Project

Digital Signal And Image Management

Agazzi Ruben 844736

Sommario—This project is divided in three parts: audio classification were I tried to classify urban sounds into 10 different classes using machine learning algorithms, image classification where I tried to classify images belonging to 10 different classes like ”Forest”, ”Building”, etc... and content based image retrieval on food images from a dataset containing 10 types of food.

INDICE

1 AUDIO CLASSIFICATION

The first task of the audio consists in the classification of 1-d audio signals into 10 different classes. I solved this task by doing some preprocessing of the dataset, feature extraction and finally classification using the features with a machine learning algorithm.

1.1 Dataset

The dataset can be found on Kaggle and consists in sounds recorded in an urban environment.
The dataset contains 10 different classes:

- Air conditioner
- Car horn
- Children playing
- Dog bark
- Engine Idling
- Gun shot
- Jackhammer
- Siren
- Street music

The dataset contains 8732 labeled sounds, and all the sound have a duration that is less or equal to 4 seconds.

1.2 Preprocessing

After an initial exploratory analysis of the audio data I found out that some audio were mono channel and some were stereo audio. For the stereo audio I took the first channel in order to process the data in an equal way for all the entries of the dataset.

1.3 Feature extraction

For the phase of feature extraction I decided to extract the following features:

- Energy: is defined as the sum of the squared values of the audio $E = \sum_{n=-\infty}^{\infty} |x(n)|^2$
- Zero Crossing Rate: is defined as the number that the signal switch from a positive to a negative value or vice versa, divided by the length of the signal
- Spectrogram: is a visual representation, or 2-d representation, of the spectrum of frequencies of a signal as it varies with time[**spectogram**]
- Mel Spectrogram: the Mel spectogram is the spectogram with its value scaled by the Mel scale. The Mel scale was introduced because humans do not easily tell the difference in the same ways for all the heard frequencies.
- MFCC features: is a representation of the short-term power spectrum of a sound.

Some features, like the Mel spectogram, change according to the duration of the audio, but we cannot have features with different sizes for the classification part, so I decided to take the first 1 second for the extraction of these features.

1.4 Classification

After extracting the features I proceeded to the classification task. In this case i trained a Support Vector Machine and a Multi Layer Perceptron in order to compare them.

1.4.1 Support Vector Machines

Support vector machines are a type of machine learning algorithms that tries to classify data by triyng to linearly separate them. Because, in most of the cases, data are not linearly separable, it first applies a transformation to the data, for example it adds some dimensions to the data, in order to make them more easily separable

For the training of the Support Vector Machine I used the python library sklearn, in particular the LinearSVC classifier. I also tried different configurations for the SVC parameters:

- C parameter: Is a regularization parameter, and the strenght of the regularization is inversely proportianl to its value. In this case the best parameter found was

After training the classifier I tried it in the test dataset in order to evaluate it's performance. The accuracy of the classifier was equal to 69% and the following values abount precision, recall and f1 on the 10 classes:

Class	Precision	Recall	F1
0	0.91	0.84	0.88
1	1.00	0.31	0.47
2	0.69	0.58	0.63
3	0.30	0.83	0.44
4	0.84	0.64	0.73
5	0.93	0.83	0.87
6	1.00	0.33	0.50
7	0.96	0.85	0.90
8	0.94	0.67	0.78
9	0.58	0.54	0.56

Tabella 1: Table of Precision, Recall and F1 measure on every class computed on the test dataset

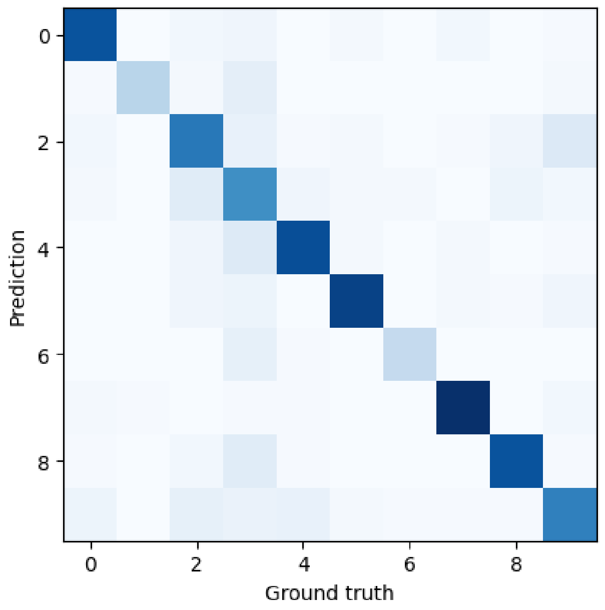


Figura 1: Confusion matrix of the classifier over the test dataset

1.4.2 Multi Layer Perceptron

The multi layer perceptron used is a simlpe fully connected neural network composed by 7 layers:

- First dense layer: Dense layer with 512 neurons with ReLU activation

- Second dense layer: Dense layer with 256 neurons with ReLU activation
- Third dense layer: Dense layer with 128 neurons with ReLU activation
- First dropout: Dropout layer with probability of dropout of 0.2 to add regularization
- Fourth dense layer: Dense layer with 64 neurons with ReLU activation
- Second dropout: Dropout layer with probability of dropout of 0.1 to add regularization
- Output layer: Dense layer with 10 neurons with softmax activation to perform classification

The model reached an accuracy of 59.8% on the validation set and an accuracy of 60% on the test dataset. The model obtrained the followed precision, recall and F1 measure over the 10 classes:

Class	Precision	Recall	F1
0	0.88	0.52	0.65
1	0.85	0.67	0.75
2	0.66	0.31	0.42
3	0.67	0.58	0.62
4	0.83	0.66	0.74
5	0.28	0.91	0.43
6	0.96	0.70	0.81
7	0.83	0.64	0.72
8	0.85	0.61	0.71
9	0.75	0.45	0.56

Tabella 2: Table of Precision, Recall and F1 measure on every class computed on the test dataset

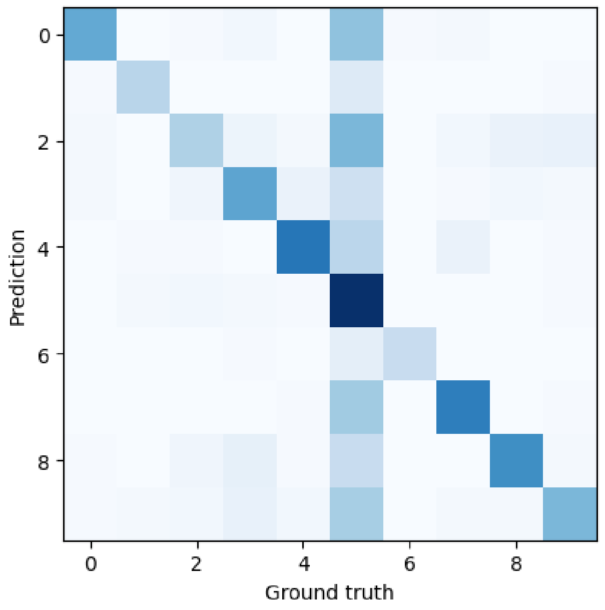


Figura 2: Confusion matrix of the classifier over the test dataset

2 Image classification

For the image classification I tried to clasify images beloging to 6 different classes using a custom Convolutional Neural Network and using a pretrained Convolutional Neural Network.

2.1 Dataset

The dataset is the Intel Image Classification available in Kaggle, it consists in 25000 rgb images of size 150x150 divided in 6 classes. The classes are:

- Buildings
- Glacier
- Mountain
- Sea
- Street

2.2 Convolutional Neural Networks

Convolutional Neural Networrks are a class or artificial neural networks, most commonly used to analyze images[**CNNs**]. They are deep learning algorithms presenting convolutional layers: theese layers are composed by a fixed number of convolutional filters of a given dimension. The network during the training phase try to learn the weight of the components of the convolutional filters

2.3 Custom CNN

For the classification problem I designed a custom CNN that I later trained using the training data. The Network is composed by 14 layers:

- Input layer: Input layer for images of size 150x150x3
- First Convolutional layer: Convolutional layer composed by 64 convolutional filters of size 3x3 with ReLU activation function.
- First average pooling: Average pooling layer with filter size 2x2
- Second Convolutional layer: Convolutional layer composed by 32 convolutional filters of size 3x3 with ReLU activation function.
- Second average pooling: Average pooling layer with filter size 2x2
- Third Convolutional layer: Convolutional layer composed by 16 convolutional filters of size 3x3 with ReLU activation function.
- Third average pooling: Average pooling layer with filter size 2x2
- First flatten layer: layer used to flatten the output of convolutional layers in order to have a vector of features to pass to the fully connected part of the network
- First dense layer: Dense layer with 512 neurons with ReLU activation function
- First dropout layer: Dropout layer with probability of 0.1 to prevent overfitting
- Second dense layer: Dense layer with 256 neurons with ReLU activation function
- Second dropout layer: Dropout layer with probability of 0.1 to prevent overfitting
- Third dense layer: Dense layer with 64 neurons with ReLU activation function
- Output layer: Dense layer with 6 neurons and softmax activation function in order to make the classification

The final network has a total of 2,539,894 trainable parameters. I trained the network on the training dataset for 128 epochs obtaining an accuracy on the validation set equal to 76%.

2.4 Pretrained CNN

In this other try i used a pretrained CNN, the VGG16 network using the ImageNet weights. I fine tuned the network in order to classify the data: I removed the initial input layer because the input size was greater than the dimension of the images of the dataset, and then I added 4 Dense layers at the end of the network that I trained to classify the images, in practice I use all the VGG16 network as a feature extractor. With this model I obtained an accuracy of 83.06% on the validation dataset and of 81% over the test set. For the training of this model I also added a step of data augmentation in order to generate more images by sheering, rotating, zoomming and changing the brightness of the original images. The classifier obtained the following metrics over the 6 classes:

Class	Precision	Recall	F1
0	0.85	0.92	0.89
1	0.97	0.94	0.95
2	0.68	0.84	0.75
3	0.88	0.46	0.60
4	0.69	0.85	0.76
5	0.89	0.88	0.89

Tabella 3: Table of Precision, Recall and F1 measure on every class computed on the test dataset

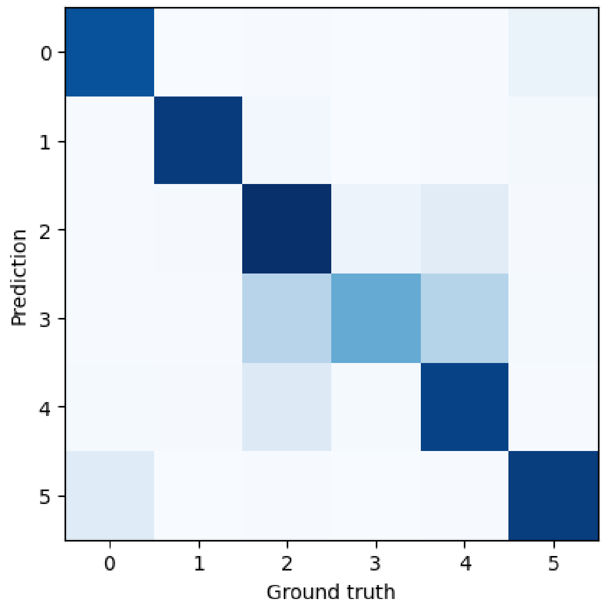
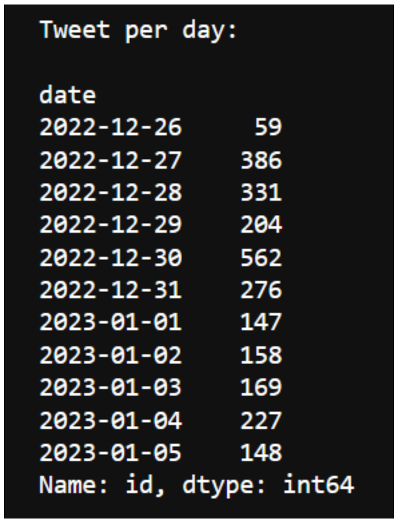


Figura 3: Confusion matrix of the classifier over the test dataset



```
Tweet per day:

date
2022-12-26     59
2022-12-27    386
2022-12-28    331
2022-12-29    204
2022-12-30    562
2022-12-31    276
2023-01-01    147
2023-01-02    158
2023-01-03    169
2023-01-04    227
2023-01-05    148
Name: id, dtype: int64
```

Figura 4: Number of Tweets per Day

3 Image retrieval

For the part of image retrieval I used the same pretrained network used in the previous task, the VGG16 model. For this phase I made two tests: I used features for the retrieval coming from the third pooling layer and from the last pooling layer.

3.1 Dataset

The dataset is available on kaggle and is the "Fast Food Classification Dataset - V2". It contains 20.000 images belonging to 10 different classes:

- Baked potato
- Burger
- Crispy Chicken
- Donut
- Fries
- Hot Dog
- Pizza
- Sandwich
- Taco
- Taquito

3.2 Third pooling layer

3.3 Last pooling layer

Riferimenti bibliografici

[1] Federico Bianchi, Debora Nozza e Dirk Hovy. *FEEL-IT: Emotion and Sentiment Classification for the Italian Language*. URL: https://github.com/MilaNLProc/feel-it.

[2] BSI. *New Smart Working Code of Practice announced*. URL: https://www.gov.uk/government/news/new-smart-working-code-of-practice-announced.

[3] Forbes. *Working From Home: VPN Use Reveals Longer Hours And May Hide Privacy Threat*. URL: https://www.forbes.com/sites/zakdoffman/2020/03/24/coronavirus-work-from-home-longer-hours-more-distractions-and-this-surprising-privacy-threat/?sh=5eb2527c7363.

[4] Ministero dell'Istruzione e del Merito. *Lavoro agile*. URL: https://miur.gov.it/lavoro-agile.

[5] *Modularity (networks)*. URL: https://en.wikipedia.org/wiki/Modularity_(networks).

[6] *NetworkX*. URL: https://networkx.org/.

[7] Rai News. *In Italia lo smart working è al palo: solo 2,9 milioni lavorano da casa*. URL: https://www.rainews.it/articoli/2022/07/gli-italiani-tornano-a-lavorare-in-ufficio-negli-altri-paesi-europei-sale-il-numero--c0201a04-5af8-40b4-b041-dbf89f4eebdc.html.

[8] Osservatori. *SMART WORKING - Il lavoro agile dalla teoria alla pratica*. URL: https://blog.osservatori.net/it_it/smart-working-cos-e-come-funziona-in-italia.

[9] Gazzetta Ufficiale. *Legge 22 maggio 2017, n. 81*. URL: https://www.gazzettaufficiale.it/eli/id/2017/06/13/17G00096/sg.