DSIM Project

Mono/Bi Dimensional Signal Processing and GAN

Vittorio Haardt (853268)
Luca Porcelli (853189)

(Group LV)
Università degli studi di Milano-Bicocca

20 Feb 2024

Outline

1. Mono-Dimensional Signal Classification
2. Bi-Dimensional Signal Classification
3. StarGAN Model

0.1 Data

Emotion classification and generation from different type of signals.

Modo Dimensional

~12,000 audio file
7 categories

Bi Dimensional

~14,000 images
5 categories

GAN

Same dataset used in bi-dimensional signal classification

Outline: Mono-Dimensional Signal Classification

1.1 Data and Preprocessing

Data from 4 different datasets [1] can make the results more generalizable.
Many emotion classification datasets are limited to speech made by the same actors pronouncing the same phrases. Unifying different datasets can make the model more generalizable.

Class

- Angry - (16.7%)
- Happy - (16.46%)
- Sad - (16.35%)
- Neutral - (14.26%)
- Fearful - (16.46%)
- Disgusted - (15.03%)
- Surprised - (4.74%)

Unbalanced class distribution was addressed through undersampling to match the less numerous class.

~ 4,200 records

[1] Audio Emotions, Uldis Valinis (https://www.kaggle.com/datasets/uldisvalainis/audio-emotions)

1.1 Data and Preprocessing

Since the audio came from different dataset, the **sound rate** was different.
The length (sound data/sound rate) was also different different.

Need to have a standard audio format, used also for deployment.

Preprocessing

- Uniform sound rate (48 kHz)
- Padding for the same length (~ 2 sec.) [Avg of sound data length]

1.2 Classification

Task: Compare the performance of two feature set on audio classification.

Features

- Set of features [2][3]
- MFCC

Feature selection strategy: implemented to remove noise and aid classification.

Evaluation

- Machine Learning Architectures
- Deep Learning Architectures

Evaluated using Accuracy

[2] Breebaart, Jeroen, and Martin F. McKinney. "Features for audio classification." *Algorithms in Ambient Intelligence.»*
[3] Liu, Zhu, Yao Wang, and Tsuhan Chen. "Audio feature extraction and analysis for scene segmentation and classification."

1.2 Classification: features' set

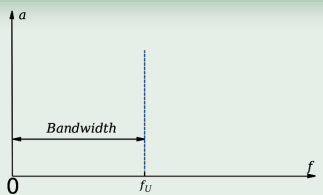
Spectral Centroid

$$Centroid = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)}$$

RMS

$$RMSE(x) = \sqrt{\frac{1}{N}\sum_{n} |x(n)|^2}$$

Bandwidth



Energy

$$E(x) = \sum_{n} |x(n)|^2$$

1.2 Classification: features' set

Spectrum magnitude

The Magnitude Spectrum of a signal describes a signal using frequency and amplitude

Zero Crossing Rate

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1} |\,\mathrm{sign}[x_i(n)] - \mathrm{sign}[x_i(n-1)]|$$

Spectral roll off

Center frequency for a spectrogram bin such that at least roll_percent (0.85) of the energy of the spectrum in this frame is contained in this bin and the bins below.

1.2 Classification: MFCC

Mel-Frequency Cepstral Coefficients (MFCC) are a feature widely used in speech and audio processing. They represent the short-term power spectrum of a sound and are derived from the human auditory system.

Steps

1. Take the Fourier trasfomation of a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosene trasformation of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

13 MFCC coefficients for each frame of the audio signal

1.2 Classification: ML models

Boruta method for feature selection and 10 fold cross validated of the training set.

Random Forest

- Max depth: [None, 10 , 20]
- N estimetors: [50, 100, 200, 300]

K-NN

- N neighbors: [3, 5, 7, 10]
- p: [1,2]

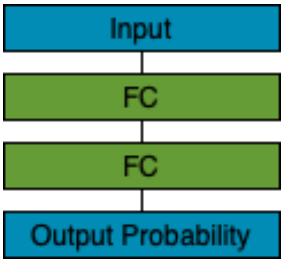
SVM

- C: [100, 500, 1000]
- Gamma: [0.005, 0.01, 0.1, 0.5, 1.0]

1.2 Classification: DL models

We tried three architectures from scratch to compare the performance.
All the networks started with random weight, and no transfer learning was involved.
Validated on 0.2 of the training set.

ANN



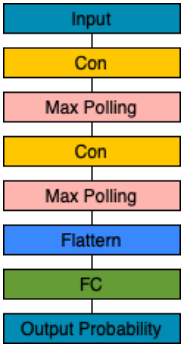
Parameters

- Epoch = 1000 (early stop)
- Batch size = 32
- Learning rate = 1e-3
- Optimizer = Adam
- Decay = 0.8 (every 10 epoch)
- Drop out = 0.5 (every fc layer)
- Loss function = sparse categorical cross entropy

1.2 Classification: DL models

Convolutional neural network should have better performance thanks to the convolutional layers.

CNN



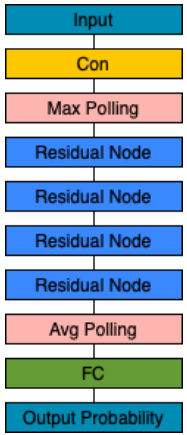
Parameters

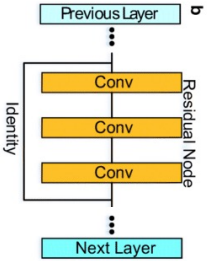
- Epoch = 1000 (early stop)
- Batch size = 32
- Learning rate = 1e-4
- Optimizer = Adam
- Decay = 0.8 (every 10 epoch)
- Regularizer = 1e-2
- Loss function = sparse categorical cross entropy

1.2 Classification : DL models

Residual blocks allow the model to learn the difference form identity and output of convolutions.

ResNN





Parameters

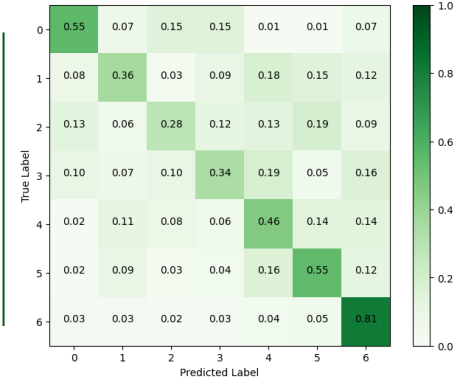
- Epoch = 1000 (early stop)
- Batch size = 32
- Learning rate = 1e-4
- Optimizer = Adam
- Decay = 0.8 (every 10 epoch)
- Regularizer = 1e-2
- Loss function = sparse categorical cross entropy

1.3 Results

The feature achieve similar result as the MFCC, the deep learning architecture don't have a big impact with this sets of features.

	RF	K-NN	SVM	ANN	CNN	ResNN
Features	0.478	0.443	**0.502**	0.450	0.432	0.455
MFCC	0.444	0.402	0.435	0.432	0.437	0.431

The features have in general always better results then the MFCC.



Outline: Bi-Dimensional Signal Classification

1. Mono-Dimensional Signal Classification
2. Bi-Dimensional Signal Classification
    1. Data and Preproccessing
    2. Classification
    3. Results
3. StarGAN Model

2.1 Data and Preprocessing

Used a kaggle dataset [4] where the images were collected by scrapping social nets as Facebook and Instagram, scrapping YouTube videos and already available datasets as IMDB and AffectNet.

This may cause some quality issues in the images.

Class

- Angry - (9.2%)
- Happy - (26.2%)
- Sad - (28.3%)
- Neutral - (27.6%)
- Surprise - (8.8%)

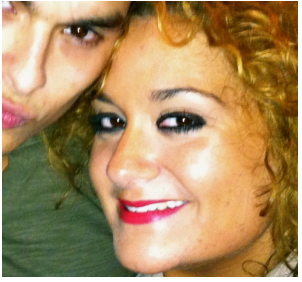
Unbalanced class distribution was addressed through undersampling to match the less numerous class.

~5,500

[4] Facial Emotion Recognition Image Dataset, Sujay Kapadnis (https://www.kaggle.com/datasets/sujaykapadnis/emotion-recognition-dataset)

2.1 Data and Preprocessing

Data poses several challenges, such as the presence of multiple faces in the same image, obscured parts, and poor image quality. We have chosen not to tackle these issues, understanding that doing so could negatively impact our performance.

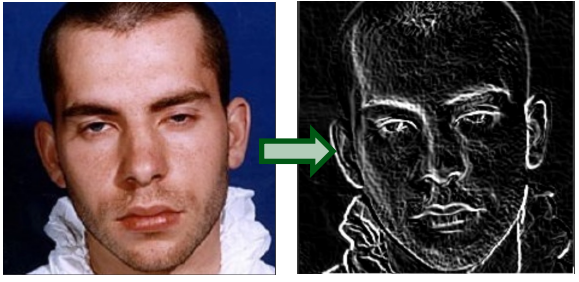


2.1 Data and Preprocessing

We hypothesized that facial emotions could be understood solely by considering the edges of faces.
Therefore, we employed edge extraction on our dataset and trained the classifiers using these processed images as well.

Edge extraction

- Image in gray scale
- **Scharr kernels** on both directions (x/y)
- Separate application of both kernels
- Extraction of vertical and horizontal edges
- Chopped to obtain the final image

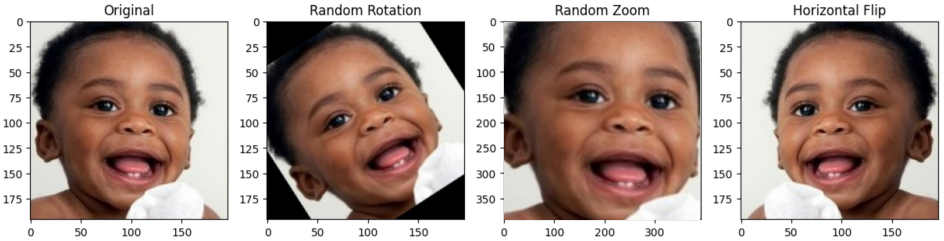


2.2 Classification: Data agumentation

Data augmentation allow to enhance generalizability, expand the data dimensions, and mitigate overfitting.

Strategies

- Rotation = 40
- Zoom range = 0.2
- Horizontal flip

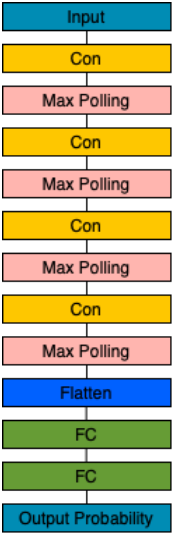


2.2 Classification: CNN

Model built and trained from scratch. Validated on 20% of the training set.

Parameters

- Epoch = 50 (early stop)
- Batch size = 64
- Learning rate = 1e-4
- Optimizer = Adam
- Decay = 0.5 (Every 10 epoch)
- Dropout = 0.25 every fc layer
- Loss function = Categorical cross entropy



2.3 Classification : ResNet50 fine tuning

ResNet50 in fine tuning (ImageNet weights).
Freeze weights to the last convutional block of pretrained models.

Architecture Modify

- \- Input layer
- \- Output layer
- \+ Input layer (224, 224, 3)
- \+ Global average pooling 2D
- \+ Dropout 0.3 (to reduce overfitting)
- \+ Fully connected layer, with softamax (5)

Parameter

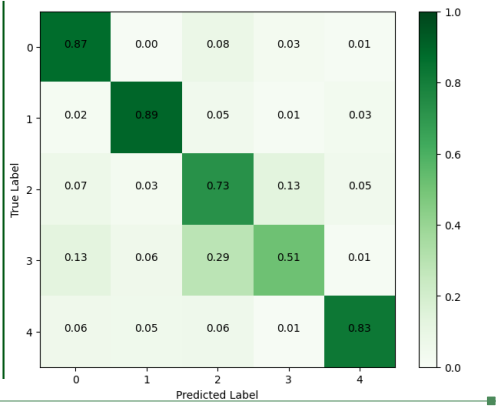
- Epoch = 50 (early stop)
- Batch size = 64
- Learning rate = 1e-3
- Optimizer = Adam
- Decay = 0.5 (every 10 epoch)
- Regularizer = 1E-2 (last layer)
- Loss function = Categorical cross entropy

Tried to unfreeze all the network, assigning a lower learning rate (0.0001) to the layer until the last convolutional block. This greatly improved the performance.

2.3 Results

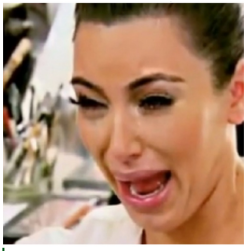
The unfreezing strategy has proven to be successful. Both RGB and the image with only the edges exhibit similar performance in the best model, affirming that our idea was correct.

	RGB	Edges
CNN	0.5632	0.5865
ResNet50 (freezed)	0.6272	0.5387
ResNet50 (unfreezed)	0.7423	**0.7669**

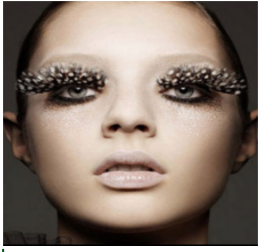


2.1 Results

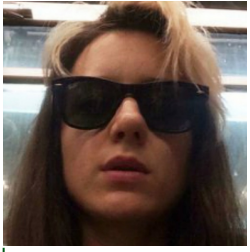
Some facial features like sunglasses or extreme expressions can lead the model to misinterpret the emotions. Additionally, children are more likely to be misclassified, probably due to their different facial proportions.



Gt: Sad
Pred: Angry



Gt: Neutral
Pred: Angry



Gt: Neutral
Pred: Happy

Gt: Happy
Pred: Angry

Outline: StarGAN Model

1. Mono-Dimensional Signal Processing
2. Bi-Dimensional Signal Processing
3. StarGAN Model
    1. From CycleGAN to StarGAN
    2. Training
    3. Results

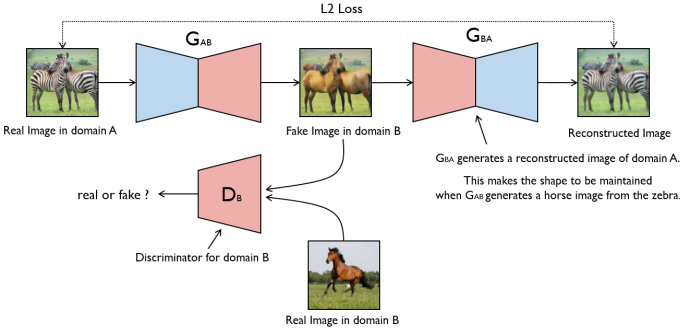
3.1 From CycleGAN to StarGAN

GANs can be applied to the translation of images. The simplest approach train an autoencoder to translate the photo, used for paired image to image translation.

In some other cases all we have in terms of data is two data sets which don't overlap at all.

Cycle GAN

- 1°Generator: translate the image
- Discriminators: real or fake
- 2°Generator: translated image still looks like the original in some way.

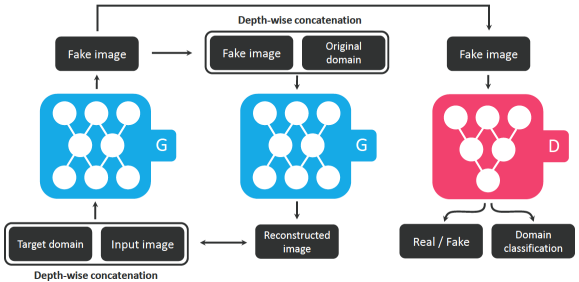


3.1 From Cycle GAN to StarGAN

Standard architectures: 4 generators and 2 discriminator for every pairs.
StargGAN: single generator and discriminator. Use label for a target class (1-hot-vector), appended as an additional channel. Discriminator is now also tasked with outputting a classification.

SrarGAN Learning Objectives

- How much the generated image looks like a real image
- How much a generated image looks like the target domain in comparison to the target label
- Translate it back into the original image, and penalized for differences btw the original and reconstructed images



3.2 Training

StarGAN [5] Discriminator and Generator are trained alternately (5 to 1).

Objective Functions

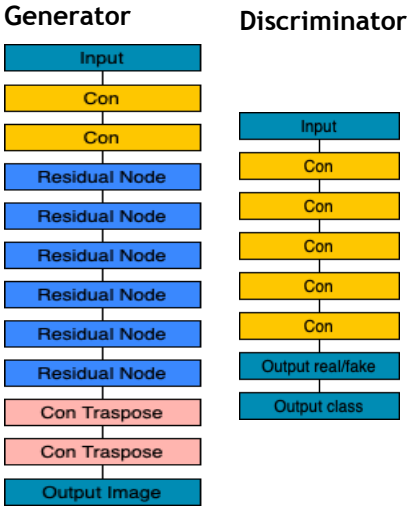
- Discriminator: $L_D = -L_{adv} + \lambda_{cls} L^r_{cls}$
- Generator: $L_G = L_{adv} + \lambda_{cls} L^f_{cls} + \lambda_{rec} L_{rec}$

Where: $\lambda_{cls} = 1$ and $\lambda_{rec} = 10$

Adversal loss use **Wasserstein GAN objective** with gradient penality, for stability and higher quality

Training Parameters

200.000 iteration, batch size 16
Lr 0.0001 for first 10 epoch, linerly decay after



[5] Choi, Yunjey, et al. "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation."

3.4 Results

The outcomes align with expectations. The poorest results, indicated by surprised facial expressions, likely stem from a lack of training images.

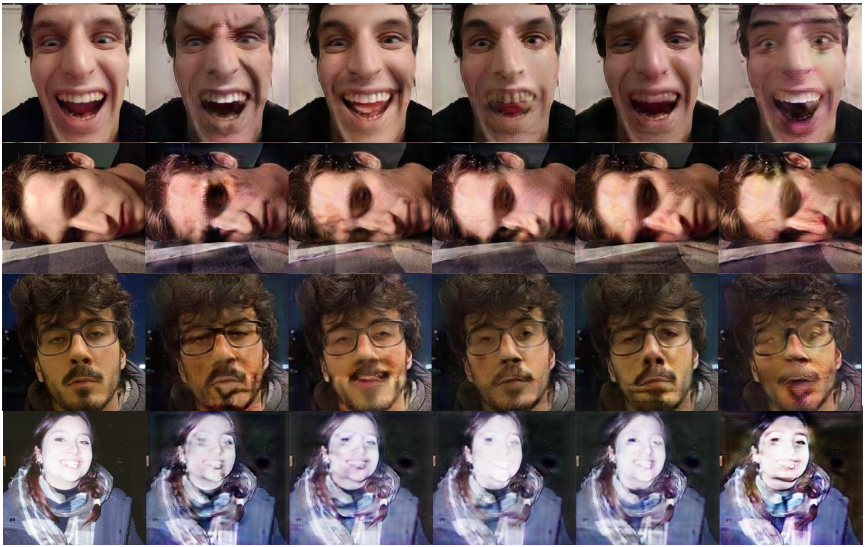
	😡	😄	😐	😢	😳
FID	66.6	40.6	45.0	47.9	73.0

3.4 Results

Problems

Unfavorable outcome due to:

- Poor image quality
- Extreme expressions
- Presence of facial hair and glasses
- Profile or sideway
- Wider field images



Thanks for the attention!
Questions?