Classification of time signals by CNN using spectrogram

Thanh Toan, Truong: email

Hoang Hai, Pham: hhoang@student.fra-uas.de

Phan Bao Viet, Nguyen: email

Riyad-Ul-Islam: riyad-ul.islam@stud.fra-uas.de

Link to project:

*Abstract*—This electronic document is a “live” template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. *\*CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract*. (*Abstract*)

Keywords—component, formatting, style, styling, insert (key words)

# Introduction

Convolutional neural network (CNN) is a deep learning algorithm used to process the data of image. It is commonly used in computer vision as a classification technique to distinguish different objects. On the other hand, spectrogram is a representation method used to present three-dimension measured signals in two-dimensional diagram.

Based on the dataset provided by Professor Pech in the module Computational Intelligence at Frankfurt University of Applied Sciences (FRA-UAS), the goal of this project is to classify the reflected signals of different objects using CNN and spectrogram.

# Literature Review

## Gabor Transformation

Initially, all dataset provided for the subject are sets of analog signals in time domain. However, the interest of this project lies on the change of frequency spectrum with respect to time of the reflected signal from the object, or the change of frequency spectrum in different projection from the sensor to the object. In the path to pursue this goal, Gabor Transform [1] was used to convert the signal in time domain to time-frequency representations (TFRs).

In a quick overview, Gabor transform first filters the signals with a Gaussian window. The remain part of the signal from the filtering process would then undergo Fourier Transform. The filtering window shifted through a fixed number of timestamps every cycle until it reaches the end of the input sample. After applying Gabor Transform to the sample (1 sample = 1 time series of analog signal), the output set of Fourier transform for each window with specific begin timestamps formed together a TFRs. these TFRs can be presented in the forms of spectrogram. The following formula is the applied filter as discussed:



Figure 1 Gabor Transform formula

From the formular, the Fourier Transform and the Gaussian filter can be observed in two different multiplicators, with the Gaussian filter is the exponential which contains **τ**. The shift **τ** represent the time step of the Gaussian filter in each iteration.

For a more rigorous definition, this method belongs to a family of short-time Fourier transforms and was named after Dennis Gabor upon his introduction of it. The goal of Gabor transforms is to give a clear view on what is happening on the frequency characteristic (strength of sinusoidal frequency and phase) of a change signal in time domain [2].

## Convolutional Neural Network

The concept of neural network or artificial neural network is commonly known as a combination of different layers connected to each other to make decisions based on different types of input. Biologically speaking, the neural network is a technique that mimics approximately how a brain function. Each layer contains various nodes acts as a system of neurons that can interconnect between layers. Besides, dependent on the importance of each specific neuron, or node, a factor called weight is introduced to bias for the purpose of the system. These layers are commonly known as the hidden layer.

....

On going...

## Matlab

(Viet Nguyen)

# Methodology

## Overview

## Gabor Transform and Creation of Spectrogram

(Toan Truong)

## Convolutional Neural Network configuration

(Toan Truong)

## Graphical User Interface GUI

(Hai Pham)

A user-friendly GUI was created to implement the experiment of Gabor transform as well as CNN classifier. The app is named “GUI.mlapp”. The figure below shows the GUI of this project.

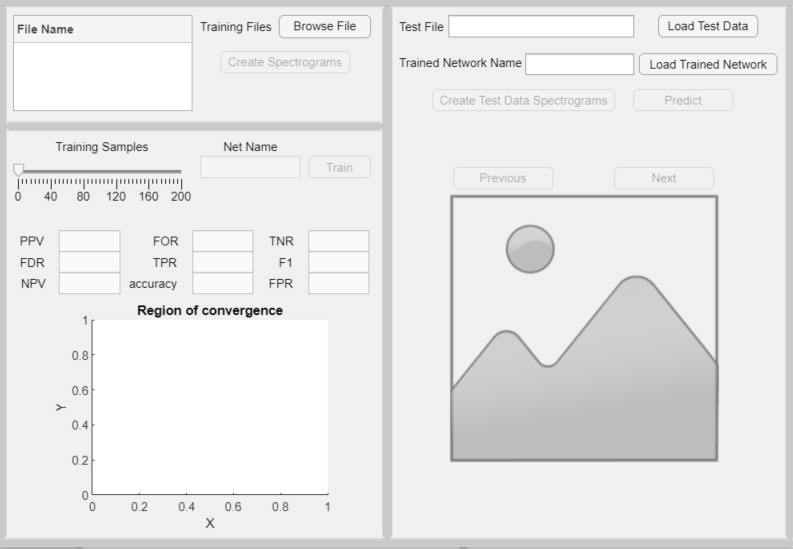
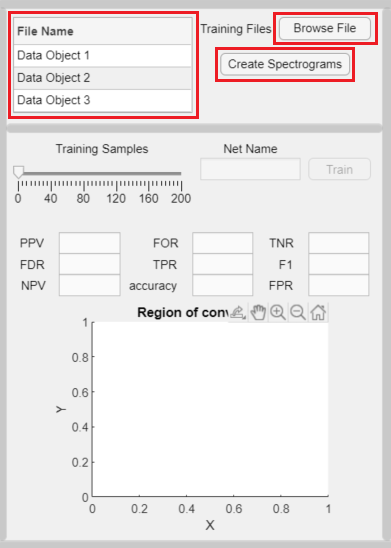


Figure [NUMBER] shows the GUI is divided into three sections:

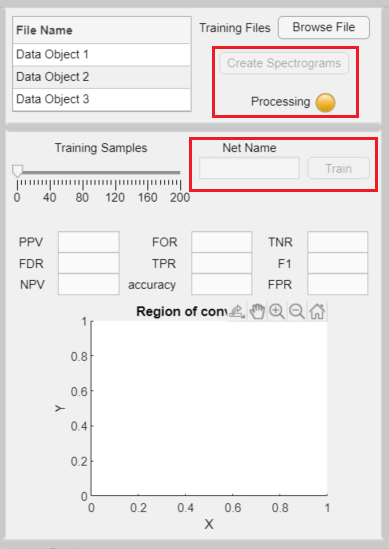
* The first section is located at the top left corner of the frame. This one is used for loading training data files and create spectrograms for each sample in the training data.
* The section which is used for training and validating the data is right under the first section.
* The last section is situated at the left panel of the GUI. This section indicates the predicted results of test data files.

The following is a short manual instruction in order to run the GUI application:

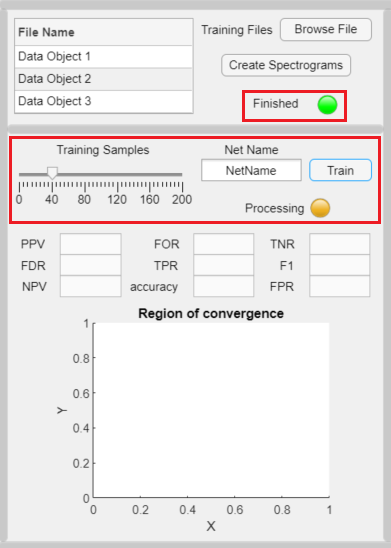
* ***Step 1***: First and foremost, user needs to load the training files by pressing the button “Browse File” as shown in figure [NUMBER]. Currently, the training dataset includes three files “Data Object 1”, “Data Object 2”, “Data Object 3” which are located inside folder “dataset”. User can choose one file, two files or three file files at the same time for training purpose. After choosing training files, the names of all chosen files are illustrated in the table to help the user can check it again.



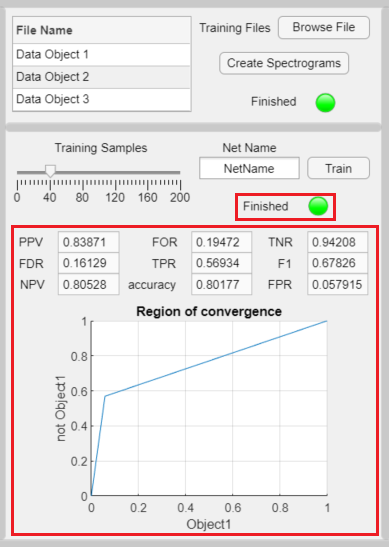
* ***Step 2***: Press the button “Create Spectrogram” to generate all spectrograms from the “Data Object” files. All the images are stored in folder “trainingData”, which is created automatically by Matlab. There is a LED in the lower area of the “Create Spectrogram” button to illustrate status of the process. As depicted in the figure [NUMBER], the LED shows orange and the status indicates “Processing” while the program is running. At this state, the “Net Name” field in section 2 is disabled.



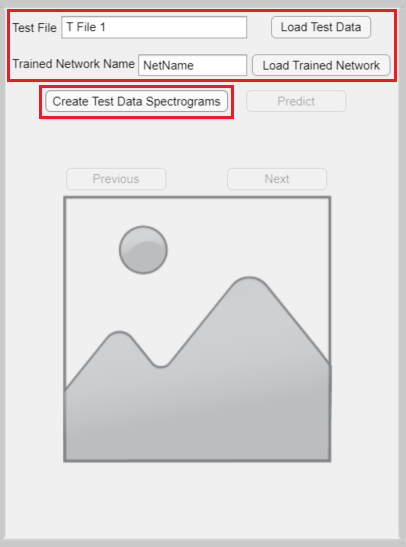
* ***Step 3***: After generating spectrograms, the LED changes to green and the status also alternates to “Finished”. There is also a notification to show that the process of creating images is finished. Then, the user moves to the second section for the training data purpose. Before pressing the “Train” button to start the training process, user must fill in the “Net Name” field as well as specify the “Training Samples” value using the slider as indicated in the figure [FIGURE]. A second orange LED with status “Processing” are also displayed.



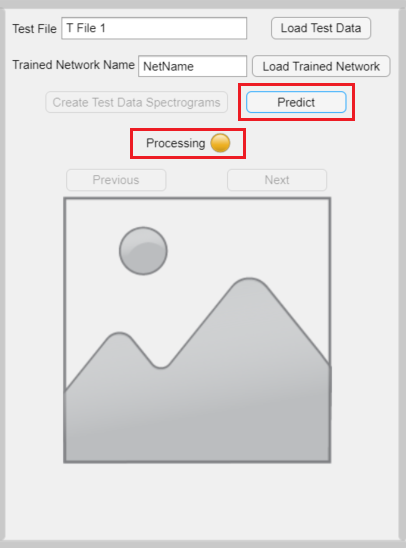
* ***Step 4***: The training process is done when the LED is green, and the status is “Finished”. The figure [NUMBER] illustrates the statistic validation as well as the graph region of convergence after the training process. The validation process also compels the data to have its label to generate the confusion matrix (pop up window) and the Region of Convergence (ROC). Validation needs a loaded CNN with appropriate resolution with respect to the images.



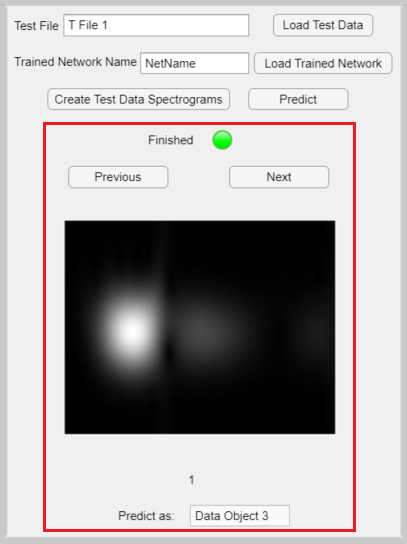
* ***Step 5***: Figure [NUMBER] shows the third section of the GUI. The “Test File” field requires the user to input the test .xlsx file, which is located in the “dataset” folder, by pressing the “Load Test Data” button. Then, the user presses the “Load Trained Network” button to load a trained network from the current directory, the name of the file “.mat” should be inputted. At the beginning, the “Create Test Data Spectrograms” is unable to press because there is no test data file. Therefore, the “Test File” and the “Trained Network Name” must be fulfilled to be able to proceed to the next step. Press the “Create Test Data Spectrograms” button when it is enabled.



* ***Step 6***: The data of test file is read into images under “testingData” folder, this data is then be loaded into the program. Eventually, the “Predict” button is able to be pressed. The predict process starts after the button is pressed and the third orange LED appears to show the “Processing” status as depicted in figure [NUMBER].



* ***Step 7***: Figure [NUMBER] depicts the result after the predict process is done. The LED turns to green and the status is “Finished”. The result pictures as well as the predict output are displayed in the lower area of the LED. The pictures can be browsed using the 2 buttons “Previous” and “Next”.



*NOTE*: If the user already has the trained network, step 1 to step 4 can be skipped and user can start directly at step 5 for the prediction purpose only. If the user only wants to train a new network, then please perform only step 1 to step 4 and ignore step 5, step 6 and step 7.

# Experiment

## Experiment’s result

(Riyad)

## Result on Given Test data

(Riyad)

# Conclusion

# Further Development

This Project has opened a variety of development to proceed due to its modular arrangement in code and design.

## Different types of Spectrogram:

With changes added to the output settings of each Spectrograms, the provided system can access a variety of different output networks. Instruction for the changes can be found in the above section.

It is also worth mentioning here that CNN can work with multiple representations of a signal. Aka. It can take 2 or more spectrograms as input for 1 label. This method also includes modification in the export of spectrogram. However, the step for including this code is not yet applied to the current state of the project.

Additionally, adjusting the output images in different size may also draw a better result in classification.

## Streaming Networks:

The current model CNN works on static data that was recorded and predict a static test dataset. This can be extended to predicting a real time reflected signal reading, with delays in multiples of window length. The real time graphing and predicting may be a possible achievement in an upcoming project from this state.

A notice here is that CNN may performs in lesser accuracy than the state-of-the-art technology in predicting real-time stream such as LSTM or RNN. Which is also a open path for development.

## Sufficient dataset:

Experimentally, the project was found to be lack of training data. The provided data (315 for ‘Data Object 1’, 200 for ‘Data Object 2’ and 400 for ‘Data Object 3’) is not clean and large enough for the sufficient training of the network.

The process of taking the data was not available from the authors at the time of this work. Creating a network with an understanding about the system would draw better results in designing a network for that system.

##### References

[1]"Gabor transform - Wikipedia", *En.wikipedia.org*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Gabor\_transform#cite\_note-1. [Accessed: 29- Sep- 2021].

[2] D. Gabor, Theory of Communication, Part 1, J. Inst. of Elect. Eng. Part III, Radio and Communication, vol 93, p. 429 1946 (<http://genesis.eecg.toronto.edu/gabor1946.pdf>)