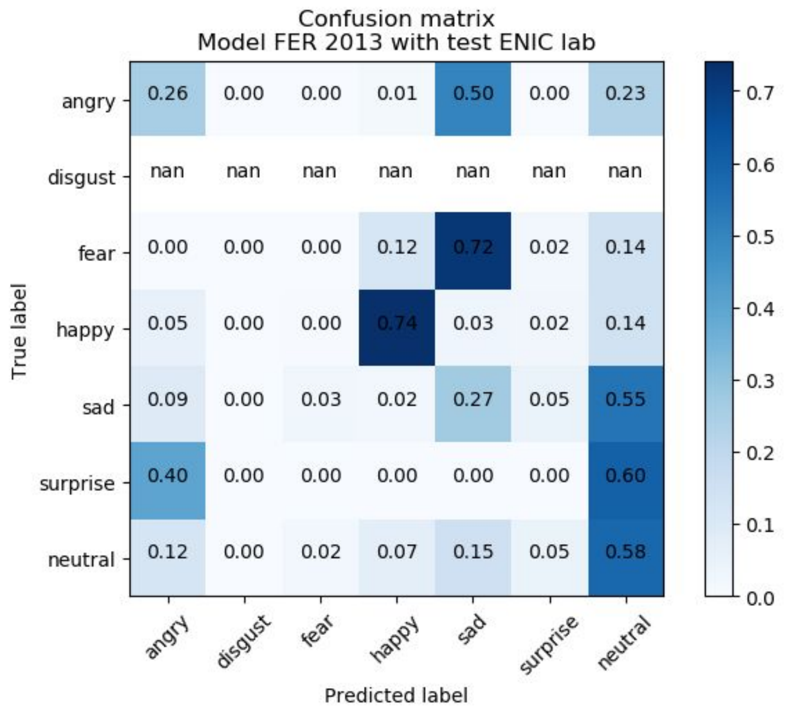
We can explain the important prediction bias to the happy, sad and neutral classes, by the emotion distribution of the ENIC dataset. Indeed, the training database has been classified in these three categories mostly, and thus the model tends to predict essentially those classes over the 7 ones.

Parameters tuning

We tried to find optimal optimization parameters for training the emotion recognition model with our data.
We tried different optimizers, such as SGD, Adagrad, Adam, and we run the training for different batch size : 4, 16, 32, and checked some learning rate values.
We loaded the pretrained weights that were in the code repos, and freeze 38 or 42 layers. Indeed, we assume (according to theory of deep learning) that in the first layers, we've learned general parameters, which are not influenced by differences between datasets. It allows to get rapid progress and to be able to continue training instead of start training from scratch.
Finally, we get that the optimal parameters were training with Adam, with a learning rate of 0.025 with batch size of 32 and with 38 freezed layers.

The chosen model

We've concluded that the best model is the one trained with fer2013 only. Indeed, this dataset is very large (~ 30 000 pictures) and every other dataset we could use was at least 15 times smaller. Thus, the training with those small dataset create overfitting. For example, the pictures in CAFE don't reflect the reality and thus it was harder to get a good generalization. In the same way, the labelled data we've get from the ENIC lab was small and unbalanced in the emotions categories.
We can find below the confusion matrix of the model trained with fer2013 only tested on the the labelled data we've get from the ENIC lab.



We get better results with this model than with the model trained with CAFE or the model trained with ENIC lab data. We can explain those results by the fact that fer2013 contains more realistic faces images than CAFE (for example: profil images, hands on face...).
In conclusion, the model trained with fer2013 only is the most stable and gives the best results.

Tracking

One of the main challenge in our project is that we need to track children in the videos. Indeed, the network we found recognize faces and emotion for each bounding box per frame in the video. It doesn't make relations between children over frames.
We choose to create by ourselves a tracking algorithm and data structures for saving bounding boxes along the video. In a first time, we supposed that if a bounding box (BB) was close enough to another one in the previous frame, it was the same child.
To evaluate the distance between BB, we find the center coordinates of the BB and calculate the euclidean distance for each BB in the current frame with the BBs of the previous frame. After finding the nearest BB, we fix a maximal authorized distance (named *max_dist*) between BB of two consecutives frames. If the authorized distance is exceeded, we consider that we've discovered a new child. Otherwise, we've recognized the child, so we add his BB to the BB list of the corresponding child.

Later, we decided to authorize a frames threshold (named *frames_threshold*) between the current child and the nearest child founded from the previous frames. Its allows to continue the child tracking even if we lose him for *frames_threshold* frames.

How we fixed the parameters of *max_dist* and *frames_threshold* ?
We made a lot of tests. For *max_dist*, we've printed the distance values between current and nearest bounding boxes and tried to find the best parameter that never wrong about the similarity of two children. Thereby, the algorithm can give a "false negative" result, meaning the result is that two children are different while they are similar, but never gives a "false positive" result, meaning two children are similar while they are different.
In the same way, we made similar tests for *frames_threshold* and founded the best value that never gives "false positive" results.

Face recognition

After we get an end-to-end module, we wanted to improve our algorithm. Indeed, the children tracking with the distance between BB gives very fast results but had the inconvenient of finding too much children in the video. In many cases, the algorithm gives that the new BB is a new child while it corresponds to an existing child.
That is why we choose to use a face recognition algorithm. We founded an algorithm of Adam geitgey on github (see References).
This algorithm needs a reference directory with directories of pictures for each person we want to recognize (see Appendix). It works with a KNN algorithm. It had to be trained with those pictures. With the trained model, it can recognize new pictures.

Semi-automatic

In first place, we tried to create a reference directory with pictures we get from the ENIC lab. Those was high resolution pictures of some children from the filmed classes. The problem was that not all of the children were in the pictures and we get rather bad results.

After this experience, we thought about taking some bounding boxes saved in a previous run of a video and use them as references for face recognition.
In case of the face recognition algorithm returns “unknown” about a child, we decided to use our previous method that find the nearest child in the previous frames. In this way, we maximize the number of good recognition.
The results were better. The advantage was that the training was pretty fast because the set of selected pictures was small. But the inconvenience was that we needed to manually add pictures to the directories for each new video. This is not a user friendly method.
This why we came to the automatic tracking method.

Automatic

In this solution, we decided to automatically add new bounding boxes to the reference directory. That means that every time we add a new picture, we needed to retrain our model with the new set of pictures.
<u>Our algorithm:</u> The face recognition algorithm returns the id of the recognized child. Sometimes, it returns “unknown”. In this case, we are looking for the nearest child in the previous frames. If the maximal distance is not exceeded, we chose to add the image to the child directory in order to improve the face recognition of this child for next time. Else, we consider that a new child is discovered, so we create a new directory containing the single image in the reference directory of face recognition.
The results of children recognition were better than in the original algorithm and pretty much the same as in the semi-automatic (but this time the solution was totally user friendly, with no need to provide prepared pictures for the run).
For example, for 1200 frames from a video of the dataset, we recognized 96% of the children. With the face recognition algorithm, we recognize more than 87% of the BB as known children. Then, with the distance algorithm, we recognize 70% of the unknown BB left.
But one of the inconvenience was that we needed to train again after each image insertion. Additionally, the more photos in the set, the more the training takes time.
Another inconvenience was that we are not able to check the quality of the added BB: Is it frontal face ? Is the face hidden ? And thus, we can’t be sure that the image insertion would improve the face recognition algorithm or not.

Run time improvement

After getting better results in children recognition, we wanted to improve the run time of the algorithm of face recognition. In fact, the children recognition with the automatic algorithm was 20% better than with the original algorithm but the run time was 30 times longer !

In first step, we’ve tried to change the minimal distance in the KNN algorithm. This parameter had a huge influence on the running time. For example, changing between 0.4 to 0.5 for the euclidean distance gives a run 2 times shorter. But the algorithm gives a lot of false positive (a child is recognized as another child) results (we remark this by checking output directories of bounding boxes) and we try to avoid these errors as much as possible. Finally, we decided on a minimal distance of 0.4 because it was the maximal distance that doesn’t give false positive results in our experiments.

We use knn algorithm with k equals to three, in order to be more resistant to noise.

In second step, we tried to limit the number of authorized pictures in the children directories for face recognition. We've thought that limiting the number of pictures would reduce the training time. But we founded that with an unlimited number of pictures, the quantity of recognized children was higher than in the limited version. We can conclude that the improvement of the recognition (using more pictures) is good enough to balance the fact that the training was longer.

Saving sequences

After understanding that the output results can be unstable during the emotion recognition (for example, getting one frame with sad emotion in a sequence of frames with happy emotion), we decide to look for sequences of bounding boxes (BB) with the **same emotion.** When we found a new BB for a child, we compared the emotion of the new BB and the BB sequence:

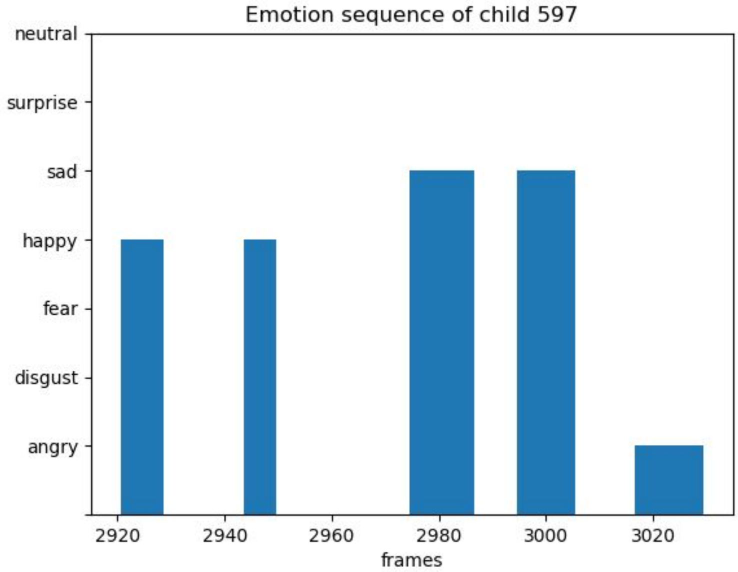
- If the emotion is the same, we add the BB to the BB list.
- Else, we save the sequence of type *BBEmotionSequence* in the *childrenSequenceDict* dictionary with *children_id* as a key (See more information at Data structures) and initialize the new list to contain the new BB.

Later, we decide to fix a minimal sequence length (named *seq_min_len*) that shows relevant results. This size is a parameter. Usually, we used *seq_min_len*=6 because this size shows relevant results.

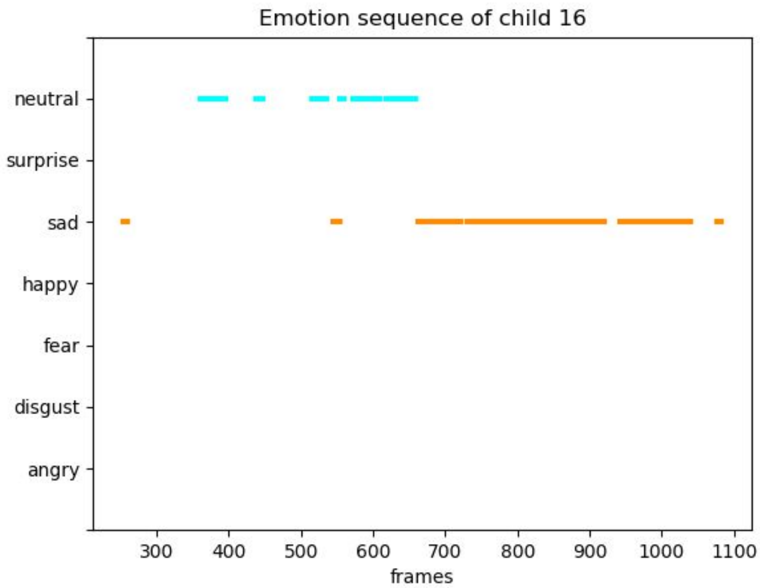
When the algorithm finish to run on all the video frames, we save the BB images in output directories. The root is *sequences/video_name*. For each child that we saved sequences for, a directory is created (named *child_id*). For each sequence of BB, a directory with the limits frame num and emotion is created (named *frames_min-max_emotion*). In those directories, we saved the BB images.

Graphs

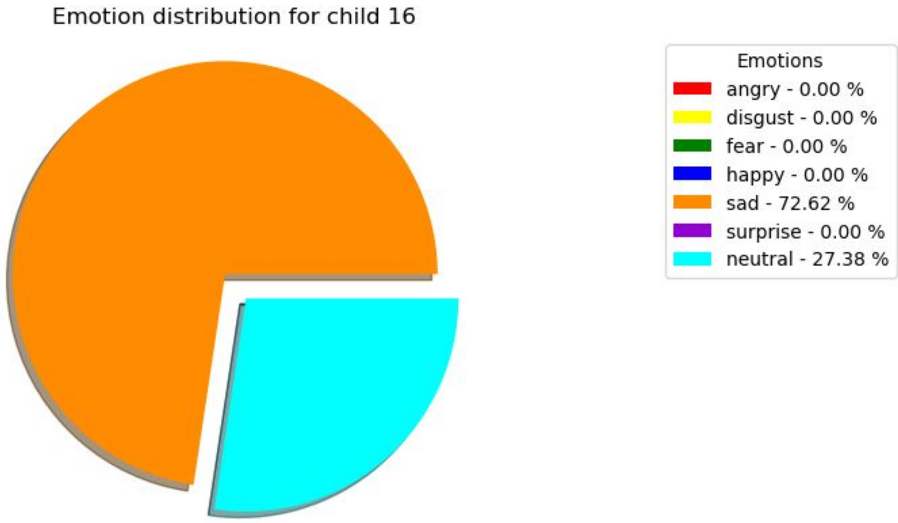
One of the main goals in our project is to display the results in a meaningful way. Thus, we created graph of emotions as function of time for each child.
The graph data is based on our saved sequences for each child.
At first, we created histogram graphs that look like the following graph:



Secondly, we decided to change the plotting display and create graphs that look like the following:



We also create graph with percentages of emotions for a child:



Conclusion and next steps

Conclusion

The project goal is to create an algorithm that is able to detect emotion of children in a video, tracking after the children in the video and display the results for each children in function of time. Finally, we've managed to achieve this goal.
We've succeeded to recognize children in the video with 90% accuracy. The emotion recognition success is hard to measure because it's very subjective. Indeed, even a human recognize emotion in 70% accuracy and don't get only a bounding box to discern emotions. He also consider the context, and other cues like the sound).
Additionally, some of the emotions are similar and hard to differentiate. For example, the labelled data we get from the ENIC lab was pretty small and was labelled by only one person so its reliability was limited.
Globally, the emotion recognition is performant given the video quality we work with.
But we have some ideas on possible improvement solutions.

Next Steps

- Training

We could have looking for more database and maybe found one that really improves our model. Maybe using others adults databases could also improve the model after training.
We could also prepare balanced labelled dataset of emotions pictures with several people to label it, in order to get a more reliable data.

- Filter emotion

We've tried to reduce the number of emotions in the output. The difference was only in the results display that was more homogeneous. We tried to change the number of emotions for the training but it required too many modifications in our code. So a next step for improvement would be to reduce emotion classes. For example, merging between angry and disgust. According to the paper *Anger and disgust: Discrete or overlapping categories* (see reference 7) , those emotions are hard to differentiate. We think that this change should improve the results quality.

- Tracking

We've spend a long time testing other algorithms of face recognition and changing our tracking algorithm. We get good results but we expect that using deep learning based algorithms would improve our tracking.
Another idea would be to pass over the children directories at end of run in order to merge corresponding children. Indeed, at the end of the process, the directories contain many pictures of each child, and consequently the recognition model is expected to be more accurate and more robust to noise.

Appendix

Data structures

Bounding Box

We create a Bounding Box class in which we save all the parameters needed.
When we detect a face in the video, we create a new Bounding Box for the relevant child.

Fields

- face coordinates
- center coordinates
- frame number
- emotion label
- emotion probability
- child id
- image

when:
emotion_label, emotion_proba = detect_emotion_for_face(gray_image, coordinates)

Child Dictionary

Dictionary of list of BB that we save along the video process.
The keys are the children ids. We add BB to the relevant child (according to the recognition algorithm), or if we do not success to recognize the child, we add a new child to the dictionary

Frame

Data structure that stores data about the current frame. We detect faces and emotions once per frame, and then process this information BB per BB.

Fields

- Bounding Box Dictionary: we create a dictionary in which we save all the bounding box we detect in the current frame.
- points: list of the centers points of BB in the Bounding Box Dictionary
- frame number

Bounding Box Emotion Sequence

Data structure in which we save a list of BB for a specific child when we detect a series of same emotion.

Fields

- emotion label
- Bounding Box List: List of bounding box with same emotion
- starting frame
- ending frame
- emotion probability : the mean of the probability emotion of each bounding box in the sequence

Children Sequences Dictionary

Dictionary of BB Emotion sequences that we save along the video process.
The keys are the children ids.

User Guide

Modes

Original

Running the algorithm in this mode uses our original tracking algorithm.

command

detect_emotion_children_videos.py video_name -orig
or
detect_emotion_children_videos.py video_name --original_face_rec

Manual

This mode demands to pass the pass of a reference directory (see example of a children images directory in the appendix). In this mode, we provide to the recognition algorithm all the images for training and we do not add any images to the directory. Thus we don't need to train again.

command

detect_emotion_children_videos.py video_name -manual -child_dir child_dir_path

Semi Automatic

This mode demands to pass the path of a reference directory (see example of a children images directory in the appendix). In this mode, we provide to the recognition algorithm images to the initial training, but during the running, we add children images to the children directories to improve the recognition results.

command

detect_emotion_children_videos.py video_name semi_auto -child_dir child_dir_path
or
detect_emotion_children_videos.py video_name semi_auto_face_rec -child_dir child_dir_path

Automatic

In this mode there is no need to pass any directory. This is our user friendly mode(See more information here).

command

detect_emotion_children_videos.py video_name

Additional Options

Start

Choose the beginning minute of the video for which we want to start to process the video.
We expect to get a float value.
If we do not specify this parameter, the default value is to start from the first frame.

command

detect_emotion_children_videos.py video_name -start X …

Time Process

Choose the length in minutes we want to process the video.
We expect to get a float value.
If we do not specify this parameter, the default value is 1 minute.

command

detect_emotion_children_videos.py video_name -time Y …

Process entire video

To process the all video from the start minute we've fixed (by default it will process simply the all video).

command

detect_emotion_children_videos.py video_name -all …

Minimum sequence length

Choose the minimum sequence length. By default, is equal to 6. We expect to get an integer.

command

detect_emotion_children_videos.py video_name --seq_min_size Z

Children Image Directory structure

Example of directory

$ tree data/images/children_face_recognition/video_name

```
1  # child 1
        ----1.jpg
        ----2.jpg
        . . .
        ----m.jpg

. . .
1  # child k
        ----1.jpg
        ----2.jpg
        . . .
        ----m.jpg
```

References

1. Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE, 2001.
2. Adam Geitgey, “Machine learning is Fun!”, https://medium.com/@ageitgey/machine-learning-is-fun-80ea3ec3c471
3. Arriaga, O., Valdenegro-Toro, M., & Plöger, P. (2017). Real-time Convolutional Neural Networks for Emotion and Gender Classification. *arXiv preprint arXiv:1710.07557*.
4. Chollet, F. (2016). Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*.
5. LoBue, V., Baker, L., & Thrasher, C. (in press). Through the eyes of a child: Preschoolers’ identification of emotional expressions from the Child Affective Facial Expression (CAFE) Set. Cognition & Emotion.
6. LoBue, V., & Thrasher, C. (2015). The Child Affective Facial Expression (CAFE) set: validity and reliability from untrained adults. Emotion Science, 5, 1532. http://doi.org/10.3389/fpsyg.2014.01532
7. Widen, S. C., Russell, J. A., & Brooks, A. (2004, May). Anger and disgust: Discrete or overlapping categories. In *2004 APS Annual Convention, Boston College, Chicago, IL*.