# Time of day

Classification of Time Problem

Determine the time of the day based on the input image.

Computer Vision Project

The following Question must use a CNN based architecture.

# Link:

<https://github.com/Nolech7792>

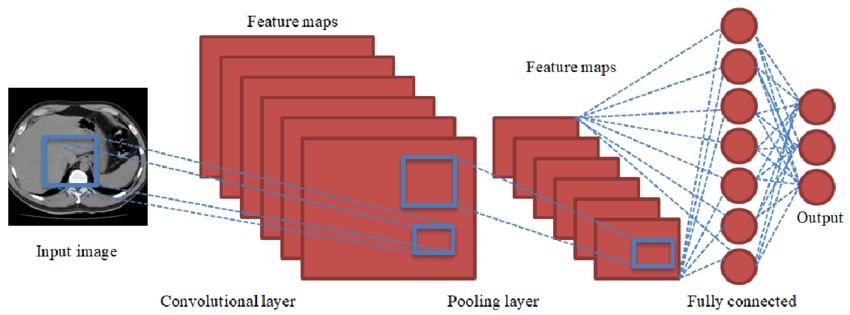
<https://github.com/samuelppd>

# Team Members:

|  |  |  |
| --- | --- | --- |
| Student Name | Student ID | Contribution in the project |
| Nolann Poisson | 73173 | Mutual coding |
| Samuel Poupardin | 73160 | Mutual coding |

# Model Architecture:

We used the Convolutional Neural Networks (CNN) architecture for this project. CNN is a class of Deep Neural Network. This method is an efficient to recognize and classify images and videos.



Our model is composed of four layers. Two of them are convolutional layers and two of them are linear layers :

* conv\_1 : hidden layer, 16 neurons
* conv\_2 : hidden layer, 32 neurons
* linear\_1 : hidden layer, 256 neurons
* linear\_2 : output layer, 10 neurons

We inspired ourselves from the model covered in class, but we also researched to enhance the model."

# Dataset Description:

The dataset we have is the time of day dataset. This dataset is composed of 3 classes :

* daylight : 978 images
* sunrise : 1040 images
* nighttime : 762 images

We ensured that the size of each image in the database is the same (224x224), and we made sure to transform the database into a Tensor. We used a bach\_size of 625 images in this CNN method.

A sandy beach with blue water and clouds in the sky

Description automatically generatedA city skyline with lights reflecting on water

Description automatically generated

# Methodology:

Training parameters:

* Number of epochs : 50
* Learning rate : 0.001
* Optimizer : Adam Optimizer

The loss function we used for training is CrossEntropyLoss()

We changed the batch size multiple times. After numerous attempts, we concluded that the optimal parameter was a batch size of 625. The accuracy and loss stopped changing after a certain number of epochs. In our case, 50 epochs were sufficient.

# Results:

The training began with a loss of 1.3425 and an accuracy of 79.74%. The confusion matrix looked like this:

A screenshot of a computer

Description automatically generated

After 50 epochs, the training had a loss of 0.0094 and an accuracy of 99.74%. The confusion matrix looked like this:

A screenshot of a computer

Description automatically generated

Finally, the tests showed excellent accuracy ranging between 99.84% and 100%. Below are the confusion matrices for the tests.

A screenshot of a computer program

Description automatically generated