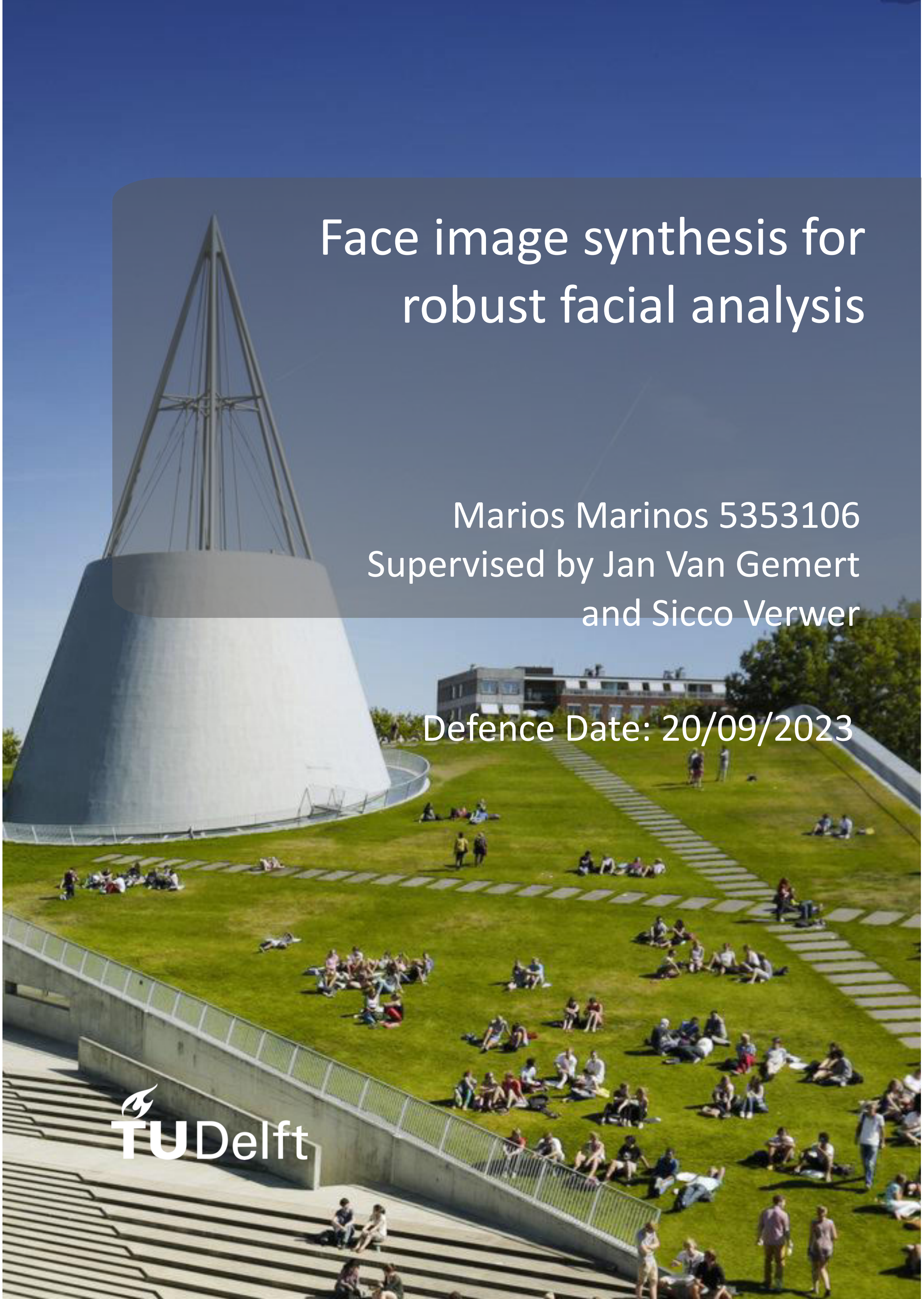


Face image synthesis for robust facial analysis

Marios Marinos 5353106
Supervised by Jan Van Gemert
and Sicco Verwer

Defence Date: 20/09/2023

TU Delft

Marios Marinos
TU Delft

October 17, 2023

Abstract

Emotion recognition is a challenging problem in the field of computer vision. The automatic classification of emotions using facial expressions is a promising approach to understand human behavior in various applications such as marketing, health, and education. Yet, recognizing some emotions, such as anger, jealousy, contempt, and disgust, is more challenging than others due to their subtlety and rarity in the training data. Self-pseudo labelled data uses an existing classifier to label unlabelled data and then make use of them. In this paper, we endeavor to investigate whether employing (self)pseudo-labelled data to train an Expression Manipulator [10] generator, with the purpose of generating a training dataset for training a classifier offers better balance between the Action Units, and thus, better performance. We aim to explore whether this approach surpasses the method of directly utilizing an equivalent quantity of (self)pseudo-labelled data to train the classifier[11]. Specifically, we focus on augmenting the Action Units (AUs) of facial expressions, which are the basic units of facial movement that correspond to specific emotions. The experimental results demonstrate that while pseudo-labelled data does improve accuracy over baseline supervised training, EM-augmented training does not yield any additional improvement when it comes to toy data set, whereas in the faces EM-augmented data outperforms pseudo-labelled data. We think this happen due to the simplicity of the problem in the toy data set whereas in faces we have more diversity using AffectNet [14] for both pseudo-labelling and EM.

1 Introduction

Automatic facial emotion estimation is important because it enhances human-computer interaction by enabling computer to understand human emotions easier, it contributes to healthcare and well-being by understanding whether a person is depressed or angry and foreseen to stop it progress further, it can also be used to enhance security and surveillance systems by being able to identify strange behaviours (anger,stress) and prevent something unexpected in public areas. Therefore, these systems are a valuable asset in various fields.

Facial expressions convey emotional states, behavioral intentions, and physical state [18]. To be able to decode these expressions the Facial Action Coding System (FACS) is used [5]. FACS decompose the expressions into action units (AUs) where combinations represent all possible facial expressions. There has been a lot of advancements recently in the field of action unit detection, but there are some problems that require solving.

Firstly there is a major problem that AUs occurring in existing facial datasets are often correlated with each other and do not cover all combinations of AU activations. That makes it hard for the AU classifier to be able to identify **independent** AUs. This is because natural human facial expressions use certain combinations of AUs and therefore these cooccur. In addition, some AUs exist rarely whereas some AUs occur often, which leads to an imbalanced learning problem. Imbalanced datasets pose significant challenges. When one class vastly outnumbers the others, models tend to favor the majority class, minority class patterns may go unnoticed, leading to biased predictions and poor generalization.

Secondly, a vast amount of unlabelled data exists that can be used, but to be able to annotate AUs you should be officially certified [4]. In addition, AU annotation is a time consuming and tedious process, can take several minutes per image, which makes manual annotation expensive and slow [6].

In this paper, we study how we make the most out of the available unlabelled data to reduce the need for expensive AU annotations by studying different approaches that exist to augment AUs on faces to de-correlate AUs on a toy data set and then on faces. We compare self pseudo-labeling and Expression Manipulator when it comes to labeling AUs on faces and toy data set. Finally, we test how advantageous is to manipulate AUs to increase the training data set and therefore improve model's F1-score performance.

2 Related Work

Limited AU annotations is a major problem. While there are a vast amount of data available through publicly available data sets such as AffectNet [14] high quality AU annotation is missing.

To alleviate the problem, may approaches have been tried such as weakly-supervised, semi-supervised and self-supervised. Weakly-supervised tries to make use of inaccurate or inexact annotations. Zhao et al., in his research [22] proposed a weakly supervised clustering technique that utilizes a large set of free web images that have inexact annotations. The weak annotations were from pre-trained models or query strings. In other fashion, Zhang et al. utilize expression-dependent and expression-independent joint AU probabilities as prior knowledge and were able to detect AUs without any annotation [20].

Semi-supervised approaches uses both annotated and unannotated data. They aim to make use of the vast amount of the unlabeled that exists publicly with the assumption that they follow the same distribution with the labelled data [22]. Semi-supervised using pseudo-labelling is another approach used for AU annotating. Loog argues that semi-supervised classification using pseudo-labelling estimates are never worse than the supervised solution in terms of the log-likelihood [11]. We test the application of pseudo-labelling to both toy and real data set to evaluate the F1-score performance and the ability of the approach to de-correlate AUs.

Recently, GAN-based and auto-encoders have received huge attention that make use of deep learning to alter facial expressions. Y Choi et al. proposed StarGAN a novel approach that perform image-to-image translation for multiple domain with a single model and they are able to transfer a facial attribute and expression from one subject to another [1]. The limitation there was that it could only generate a discrete number of expressions that heavily depends on the content of the dataset [16]. That is why Pumarola et al. proposed GANimation where the approach tries to address that limitation. GANimation allows the control of magnitude of activation per AU and it also deploys a fully unsupervised strategy to train the model as it only requires images annotated with the activated AUs. GANimation was a very important breakthrough but it suffered from changing condition-irrelevant regions, and was extended by Expression Manipulator [10]. Ling et al. in their work, they make use of relative action units to solve that and the generator learns to only transform the regions of interest which are the activated AUs [10]. Finally, an alternative to using GANs are Additive Focal Variational Auto-encoder (AF-VAE) an approach that arbitrarily manipulates high-resolution face images using a simple yet effective model and only weak supervision of reconstruction [17]. In our research, we make use of the Expression Manipulator to evaluate how F1-score performance and AUs de-correlation is affected by GAN-based approaches.

For the problem of AUs rarity, many up sampling techniques have been used on the minority AUs [21] or under sampling the majority AUs [2]. However, re-sampling is happening inside the dataset, which do not contribute additional information for the infrequent AUs. In our research, we make use of up-sampling the toy data set and again evaluate in terms of F1-performance and de-correlation.

3 Method

In this section we'll go through the 3 methods used to test which augmentation approach yields the best results in terms of F1-score performance and decorrelating AUs. The first uses fully supervised learning with random sampling, second is using semi-supervised learning using pseudo-labelling and lastly semi-supervised using unsupervised pre-training.

3.1 Fully supervised learning with random up-sampling

Up-sampling is a method used to increase the number of instances in a dataset. Fully supervised learning with up-sampling is primarily used to address imbalanced datasets [9], where some classes significantly outnumbers others, by creating more instances of the minority class. In this approach, we up-sampling from the existing labelled data set to increase the training set size. Overall, up-sampling is a versatile tool that enhances the learning and generalization capabilities. In Figure 2 the full procedure is shown.

3.2 Semi supervised learning using pseudo-labelling

This method relies on making use of the vast amount of unlabelled AU data that exist out there. It make use of labelled data to train an AU classifier, then using the classifier, label the unlabelled data and use all the data to train the AU classifier again. Figure 3 shows the procedure.

3.3 Semi supervised learning using unsupervised pre-training

This method uses data to train an Expression Manipulator [10] which is able to generate additional data. Here we'll go through the main parts of Expression Manipulator as described in [10].

It consists of relative AUs, MSF module, network structure and loss function.

3.3.1 Relative Action Units

Instead of using discrete emotion labels or absolute facial AUs to control the facial expression as it was used to do in the past, the authors proposed a method that uses relative AUs. RAUs are defined as the difference between the AUs of the target expression and the AUs of the input image. This allows the generator to learn to only modify the regions of the face that are relevant to the desired expression.

3.3.2 Multi-scale feature fusion (MSF)

The generator in the proposed method is a U-Net architecture that is augmented with an MSF module. The MSF module helps to improve the quality of the generated images by fusing features from different scales. The idea for MSF feature is that fusing the features from low-to-high, and the two kinds of conv streams. MSF module takes the feature across the encoder and the MSF modules as well as RAUs as input to learn to alter image features at different spatial sizes. Secondly, MSF inject the condition at each MSF module which helps MSF to learn the consistency of expressions of features with different resolutions, between encoder and decoder features.

3.3.3 Loss function

The proposed method uses a combination of loss functions to train the generator [10] to make sure that the generated images look realistic, have the desired AUs and keeps the identity of the person. These loss functions include:

- The adversarial loss, which encourages the generated image to be realistic and indistinguishable from real images.
- Conditional Fulfillment makes sure that the output image possess the desired AUs.
- Reconstruction Regularization loss to make sure that the faces in both input and output images keep the identity of the person untouched.

To wrap up, the input to EM is a face image and a target expression. The target expression can be specified as a discrete AU label, or as a continuous value, such as the intensity of the AU. The network make use of the components and the output of EM is a new face image that has the desired expression. The quality of the output image depends on the quality of the input image, the accuracy of the RAUs, and the complexity of the expression.

4 Experiments and Results

4.1 Datasets

We perform all of our experiments on two datasets. The first is a "toy" data set that we have created using MNIST digits [3].

MNIST This dataset is a toy dataset that we have created using the MNSIT digits [3]. It contains images 96x96 in size, where we consider each digit as a unique action unit. The existence of an action unit is the digit existence or not. If it exists, then it's activated, otherwise it has a 0. An example is shown in Figure 1. The labels for the easy MNIST example would be [1,1,1,0,1,1,0,0,0].

When creating the images the probability an action unit to be activated is 20%. To make it more realistic, we included a correlation between digits 2 and 3, meaning that if digit 2 is activated there is 70% probability 3 will be activated too. The algorithm is also shown in 15. That is done to mimic the real behaviour of AUs, as for example, according to FACS [5], if AU 6 and 12 co-exist (Cheeck Raiser and Lip Corner Puller respectively) then the emotion in the face of the person is Happiness, thus, a lot of AUs co-exist in one face.

Finally, we also created images using gaussian noise to make the dataset harder. An example can be shown in Figure 1 with the 2nd being an MNIST image with gaussian noise ($\mu = 0.5, \sigma = 1$) and the 3rd being harder with gaussian noise ($\mu = 1, \sigma = 5$).

Faces datasets

The Faces datasets consists of 5 different datasets with faces. 4 Publicly available datasets which are: DISFA [13], DISFA+ [12], ADFES [19], WSEFESP [15] and also 1 private datasets owned by Vicar Vision [7]. In total the face combined dataset contains 72,901 images from different participants that are different age, ethnicity, gender, etc.

Finally we use AffectNet [14]-a publicly available unlabelled data set, to self-pseudo label.

Algorithm 1 Grid Initialization Algorithm

```
 1: Initialize a 3x3 grid.
 2: for digit from 1 to 9 do
 3:     Generate a random number between 0 and 1.
 4:     if random number $\leq 0.2$ then
 5:         Activate the action unit.
 6:     end if
 7:     if action unit is activated then
 8:         Assign the digit to the corresponding position in the grid.
 9:     else
10:         Assign 0 to the corresponding position in the grid.
11:     end if
12: end for
13: if digit 2 is present in the grid then
14:     Assign digit 3 to the grid with a probability of 70% and vice versa.
15: end if
```

4.2 Set up of action unit classifier

The hyper parameters of the classifier of the AUs, has been already tuned already so the settings like: batch size, optimizer,etc. are fixed. More specifically:

Parameter	Value
Batch size	100
Max epochs	1000
Min epochs	0
Patience	20
Loss	mse
Optimizer	radam
Learning rate	0.001

Table 1: Model Training Parameters

All experiments are evaluated with the metric F1-score and False Discovery Rate. F1-score measures:

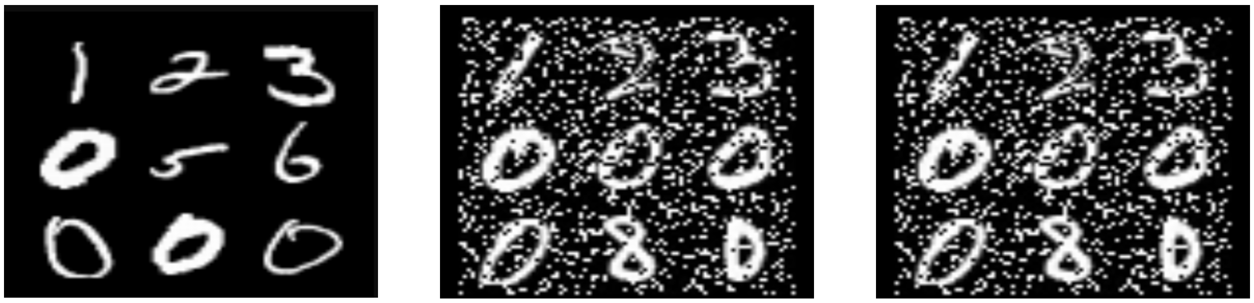


Figure 1: Correlated examples between digit 2 and 3 for normal/medium/hard MNIST toy dataset

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision+recall}} \quad (1)$$

Whereas False Discovery Rate measures:

$$(FDR) = \frac{\text{False Positives}}{\text{False Positives + True Positives}} \quad (2)$$

to measure how well we are able to de-correlate the AUs, the lower the FDR the better de-correlation we are able to achieve.

To account for the inherent stochasticity of neural networks, each experiment was repeated five times and the average F1 score, FDR was calculated across all runs. This was done to ensure that the reported results were representative of the model's performance, while also mitigating the impact of any random variation that could occur during individual runs.

4.3 Set up to train Expression Manipulator

To train the Expression Manipulator [10] we had to set up the hyper parameters. In the paper they proposed tuned hyper parameters that we adopted. More specifically some of the core hyper parameters:

We train the model by Adam [8] optimizer with settings of $\beta_1 = 0.5$, $\beta_2 = 0.999$ for 30 epochs at initial learning rate of 0.0001, and then linearly decay the rate to 0.0001 for fine-tuning.

To evaluate Expression Manipulator [10] and pick the best model among 50 epochs, we used 2 different metrics:

- Qualitative Evaluation: Comparing the same images and how they evolve during each epoch got us an approximation of which epoch the generator starts to create normal looking fake data.

- Quantitative Evaluation: Measuring the conditional loss convergence for the minimized loss of the discriminator-generator [10].

We applied EM in 2 different data sets, grid MNISTS toy data set and on Faces. However, we had different approach in terms of how to apply EM [10] in both set ups, we trained Expression Manipulator [10] with both labelled and self-pseudo labelled data, as we found out that the amount of labelled data weren't enough to train it properly.

- Experiments on MNIST: To generate MNIST data using Expression Manipulator [10], we created images and then activated only 1 number per time by setting it to 1 and all others 0. This was done to create **independent** AU samples to feed the model.

- Experiments on Faces: To generate fake images using Expression Manipulator, we found the neutral images (All AUs set to 0) in our datasets which were 3K out of the total images. Then, we created new images activating one AU per time by setting it to 0.7 because if more it was too intense and not realistic. So, 1 neutral image would generate 20 new images. Another approach we tried, was to deactivate an AU from the images we have. This approach we found early doesn't yield promising results as the images generated were unrealistic and not suitable to use. Furthermore, we generated images that have multiple AUs at the same time to create an emotion, as for example according to FACS [5] AU 6 and 12 combined form the happy emotion. However, due to limitations of time and data we couldn't use them.

4.4 Experiments on MNIST

Different set-ups were used on how to train the action unit classifier that we will go through in the following sections. Test set consist of 9000 samples from which each time 1 AU is activated independently from each other (1000 samples each).

4.4.1 Train AU classifier fully supervised learning

The results that we have obtained by training the Action Unit classifier using 3,000 and 30,000 MNIST of each mnist data set (medium and hard) and then tested the uncorrelated data set.

Dataset	F1-test	FDR test
MNIST 3K medium	64.70	7.76
MNIST 3K hard	46.42	23.59

Table 2: F1 Scores and FDR for fully supervised learning for 3K training MNIST

Dataset	F1-test	FDR test
MNIST 3K medium	84.82	2.15
MNIST 3K hard	65.01	11.46

Table 3: F1 Scores and FDR for fully supervised learning for 30K training MNIST

4.4.2 Train AU classifier Semi-supervised learning using pseudo-labeling

The results that we have obtained by training the Action Unit classifier using 3,000 and 30,000 MNIST of each mnist data set (medium and hard) trained on best model from 4.4.1 and then tested the uncorrelated data set.

Dataset	F1-test	FDR test
MNIST 3K medium	64.22	4.64
MNIST 3K hard	45.45	24.88

Table 4: F1 Scores and FDR for Semi-supervised using pseudo-labeling training on 3K MNIST and 9K self-pseudo labelled

Dataset	F1-test	FDR test
MNIST 3K medium	84.85	1.77
MNIST 3K hard	64.64	10.36

Table 5: F1 Scores and FDR for Semi-supervised using pseudo-labeling training on 30K MNIST and 90K self-pseudo labelled

4.4.3 Train AU classifier Semi-supervised learning using unsupervised pre training (Expression Manipulator)

The activation level we used for the Expression Manipulator [10] data is 1 as for MNIST data set we either have the Action Unit activated or not. Expression Manipulator was trained using 3000 labelled and 9000 pseudo labelled data in case of MNIST 3K data set whereas in MNIST 30K data we used 30000 and 90000 respectively.

Dataset	F1-test	FDR test
MNIST 3K medium	65.07	6.10
MNIST 3K hard	44.52	24.84

Table 6: F1 Scores and FDR Semi-supervised using unsupervised pre- training for 3K MNIST and 9K Expression Manipulator data

4.4.4 Train AU classifier on labelled and random sampling from labelled data

The results that we have obtained by training the Action Unit classifier using 3,000 and 30,000 MNIST

Dataset	F1-test	FDR test
MNIST 30K medium	83.59	3.09
MNIST 30K hard	58.84	12.75

Table 7: F1 Scores and FDR Semi-supervised using unsupervised pre- training for 30K MNIST and 90K Expression Manipulator data

of each mnist data set (medium and hard) trained and labelled data random sampled 9,000 and 90,000 respectively.

Dataset	F1-test	FDR test
MNIST 3K medium	60.68	10.11
MNIST 3K hard	41.18	28.15

Table 8: F1 Scores and FDR fully supervised learning 3K MNIST and 9K Random Sampled data

Dataset	F1-test	FDR test
MNIST 30K medium	80.2	2.26
MNIST 30K hard	61.8	12.41

Table 9: F1 Scores and FDR fully supervised learning 30K MNIST and 90K Random Sampled data

4.4.5 Comparison of the methods

The combined results using the different approaches are shown for 3K data sets in 10, 11 and for 30K data sets in 12 and 13. In 3K data sets case, in medium difficulty the best performance is achieved by pseudo-labelling for FDR whereas the F1 score is achieved by EM. On the other hand in 3K hard difficulty data set, fully supervised works best in both cases, as the data contain too much noise and they are too few. In the case of 30K medium and hard difficulty data set we expected EM to be performing best according to our hypothesis, however, pseudo-labelling yields the best results in both FDR and F1-score.

4.5 Experiments on Faces

The experiments on the faces had the same parameters as mentioned 4.2. The train of the model was done using 58,320 images from the datasets mentioned in 4.1 and then tested with 4000 unseen images.

Baselines	**FDR**	**F1 score**
4.5.1	7.76%	64.70%
4.5.2	**4.64%**	64.22%
4.5.3	6.10%	**65.07%**
4.5.4	10.11%	60.68%

Table 10: The table shows the FDR and F1 score for each of the augmenting baselines tried for the 3K size medium difficulty data set. The best de-correlation (FDR) between AU 2 and AU 3 happens with semi-supervised learning using pseudo-labeling whereas the best F1-score is achieve with semi-supervised learning using pre-training

Baselines	**FDR**	**F1 score**
4.5.1	**23.59%**	**46.42%**
4.5.2	24.88%	45.45%
4.5.3	24.84%	44.52%
4.5.4	28.15%	41.18%

Table 11: The table shows the FDR and F1 score for each of the augmenting baselines tried for the 3K size hard difficulty data set. The best results in both de-correlation (FDR) of AU 2 and AU 3 and F1 score is achieved by fully supervised learning

4.5.1 Train AU classifier fully supervised learning

The results that we have obtained are by training the Action Unit classifier using 58,320 faces images with the 5 datasets mentioned at 4.1 combined and then tested in 4000 unseen images.

4.5.2 Train AU classifier Semi-supervised learning using pseudo-labeling

The results we obtained training the classifier on 58,320 labelled images plus pseudo labelled AffectNet [14] with 60K and 300K respectively.

Baselines	**FDR**	**F1 score**
4.5.1	2.15%	84.82%
4.5.2	**1.77%**	**84.85%**
4.5.3	3.09%	83.59%
4.5.4	2.26%	80.2%

Table 12: The table shows the FDR and F1 score for each of the augmenting baselines tried for the 30K size medium difficulty data set. The best results in both de-correlation (FDR) of AU 2 and AU 3 and F1 score is achieved by semi-supervised learning using pseudo-labeling

Baselines	**FDR**	**F1 score**
4.5.1	11.46%	**65.01%**
4.5.2	**10.36%**	64.64%
4.5.3	12.75%	58.84%
4.5.4	12.41%	61.8%

Table 13: The table shows the FDR and F1 score for each of the augmenting baselines tried for the 30K size hard difficulty data set. The best results in de-correlation (FDR) of AU 2 and AU 3 is achieved by semi-supervised learning using pseudo-labeling and best F1 score is slightly better with fully supervised learning

4.5.3 Train AU classifier Semi-supervised learning using unsupervised pre training (Expression Manipulator)

The activation level we used for the Expression Manipulator [10] data is 0.7 as we discussed in 4.3. Training only with EM data and labelled data we obtained the following results.

4.5.4 Comparison of the methods

The combined results on faces using different approaches are shown in 17. The findings are that Expression Manipulator outperforms pseudo-labelling which is unexpected according to the toy data set experiments, however, our guess is that AffectNet [14] data set offers diversity and EM is able to generate higher quality images and thus, improve model's performance.

Dataset	F1-test	FDR test
Faces	61.31	47.75

Table 14: F1 Scores and FDR fully supervised learning for faces

Dataset	F1-test	FDR test
Labelled 60K + AffNet 60K	63.04	47.92
Labelled 60K + AffNet 300K	64.04	47.86

Table 15: F1 Scores and FDR Semi-supervised using pseudo-labeling

5 Discussion

In this paper we examine which of the methods presented, work best for de-correlating AUs and simultaneously increase the overall performance measured with F1 score. However, there are some limitations.

Semi-supervised using unsupervised pre training method that uses Expression Manipulator [10] to augment AUs and pseudo-labelling need a solid amount of data to be able to generate high quality images as it shown in our experiments as with 3K hard data set it fails. In addition, the toy data set is simplistic as it has only one pair of highly correlated AUs and the grid is 3x3 whereas in the real world problem the face is much more complicated. Finally, the experiments on faces were limited due to hardware and data availability constraints.

6 Conclusion

To conclude, we explored the value of different augmenting techniques to improve the performance of an AU estimator and be able to de-correlate the AUs. The main goal of this study was to discover if using self-pseudo-labelled data to train an Expression Manipulator [10] to generate a training set for training a classifier is a better alternative to directly using an equal amount of self-pseudo-labelled data for training the classifier [11] as we can create infinite amount of independent AUs to de-correlate the AUs.

The experiments we have done in the toy MNIST data set led us to some conclusions. Firstly,

Dataset	F1-test	FDR test
Labelled 60K + EM 60K	65.34	46.83

Table 16: F1 Scores and FDR Semi-supervised using unsupervised pre training

Baselines	**FDR**	**F1 score**
4.5.1	47.75%	61.31%
4.5.2.1	47.92%	63.04%
4.5.2.2	47.86%	64.04%
4.5.3	**46.83%**	**65.34%**

Table 17: The table shows the FDR and F1 score for each of the augmenting baselines tried for the faces Dataset. The best results in de-correlation (FDR) of AU 1 and AU 2 is achieved by Semi-supervised learning using unsupervised pre-training

semi-supervised learning using pseudo-labelling is the **most promising when it comes to de-correlating AUs and performance-wise**. We think that happens as the toy data set is artificial and simplistic, the fully-supervised method can train a decent classifier to label the unlabeled data. This is mostly the case when it comes to the data set that was trained with 30K data set as when we have little data available Expression Manipulator works almost as good as pseudo-labelling and outperforms random sampling. On the other hand, when there are enough data to sample from Expression Manipulator is worse.

When it comes to faces data set, in table 17 is shown that Expression Manipulator outperforms pseudo-labelling by 1% in FDR and 1% in F1 score. Expression Manipulator make use of a vast amount of unlabelled data AffectNet[14] so this provide more diversity in our dataset and thus, Expression Manipulator is able to generate higher quality images than provided in just using AffectNet for pseudo-labelling.

References

[1] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018. 3

[2] Wen-Sheng Chu, Fernando De la Torre, and Jeffrey F Cohn. Learning facial action units with spatiotemporal cues and multi-label sampling. *Image and vision computing*, 81:1–14, 2019. 3

[3] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 5

[4] Paul Ekman. Facial action coding system, n.d. 3

[5] Paul Ekman and Wallace V Friesen. Facial action coding system. *Environmental Psychology & Nonverbal Behavior*, 1978. 2, 5, 7

[6] Paul Ekman Group. Facs manual, n.d. 3

[7] Amogh Gudi. Recognizing semantic features in faces using deep learning. *arXiv preprint arXiv:1512.00743*, 2015. 5

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[9] Pradeep Kumar, Roheet Bhatnagar, Kuntal Gaur, and Anurag Bhatnagar. Classification of imbalanced data: review of methods and applications. In *IOP conference series: materials science and engineering*, volume 1099, page 012077. IOP Publishing, 2021. 4

[10] Jun Ling, Han Xue, Li Song, Shuhui Yang, Rong Xie, and Xiao Gu. Toward fine-grained facial expression manipulation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 37–53. Springer, 2020. 2, 3, 4, 6, 7, 9

[11] Marco Loog. Contrastive pessimistic likelihood estimation for semi-supervised classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(3):462–475, 2015. 2, 3, 9

[12] Mohammad Mavadati, Peyten Sanger, and Mohammad H Mahoor. Extended disfa dataset: Investigating posed and spontaneous facial expressions. In *proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–8, 2016. 5

[13] S Mohammad Mavadati, Mohammad H Mahoor, Kevin Bartlett, Philip Trinh, and Jeffrey F Cohn. Disfa: A spontaneous facial action intensity database. *IEEE Transactions on Affective Computing*, 4(2):151–160, 2013. 5

[14] Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the

wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2017. 2, 3, 5, 8, 9, 10

[15] Michal Olszanowski, Grzegorz Pochwatko, Krzysztof Kuklinski, Michal Scibor-Rylski, Peter Lewinski, and Rafal K Ohme. Warsaw set of emotional facial expression pictures: a validation study of facial display photographs. *Frontiers in psychology*, 5:1516, 2015. 5

[16] A. Pumarola, A. Agudo, A.M. Martinez, A. Sanfeliu, and F. Moreno-Noguer. Ganimation: One-shot anatomically consistent facial animation. 2019. 3

[17] Shengju Qian, Kwan-Yee Lin, Wayne Wu, Yangxiaokang Liu, Quan Wang, Fumin Shen, Chen Qian, and Ran He. Make a face: Towards arbitrary high fidelity face manipulation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10033–10042, 2019. 3

[18] Y-I Tian, Takeo Kanade, and Jeffrey F Cohn. Recognizing action units for facial expression analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 23(2):97–115, 2001. 2

[19] Job Van Der Schalk, Skyler T Hawk, Agneta H Fischer, and Bertjan Doosje. Moving faces, looking places: validation of the amsterdam dynamic facial expression set (adfes). *Emotion*, 11(4):907, 2011. 5

[20] Yong Zhang, Weiming Dong, Bao-Gang Hu, and Qiang Ji. Classifier learning with prior probabilities for facial action unit recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5108–5116, 2018. 3

[21] Zheng Zhang, Shuangfei Zhai, Lijun Yin, et al. Identity-based adversarial training of deep cnns for facial action unit recognition. In *BMVC*, page 226. Newcastle, 2018. 3

[22] Kaili Zhao, Wen-Sheng Chu, and Aleix M Martinez. Learning facial action units from web images with scalable weakly supervised clustering. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 2090–2099, 2018. 3

A Supplementary Material

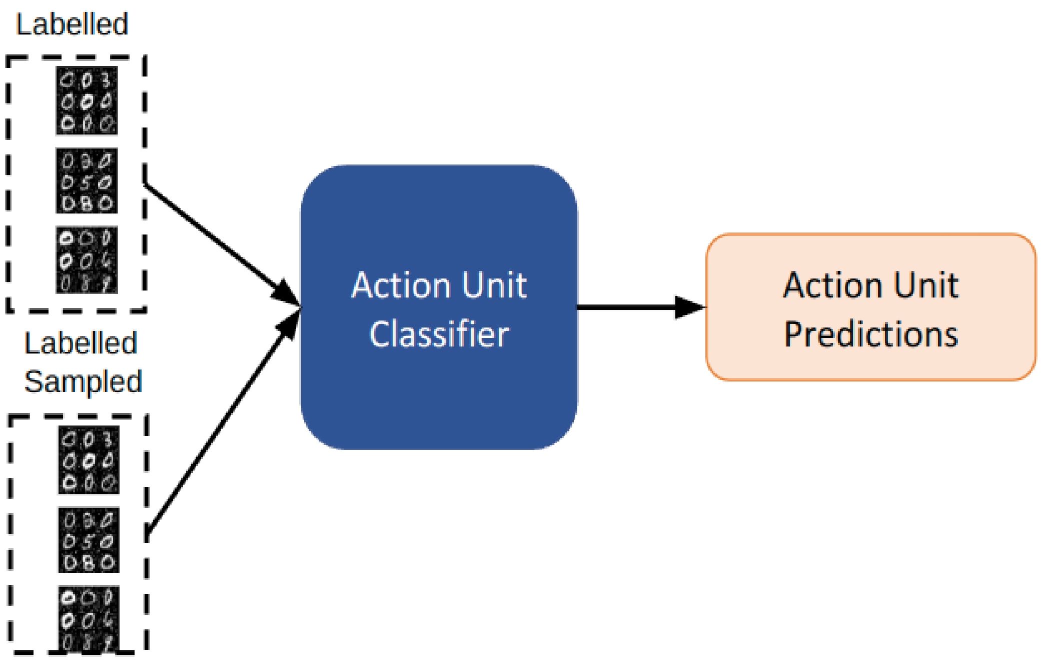


Figure 2: Random up-samping explanation

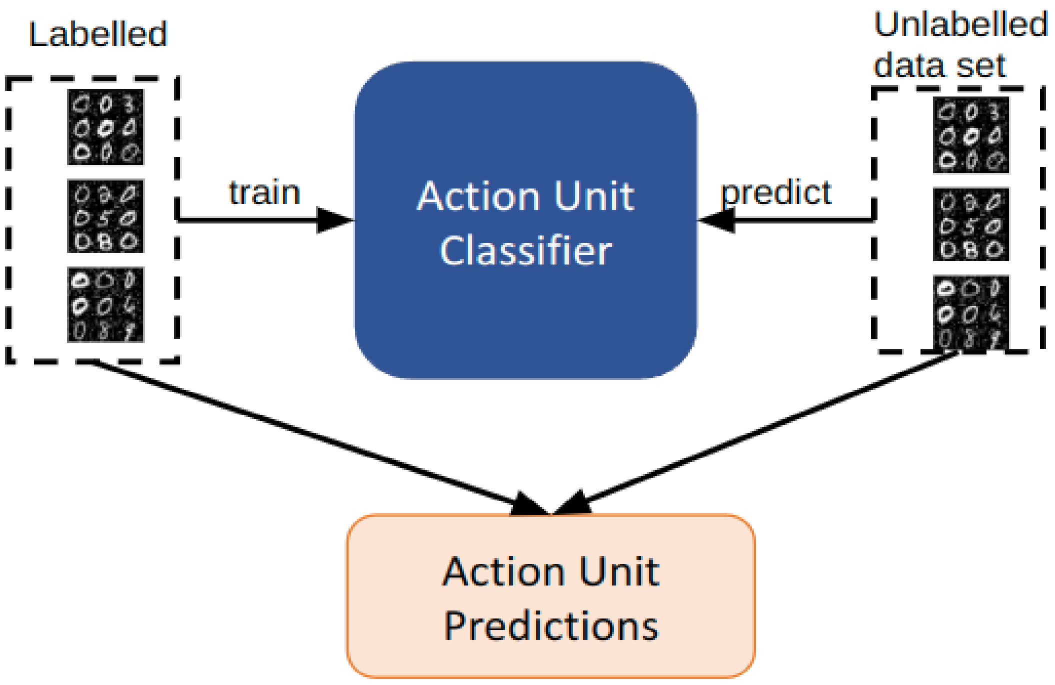


Figure 3: Self-supervised learning using pseudo-labelling

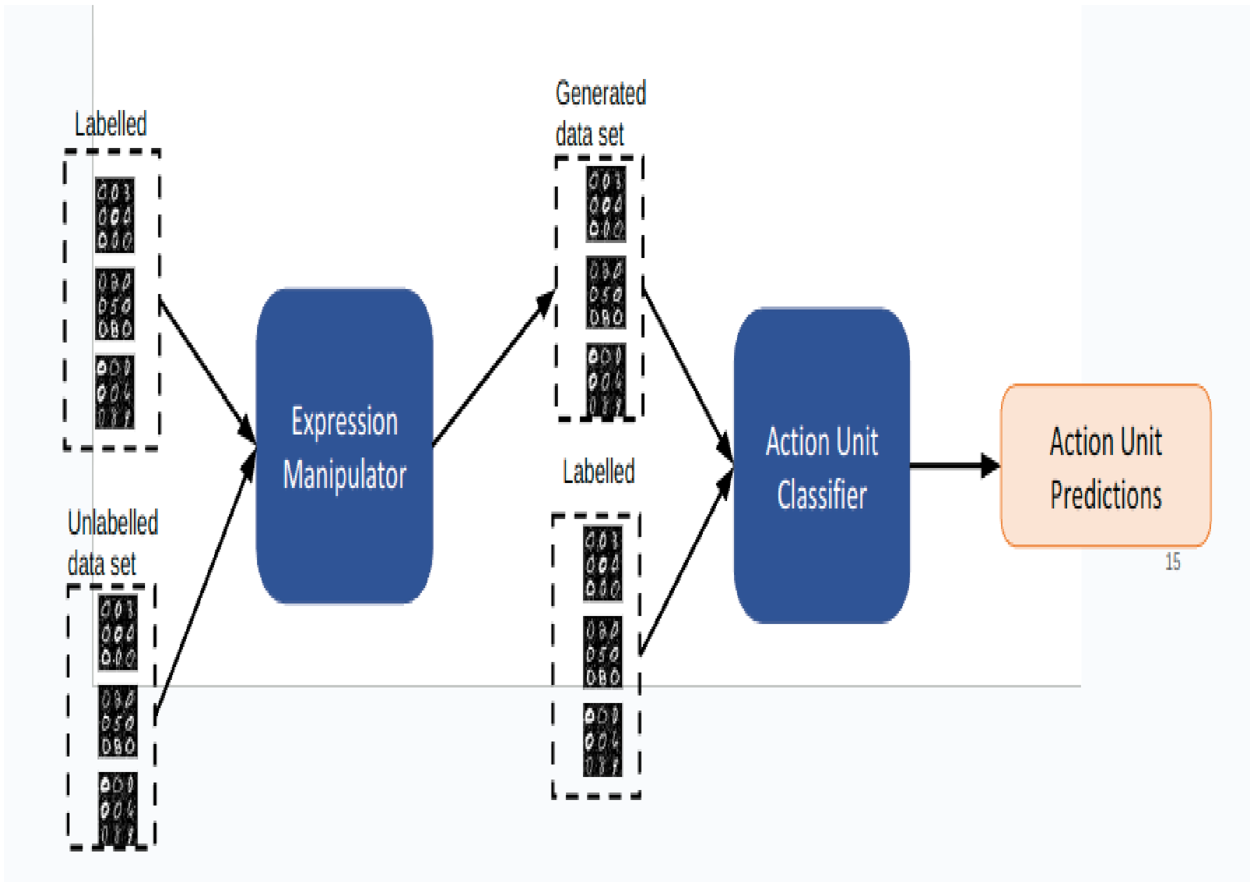


Figure 4: Self-supervised learning using pre-training

MSc thesis in Computer Science

Face image synthesis for robust facial analysis

Marios Marinos

September 2023

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

The work in this thesis was carried out in the:
Computer Vision Labat Delft University of Technology

Supervisors: Jan van Gemert,
Amogh Gudi

Abstract

Emotion recognition is a challenging problem in the field of computer vision. The automatic classification of emotions using facial expressions is a promising approach to understand human behavior in various applications such as marketing, health, and education. However, recognizing some emotions, such as anger, jealousy, contempt, and disgust, is more challenging than others due to their subtlety and rarity in the training data. In this paper, we try to investigate if using (self)pseudo-labelled data to train an Expression Manipulator [**?**] generator to generate a training set for training a classifier is a better alternative to directly using an equal amount of (self)pseudo-labelled data for training the classifier [**?**]. Specifically, we focus on augmenting the Action Units (AUs) of facial expressions, which are the basic units of facial movement that correspond to specific emotions.

Acknowledgements

I would like to express my sincere gratitude to my academic advisors, Jan Van Gemert and Amogh Gudi, for their patience, constant support, guidance and feedback. Their expertise and the input they provided to me have been invaluable in refining my research methodology and ensuring the integirty of my work.

I extend my thanks to Tim den Uyl and Amogh Gudi once again at VicarVision for giving me the chance to be an intern at their company. The assistance throughout this thesis had been valuable.

Lastly, I am deeply grateful to my friends and family, whose unconditional support, understanding, and constant motivation played a crucial role in helping me overcome challenges and persevere until the completion of this thesis. Their belief in me never wavered, even during moments when I felt overwhelmed and tempted to give up.

In conclusion, this thesis would not have been possible without the constant support and contributions of my academic advisors, VicarVision, family and friends.

Contents

1 Background about Face Analysis

1.1 Face analysis

In this section we are going to introduce the basic terms about Face analysis. Face analysis is a prominent field in computer vision and psycology which involes the study of facial expressions and their underlying components known as action units. AUs were introduced by Paul Ekman and Wallace V. Friescen in the 1970s [5].

1.1.1 Action Units

Actions units are distinct facial muscle movements that correspond to specific emotions or facial expressions. These units serve as building blocks for understanding the complex language of the human face.

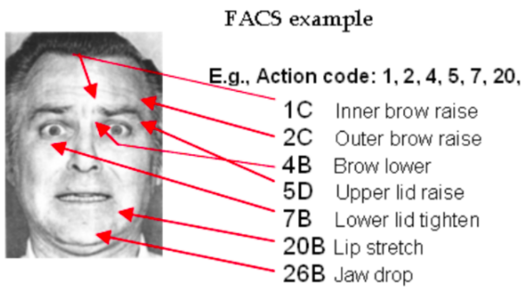


Figure 1.1: Action Units example

In figure 1.1 some action units are displayed. Combinations of them make the fully expressions. The table below shows some basic expressions and which AUs compose these expressions.

Table 1.1: Basic Expressions and Involved Action Units

Expression	**Involved Action Units**	**Description**
Happiness	AU 6 + 12	Cheek Raiser, Lip Corner Puller
Surprise	AU 1 + 2 + 5 + 26	Inner Brow Raiser, Outer Brow Raiser, Upper Lid Raiser, Jaw Drop
Fear	AU 1 + 2 + 4 + 5 + 7 + 20 + 26	Inner Brow Raiser, Outer Brow Raiser, Brow Lowerer, Upper Lid Raiser, Lid Tightener, Lip Stretcher, Jaw Drop

1.1.2 Action unit classifiers

There have been many approaches to identify automatically AUs in faces. The first and oldest was manual annotation using human experts to annotate the AUs. This method is subjective, time consuming and it requires to have FACS certificate to be able to do that.

Rule based systems was also an approach that was developed to identify Aus automatically by making use of predefined rules and heuristics. It makes use of facial landmarks and geometric features to identify specific facial movements.

Feature-based approaches are methods to extract relevant features from facial images and then use them to train a machine learning algorithm (e.g. SVM, Decision trees). This was extented by Deep learning and the use of Convolutional Neural Networks (CNNs) to learn directly features from facial images, which led to better results.

2 Background about Machine Learning

In this section we are going to introduce the basic terms about machine and deep learning.

2.1 Deep Learning

Neural Networks and therefore deep learning, mainly focus on simulating human's brain system [11]. Deep Learning has been the root for the success of Computer Vision and Natural Language Processing. Recently, autonomous cars and chat bots have been very trendy and useful.

In Figure 2.1, a single neuron is displayed. For this single neuron, a multi-dimensional input $X = x_1, x_2, ..., x_{d-1}, x_d$ will go through a linear function $\sum_i w_i x_i + b$ with w being weights and b bias. Then, the output α will be passed to a non-linear activation function g to add non-linearity to the model.

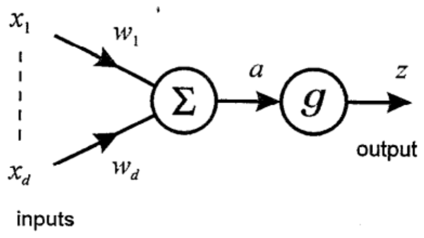


Figure 2.1: Model of a single neuron from [2]

In Figure 2.2 a 3-layer fully connected network is shown. The input vector is transformed through a series of hidden layers. The output layer will output the scores for the classification.

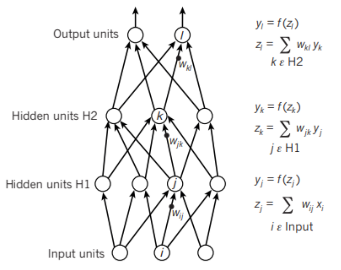


Figure 2.2: a 3-layer deep learning model from [11]

2.1.1 Activation functions

In a Neural Network, should we do not include the non-linear activation function, the final model's output will be simply a linear classifier, that has nothing to add in the existing linear classifiers (e.g. linear regression, Support Vector Machines, etc.) as it will be the weighted sum of the input. Therefore, we intimate the non-linearity with the activation functions, usually between each hidden layer, so it's applied in the output of a hidden layer.

Below we'll introduce some of the most known activation functions used:

Sigmoid

Sigmoid function is a non-linear activation function. It maps a real number in the (0,1) interval. It has one disadvantage when it comes to edge points and the output is very close to 0 or 1, the derivative will become very small and that leads to vanishing gradient problem.

$$f(x) = \frac{1}{1+e^{-bx}} \tag{2.1}$$

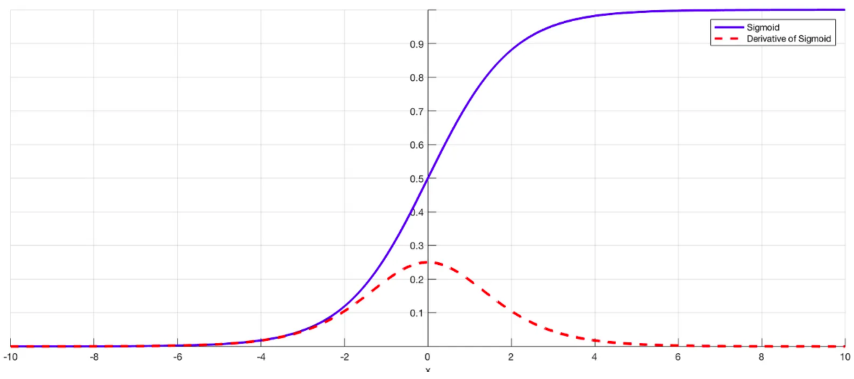


Figure 2.3: Sigmoid function and the derivative

ReLU and alternatives

ReLU (Rectified Linear Unit) is nowadays the most common activation function used, especially in deep neural networks. It is very fast and efficient computationally. ReLU is very simple as it returns 0 for anything that is negative, else it will return the positive value itself. The partial derivatives remain continuous and significant and therefore it is easy to circulate in the network [13].

$$f(x) = max(0,x) \tag{2.2}$$

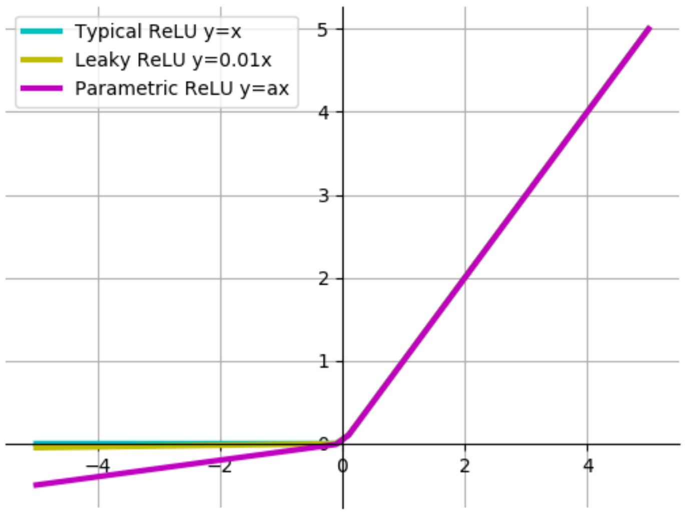


Figure 2.4: ReLU and variations

The typical ReLU, causes a problem called ”dead neurons” in the negative values of x, which leads to 0 derivatives, hence these neurons are not able to change their weights and therefore, learn. To solve this, Leaky ReLU, Parametric ReLU and different variations of ReLU have been tried throughout the years to solve the problem of having the dead neurons by assigning a (small) value to the negative inputs.

2.1.2 Loss Function

In order to apply the back propagation algorithm, we need a loss function to penalize the difference between predicted and the ground truth output during the training process. The end goal is to minimize the error as much as possible to achieve the best outcome.

For regression problems the most known loss function that is used is the Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(d_j - o_j)^2, \text{ where d is the label and o is the output (prediction)} \tag{2.3}$$

For classification problems, Cross Entropy loss function is the most common:

$$Loss = -(label * log(output) + (1 - label) * log(1 - output)) \tag{2.4}$$

2.1.3 Optimizers

As mentioned in 2.1.2, the main goal is to minimize the loss function $J(\theta)$ where θ are the weights of the neural network. The algorithm to solve the optimization process is the optimizer.

The most common algorithm used is the Gradient Descent. However, There have been different variations of Gradient Descent as it has some problems that we will discuss.

Gradient Descent

The partial gradient $\Delta_\theta J(\theta)$ is the derivative of a multi variable function.

$$\Delta_\theta J(\theta) = \frac{dJ(\theta)}{d\theta} \tag{2.5}$$

And the formula to update the weights is:

$$\theta = \theta - \eta * \Delta_\theta J(\theta; x_i, y_i) \tag{2.6}$$

The parameter η indicates how much the weights will change and it is called learning rate. It is very crucial when it comes in training any kind of Neural Network as if you have a high learning rate you might miss the global minimum whereas with a low learning rate the network is computationally expensive and make the network slow to converge.

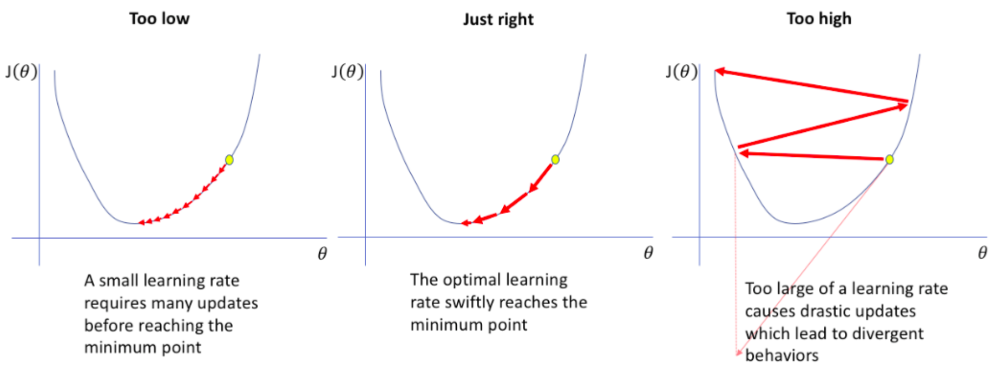

Too low
J(θ)
θ
A small learning rate requires many updates before reaching the minimum point
Just right
J(θ)
θ
The optimal learning rate swiftly reaches the minimum point
Too high
J(θ)
θ
Too large of a learning rate causes drastic updates which lead to divergent behaviors

Figure 2.5: Effect of low, good and high learning rates from [8]

The mains drawback of the Gradient Descent is the speed that the network needs to converge and the memory consumption, as the update take place after the whole dataset and the gradient of all needs to be computed. In addition, as mentioned before it is hard to tune the learning rate to avoid the problem of missing the global minimum or making it very slow.

Stochastic Gradient Descent and Mini-batch Gradient Descent

Stochastic Gradient Descent (SGD) solves both problems of speed and memory consumption as it updates the paremeters θ for each training example instead [3]. However, due to the update of each example it introduces high variance and can confuse the network.

There comes Mini-batch Gradient Descent, which is a cross over between GD and SGD [9]. It tries to find a balance between GD and SGD by updating the weights after a "batch" of n examples to avoid the high variance of some specific examples. **The batch size is a hyper-parameter as learning rate that needs to be tuned to achieve the best results.**

Adam

Lastly, Adaptive Moment Estimator (Adam) is the most common optimizer used nowadays as the algorithm itself tries to find the best learning rates and in which direction it needs to move. That makes it very easy to use and it's not heavily affect by the parameter tuning of learning rate, as previous methods. Adam has the best results so far to train efficiently deep neural networks [10].

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2.7}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2.8}$$

g_t stands for the gradients and β's are hyper parameters. Also, bias correction is applied to both formulas:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2.9}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_b^t} \tag{2.10}$$

And lastly, the update formula:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} * \hat{m}_t \tag{2.11}$$

2.1.4 Dropout

Dropout is a regularization technique to make the model avoid over fitting in the train data [15]. Simply, it ignores a number of random neurons in each different hidden layer with a probability p while the input and the output stays unchanged. This can lead to a more robust model as it relies less on local features. Therefore, dropout helps the model to perform better to unseen data and measure the overfitting of the model which is a common problem in neural networks.

2.1.5 Fully Connected Network

In a fully connected layer, each neuron is linked to every neuron in the preceding layer, as depicted in Figure 2.6. The fundamental operation within a fully connected layer is matrix multiplication. This process linearly transforms one feature space into another. When integrated into a Convolutional Neural Network (CNN) model, the fully connected layer serves to amalgamate the features extracted by the preceding convolutional layers and map them into the representation space associated with labels.

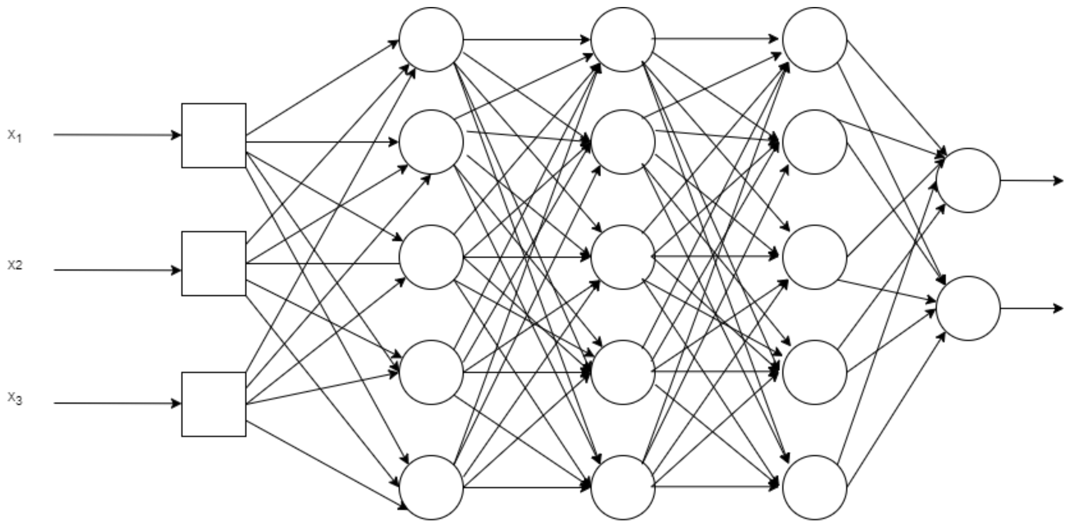


Figure 2.6: Fully connected network

2.2 Convolution Neural Networks

Convolution Neural Networks (CNN) [12] are deep learning models, that are commonly used for problems that have images as an input (e.g. image classification, object detection, etc.). The need behind CNN's is the efficiency of the model in terms of speed and memory consumption. Should we input an image without pre-processing in a neural network the network will be come very slow and the memory will overflow. Hence, CNN's try to minimize the image as much as possible by keeping the most important features while lowering the dimensions.

LeNet was the first architecture of CNN's and then more advanced and complicated Convolutional Neural Networks were made like AlexNet and VGG. [1]

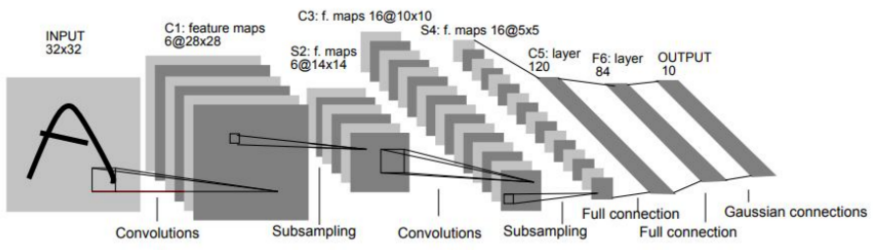
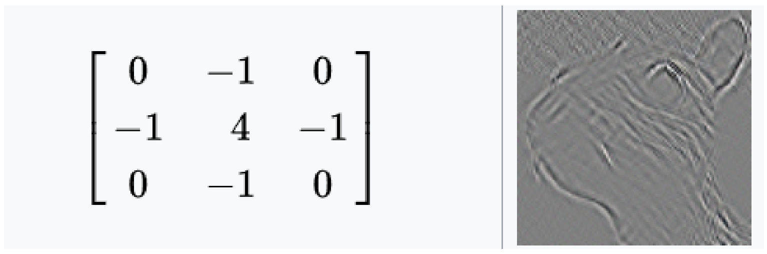


Figure 2.7: LeNet first architecture of CNN's

2.2.1 Convolution Layer

The most important part of a Convolution Neural Network is the convolution layer. It's a feature extraction mechanism for images by applying filters or kernels that represent a feature, e.g. an edge or a curve. To illustrate that, inn figure 2.8 a kernel of edges is shown that is applied to an image and the output of it.



$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Figure 2.8: Edge kernel from Wikipedia

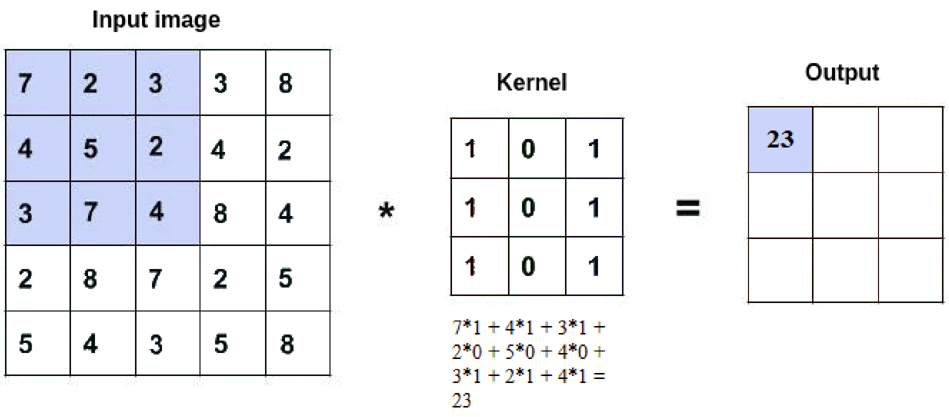


Figure 2.9: Kernel applied example

The kernel then is applied as it shown in figure 2.9. The input image in the example is 5x5 and the kernel is 3x3 and the output will be smaller as we want to compress the image (3x3 here). In addition, we have a parameter called stride and that means how many columns it will ”jump”. An example of stride = 2 is shown in figure 2.10. Finally, we can also apply zero padding in the initial input image and fill with zeros around the input image, to weight more the pixels that are in the first row/column. It’s worth noticing that the parameters of a convolution kernel are learnable during the training of the model and multiple kernels usually will be applied.

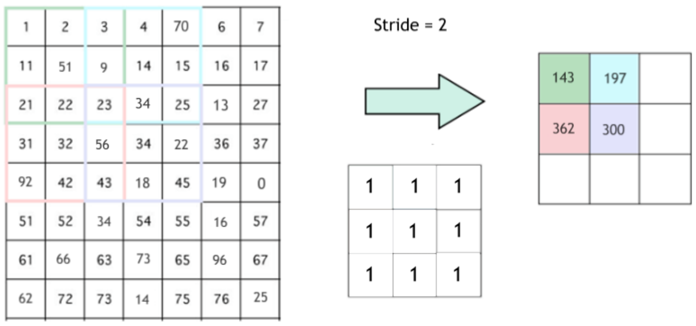


Figure 2.10: Stride example

2.2.2 Pooling Layer

Pooling layer is a mechanism to down sample the feature map by using max or average [7]. The idea behind is that we want to compress more the image and by doing that we keep the most important pixel from a specific smaller area. An example that does max pooling and average pooling is shown in figure 2.11. There is also stride as it is mentioned in chapter 2.2.1 for the same reason.

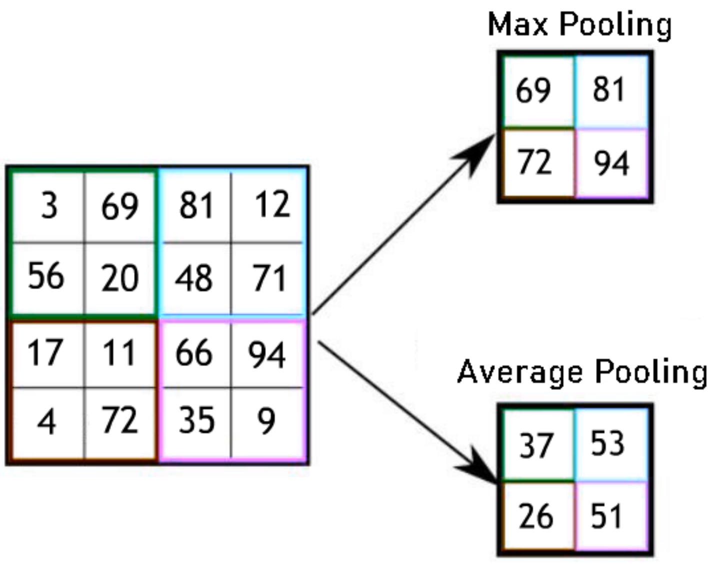


Figure 2.11: Stride example

To sum up, Convolutional layer and pooling layer makes up a convolutional block. Then we can apply ReLU or whatever activation function we need. With the convolutions blocks we form a Convolutional Neural Network that extract features in different levels of the image and that leads in the decrease of the size of the input image as desired. Finally, we flatten the input image and we feed it to a Fully Connected Layer that will do the classification of the input image as discussed in chapter 2.1.5.

2.2.3 U-Net

U-Net is a convolutional neural network (CNN) architecture that is mostly used for image segmentation task. It was introduced and developed by Olaf Ronneberger et al. in 2015 [14] and got its name by the ”U” shape it has.

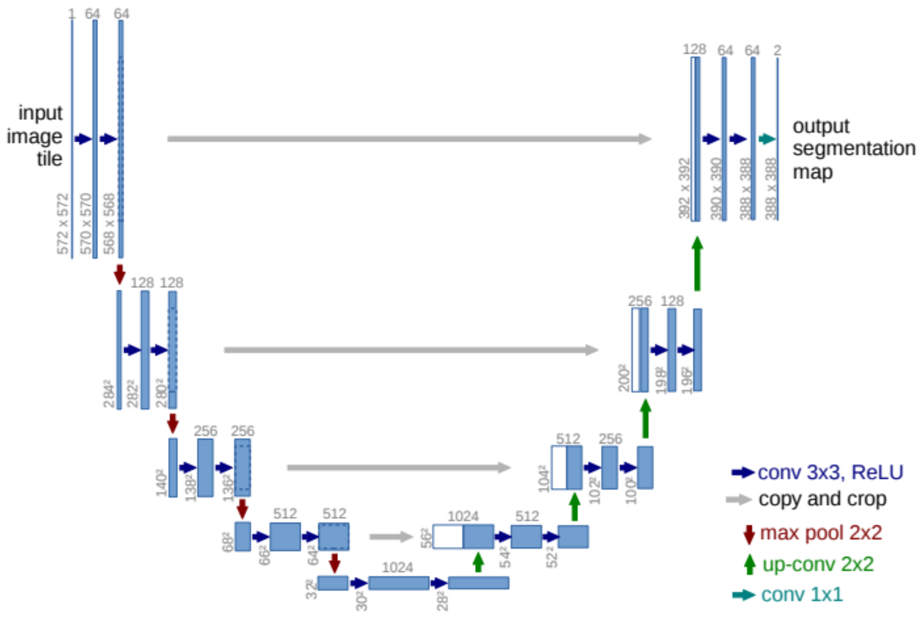


Figure 2.12: U-net architecture (example for 32x32 pixels in the lowest resolution). The blue boxes correspond to multi-channel feature map. [14]

The main parts that compose U-net are:

- Contracting Path: Consists of convolutional layers and max-pooling layers for feature extraction and spatial dimension reduction.
- Bottleneck: Center of the network, maintaining high-resolution spatial information and abstract feature representations.
- Expanding Path: Contains upsampling and convolutional layers to increase spatial resolution for segmentation.
- Skip Connections: Extensive use of skip connections to capture fine-grained details and improve accuracy.

- Final Output: Produces a segmentation mask, typically the same size as the input image, representing segmented regions.

2.3 Generative Adversarial Networks

A Generative Adversarial Network (GANs) is a framework that uses two neural networks (generator and discriminator) against each other in a zero-sum game. The generator creates new data that tries to make them as good as real data used, whereas the discriminator tries to find which data are real and fake [6].

GANs have been very popular recently to generate realistic images, text and music.

In Figure 2.13 the full architecture of a GAN is shown and how tihs works to generate the fake realistic data.

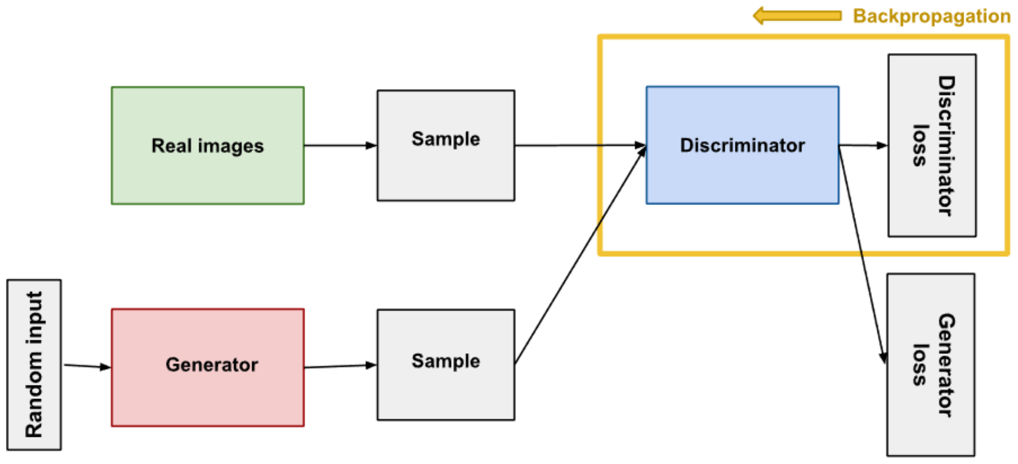


Figure 2.13: The GAN architecture with the back propagation training in discriminator is shown in the figure .[4]

The generator is trained to minimize the discriminator's probability of correctly classifying its output as real. This means that the generator is trying to create data that is so realistic that the discriminator cannot tell the difference between it and real data.

On the other hand, the discriminator is trained to maximize its probability of correctly classifying real and fake data. This means that the discriminator is trying to learn to distinguish between real and fake data as accurately as possible.

As they compete against each other, they both become better. The final goal is that discriminator is not able to detect between which data are fake and which real, where we can say safely that generator is able to create realistic and creative data. For the theoretical background and the equations Goodfellow on his work explain it very good.[6]

Bibliography

[1] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[2] C. M. Bishop. Neural networks and their applications. *Review of scientific instruments*, 65(6):1803–1832, 1994.

[3] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer, 2010.

[4] G. Developers. The discriminator, 2023.

[5] P. Ekman and W. V. Friesen. Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124, 1971.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[7] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[8] J. Jordan. A gentle introduction to learning rates in neural networks, 2018. `https://www.jeremyjordan.me/nn-learning-rate/`.

[9] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson. Mini-batch gradient descent: Faster convergence under data sparsity. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2880–2887. IEEE, 2017.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[12] K. O'Shea and R. Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[13] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[14] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[15] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.