DATASET FOR ARTIFICIAL INTELLIGENCE

Labelling data

LAB UNIVERSITY OF APPLIED
SCIENCES LTD
Faculty of Technology
Bachelor of Engineering, Information and
Communications Technology
Bachelor's Thesis
Spring 2020
Rafael Esparza Tortosa

Abstract

Author(s)	Type of publication	Published
Esparza Tortosa, Rafael	Bachelor's thesis	Spring 2020
	Number of pages	
	36	

Title of publication

Dataset for artificial intelligence
Labelling data

Name of Degree

Bachelor of Engineering, Information and Communications Technology

Abstract

Artificial intelligence programs need to be fed with a set of data which allows them to learn patterns in order to solve the required task. These sets must be collected and prepared to be useful for training. For this project the task was to collect a set of image data to later label them so that they can be used to train a program that can predict the position of a person in an image.

The theoretical section of the thesis deals with artificial intelligence and its importance, and techniques to deal with an image classification problem. The practical section describes the preparation of the dataset.

Keywords

dataset, java, labelling, machine learning, artificial intelligence

Glossary

AI	Artificial intelligence
API	Application programming interface
Axon	A long, thread-like structure attached to a nerve cell that sends out signals away from the nerve cell.
BLOB	A Binary Large Object
CNN	Convolutional neural network
Continuous variable	It can take a certain value within a certain interval.
DB	Database
Discrete Variable	It does not accept any value, only those belonging to the set.
Dendrite	A short branch-like structure which is located at the end of a nerve cell. It receives signals from other cells and transmits them to the cell body.
Generalization	Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.
IDE	Integrated Development Environment
JDK	Java Development Kit
JRE	Java Runtime Environment
JPG	Is a commonly used method of lossy compression for digital images
ML	Machine Learning
OOP	Object Oriented Paradigm
PNG	Is a raster graphics file format
RGB	Color values are supported in all browsers. An RGB color value is specified with RGB(red, gree, blue).
Synapse	A junction between two nerve cells where nerve impulses are sent across the minute gap by neurotransmitters.
UI	User Interface
Weight	A coefficient for a feature in a linear model or an edge in a deep network. The objective of training a linear model is to determine the ideal weight for each feature. If a weight is zero, then its characteristic does not contribute to the model.

CONTENTS

1 INTRODUCTION 4
1.1 Document organization 4
1.2 Project scope 5
2 ARTIFICIAL INTELLIGENT 6
3 MACHINE LEARNING 8
3.1 Supervised learning 9
3.2 Unsupervised learning 10
3.3 Reinforcement learning 11
3.4 Learning problems 12
3.4.1 Overfitting 12
3.4.2 Underfitting 13
3.5 AI Model 14
3.5.1 Neural networks 15
3.5.2 Convolutional Neural Networks 17
3.6 Datasets 20
3.6.1 Labelling images 21
3.6.2 Training data 22
3.6.3 Validation data 22
3.6.4 Test data 22
3.6.5 K-Fold cross validation 23
4 TECHNOLOGIES 24
4.1 Java and JavaFX 24
4.2 SQL and MySQL 26
5 CASES 28
5.1 Photos taken 28
5.2 Database 30
5.3 Labeller tool 31
5.4 Case conclusion 33
6 CONCLUSION 34
LIST OF REFERENCES 35
APPENDICES 39

1 INTRODUCTION

In 1950 began the interest in the artificial intelligence field when the British Alan Turing, mathematician, one of first computer scientists, logician, philosopher, logician, cryptanalyst and theorical biologist, developed the Turing test. This test consists in one human, the evaluator, and two computers making questions using only a text communication. One of these conversations is with a machine and the another with other human if the evaluator cannot guess what conversation is being done with the machine, the machine passes the test.

Since these days, the human has continued challenging the machines and their capabilities performing different tasks and winning challenges such IBM Deep Blue machine beating the then chess champion in 1996 or the victory from the Google AlphaGo AI to the GO champion in 2016. GO is a popular game in Asia and known for its difficulty (Serrano 2012, 2).

Nowadays, this approach is not the main focus. There are a lot of common areas in which AI is applied such as learning, perception or solving problems or other specific sectors like plying games, medical diagnoses or even driving cars. It is not difficult to find AI in our daily life but the people do not notice about that. Platforms like Netflix or Spotify so integrated into modern lives are using these technologies for their recommendation algorithms. In some hospital helping the doctors to obtain better diagnoses, automatic translation or image enhancement.

According to (Mauri 2020), the most worked topics lately in this field are the deep learning for image processing and natural language processing so that companies can implement customer service and technical assistance systems. Although this can already be seen on many company websites they are still to be improved. More and more corporations are starting to invest in the development of technologies with AI and the processing capacity of the devices currently allows the advances to emerge in a faster way if we compare it to the situation a few years ago.

One of the tasks being worked on in relation to this project is also the extraction of keywords associated with images in order to make classifications since image labelling is an essential and time-consuming task for the training of machine learning models (Zaforas 2018).

1.1 Document organization

The document is divided into six chapters, one with which I intend to introduce the subject presenting a brief background on what artificial intelligence is and its uses, then a point about machine learning where I will try to explain some of the ways there are to train a

machine learning model along with the most common problems that usually appear and the datasets needed to perform such training.

The tools and technologies used to make the practical case of the project will be discussed in chapter 4.

Finally, in the fifth point, the practical case of the project will be explained.

1.2 Project scope

This project has been suggested by my tutor Ismo Jakonen and Ville Teräväinen who have been guiding me to organize the work and solve doubts regarding the more specific subject. This project has been based on the process of obtaining a collection images of a person in different poses and angles to increase the database that Ville's working group has already created. These images are intended to train a machine learning model to predict the pose in which a person appears in order to improve the model that detected if a human has fallen down or not.

In this case we had to take pictures to create the dataset, create a database and tag the images. After this, I will remark what is a **machine learning** model and which type of them would be suitable to address an image classifier problem. All this is explained in depth in the following chapters.

2 ARTIFICIAL INTELLIGENT

"AI is the study of how to make computers do things which, at the moment, people do better"

(Rich & knight 1991, 5)

The **Artificial Intelligence** represents a set of software studies: Logic, computer science and philosophy with the goal that the computer will be capable of performing tasks that in the past the humans thought are exclusively human tasks such as recognizing and perceiving spoken language, writing or gestures, the ability to learn, etc.

In the last years, the term AI has been one of the most popular words used in the technology environment because it is a new field of study with high expectations. Nowadays many tech giants such as Google, Nvidia or Microsoft used AI technology and invest huge amounts of money in AI research (Techworld 2019).

Its usefulness is wide, some successful examples nowadays can be:

- **Adobe DeepFont:** It is a tool to help designers with font identification using the curves and other characteristics of a given image. This image is examined and compared with a database of twenty thousand fonts (NVIDIA 2016).
- **Insilico Medicine:** This bioinformatic company uses a ML technique to find treatments for cancer and other serious age-related diseases (PRWEB 2015)
- **Google Lens:** Google lens is a free phone application for image recognition. It allows you to identify places or things like books, animals, or monuments and translate texts in real time using the camera.

Once some of the uses of AI have been mentioned, it is appropriate to comment on how it is possible for a machine to perform these tasks. This is possible through training and continuous learning using techniques called **Machine Learning** which will be explained later.

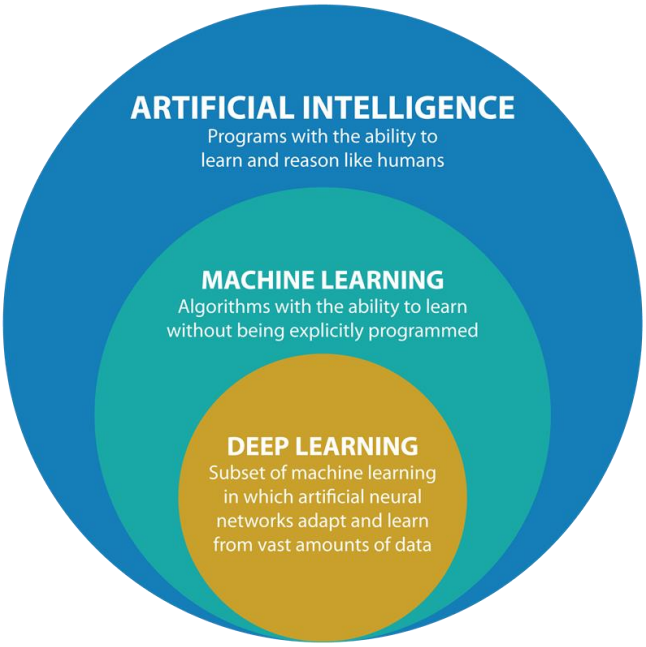

Figure 1 - Differences among AI, ML and DL

(ExperiencIA Oracle 2018)

One of the common confusions in this field is between the term artificial intelligence, machine learning and deep learning. Before continuing, it is appropriate to discuss the differences between these three terms.

As it can see in the Figure 1, **AI** covers the idea of programs that are capable of performing tasks that we humans determine require intelligence such as ours, which includes the ability to learn, understand and process the information obtained to achieve a certain purpose. This is possible through algorithms that with a previous learning process can give answers similar to those that a human would give. These learning techniques needed to be able to implement AI are called **Machine learning**. Through these techniques we want to get the machine to learn in such a way that it will be used as a basis to solve problems and more complex situations in the future. So, we could say that machine learning is a branch of AI (ExperiencIA Oracle 2018).

On the other hand, we have **deep learning** as an important technique of machine learning, which is based on emulating the learning that we humans use. It has a separate branch within the ML due to its special sophistication. This technique works through layers or neural units which will be treated in more detail in the chapter on neural networks (Bejerano Pablo 2017).

3 MACHINE LEARNING

"The more you know, the more you realize

you know nothing. "– Socrates

ML is the branch of AI where the objective is code systems capable of automatically learn and improve themselves from experience without human intervention. This field began to be relevant from the 1980s onwards when expert systems developed in this decade were adopted in the business sector, but in as early as 1957 Frank Rosenblatt and Donald Hebb developed the perceptron idea. This was installed in a machine called the Mark 1 perceptron with the objective of use in image recognition. The **perceptron** idea is based on a simple neural network used for linear classification problems. For this reason, this net is very restricted.

The high expectations of this machine with its high limitations due to perceptron, made the promised visual recognition goals unreachable and caused frustration. For this reason, research on ML did not resurface until the 1990s. Today, ML is considered to be responsible for some of the most significant advances in technology, including improvements in learning paradigms, the emergence of the Internet of Things, chatbots, spams filters, image classifiers, and more (Foote D. Keith 2019).

Learning is a quality that allows us to develop or modify skills and behaviour to better adapt to the environment, based on the information we obtain from the outside (Serrano 2012, 193). In the case of machines, the researchers and developers are trying to implement algorithms that resemble how we humans learn, the algorithms used in ML learn through a process called induction or inductive learning. Induction is a process of reasoning that can generalize from specific information. In this process, the generalization becomes important because one ML model needs to make predictions or decisions based on specific sets of data that were not seen during training.

Therefore, the steps to give knowledge to a machine could be summarized as: collect a large data set, create an ML model, use the data set to train a model and the difficult one, tune the parameters of the model to be able to obtain accurate predictions using a generalization similar to what humans do.

Some of the important paradigms in ML are **supervised learning**, **unsupervised learning**, and **reinforcement learning**. The most common paradigms used in classification problem are supervised and unsupervised learning.

3.1 Supervised learning

This paradigm is based on receiving as input tuples of **(data, label)** in the training phase so that we are providing the model with the information of which is the output we expect for each input, once this, the model will return as output the label that has been inferred with more probability. From this, the name of supervised appears, since, in a certain way, we are supervising the learning of the model by indicating the expected outputs. In this way, showing many examples to the algorithm, it should be able to make correct predictions even receiving unlabelled and unseen inputs in the training phase. This is possible by discovering a relationship between these input and output variables. For this reason, this paradigm is based on observation and should be able to generalize the knowledge obtained in the training phase. This is the basis of the supervised algorithms.

A simple example to understand could be a student who studies a series of questions together with their answers with the intention of being able to answer new questions on the same subject once they have been dominated (GoogleML 2020a).

About **advantages** of this technique, it is not dependent on the type of classification problem, and during training we can measure the accuracy of the classifier and we can stop training when we find it reasonable. On the other hand, it is necessary one person to visualize these data that will be used for learning and label them with the desired classes and that task takes a long time. In addition, we have that the training process is usually long and not infallible and the generalization of the classifier often depends on the choice of training cases.

Supervised learning is often used in (González 2018):

- **Classification problems:** These are usually used when the expected result is a discrete variable. For example, what type of animal appears in a photo or whether an email is spam or not.
- **Regression problems**: In this case, it is useful when the result can be a continuous variable. Some examples of its use can be to predict home prices or the income a customer will generate.

In this project, because the dataset collected has been intended to be able to determine the pose of a given person an image will be a multi class classification problem and hence the models that I have chosen to explain in the following sections will be the most used for this type of tasks, the neural networks.

3.2 Unsupervised learning

The idea of this paradigm is to generate knowledge only with the input data, without giving out the outputs that we expect. It is based on receiving input data to find patterns among them with the objective to split them in groups as the model learn. These input data, in contrast with the input data in supervised learning, it is not labelled as I said before, means that there are no classes defined a priori or if there are classes, they have not been assigned to the data yet (Santos 2016).

This grouping method is called **clustering** and is the basis of this paradigm. Therefore, clustering as a learning method has as objective classify in groups based on similarities among them (GoogleML a) . The name of these groups is clusters and the idea are that all the objects belonging to a cluster are more similar to each other than to the objects in the other clusters. An example of use could be to search for groups of customers with similar buying habits in order to make recommendations.

The **advantages** of this technique are that its capacity to evolve makes it easier to adapt to new and unexpected situations and can be trained without the need for preclassified data which makes the datasets less expensive to obtain.

In the part of the difficulties, we found that its implementation depends on each problem as opposed to the case of supervised learning and there are more complications when it comes to knowing what behaviour the classifier is having as it evolves with each case.

One of the most widely used algorithms in unsupervised learning is the K-Means. It is based on making clusters represented by one of the objects.

This algorithm is useful to identify clusters with a spherical shape, but it must be taken into account that the choice of the K, which would be the number of groups to specify should be indicated a priori. This means that an unfortunate choice of this value may result in the development of the grouping is not significant.

3.3 Reinforcement learning

An **intelligent agent** is a program that perceives its environment with the help of sensors and acts on that environment using actors, according to the perceived information the agent will decide. Depending on the type of problem to be solved, the agent can be aware of the environment or not. For example, in the case of a program which has to discover the path in a maze game, it will not be aware of the state of its environment unless it discovers it. (Vega 2018, 26).

Their goal is to learn to act in a way that maximizes the long-term rewards expected, so the agent must interact with the environment and learn by test and error to maximize their rewards and minimize their penalties as it shown in the Figure 2. In this way, a machine with no prior knowledge of the environment or the variables that are occurring can improve its technique to achieve the objective set.

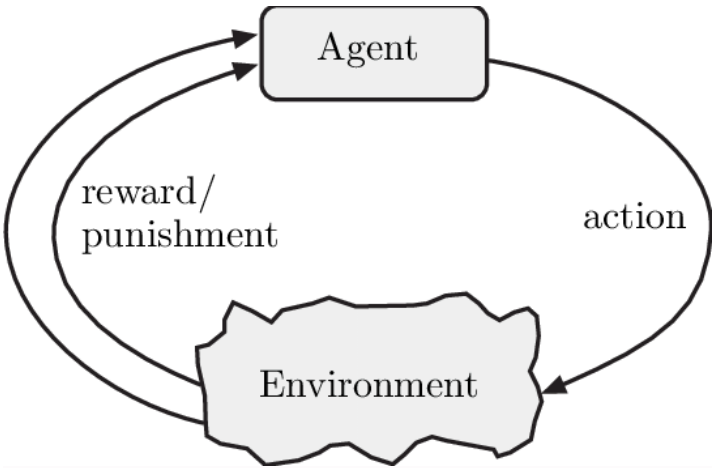

Figure 2 - Reinforcement learning diagram

The **main** parts of a reinforcement learning system are (Moni 2019):

- **Policy:** It is the main part of the agent which defines its behaviour. A policy would set out what actions should be taken given the perceived state of the environment.
- **Reward**: Every action taken by the agent is rewarded or penalized by the environment. The agent should look for the way to obtain the greatest amount of rewards since this will mean that he is getting the task done by the programmer. This value is normally a scalar value.
- **Value function**: reward signal indicates that it will be best in the short term, but in the long term the manager is the value function. Value represents the amount of reward the agent can expect in the future, starting from the current state.

- **Environment model**: this model imitates the behaviour of the environment and generally helps to infer how the environment will behave, for example, given a state and an action the model can predict the next state and the next reward. There are also trained model-free agents which are based on trial and error.

The field of robotics is one of the main fields benefiting from advances in this paradigm. An example of this is Dactyl, an OpenAI project which simulates a robotic hand in which you place a cube with the four faces of different color. As you test the cube, the robot hand manages to pivot or slide the cube between the fingers (OpenAI 2018).

3.4 Learning problems

Some of the most common problems when building an ML model arise when the model is being over trained and therefore tries to learn even the last details of the input data. Otherwise, appears problem when the model is not enough trained, or the choice of the model has not right for the problem that it was trying to solve.

These problems use to know as **overfitting** and **underfitting**. Both can lead a poor performance in our model, although overfitting is more common and difficult to detect.

3.4.1 Overfitting

An overfitted model gets a low loss during training but does not perform well when making predictions with new data. This occurs when the model tries to fit the training data too closely including noise, which leads to wrong patterns that will affect generalization. An overfitted model may appear to be working well in principle within the scope of the test, the problem comes when new data is received, the model will be not able to generalize and therefore its performance will not be as expected (Dot CSV 2019).

One way to find out if our model is overfitted, if we make a graph with the training error and the validation error we can see if there is a point where the validation error starts to rise while the training error continues decreasing. This means that the model begins to "memorize" the training data and therefore continues to improve but at the same time it is losing that capacity to generalize, which leads to a worse performance in predictions with new data. Some ways to prevent this can be (Blanco 2019):

- Have enough data for training and validation.
- In classification problems, having enough diversity of data from each class so as not to cause unbalanced learning.

- To have our dataset divided correctly to have enough data to test it and in case of overfitting to be able to detect it.

3.4.2 Underfitting

It is an easy problem to detect due to the poor performance of the model either because of lack of learning or because the model chosen is not the right one for the problem to be solved. Basically, what happens is that it does not fit the training data and therefore it is not able to fit the validation data either. Usually the solution to this problem is simply to find the right model or to let it train for a sufficiently long time (Dot CSV 2019).

In the Figure 3, you can see how it looks a model with different fittings:

in the first one, underfitting the line is totally straight, and it is basically useless because with this fit the probability of get right some prediction is not higher than someone trying guess saying random predictions.

In the second one, the model function is trying to adjust in the way where exists more points without getting all the points, so when the model will go to do a prediction, their probability to get right prediction will be satisfactory and it will also be able to avoid data that is noise due to the most correct points will be more nearly to the line.

The last one is with overfitting and it can be seeing clearly that the model had fitted almost every points, in this way the noise is directly affecting for the predictions and if new data is passed to the model at most probable will be the point is outside the correct place and the model do wrong prediction due to it will not be able to generalize.

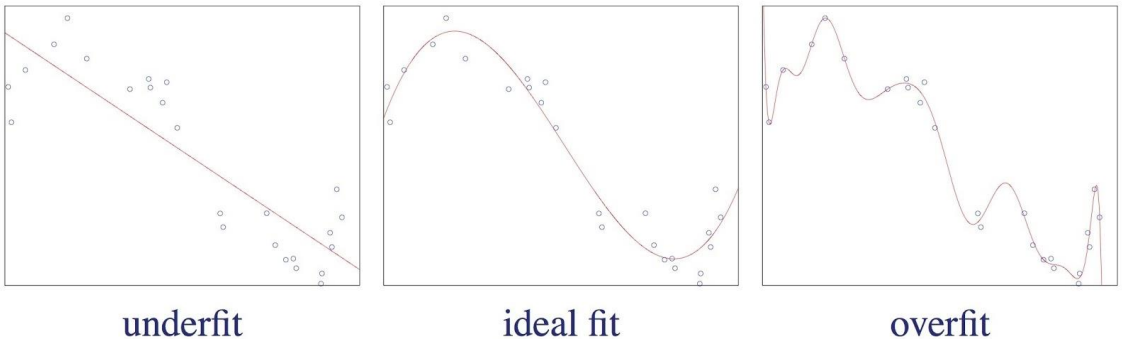

Figure 3 - Model with different fittings (Blanco 2019).

3.5 AI Model

To be able to make predictions you need to have created a program which by being trained is able to perform this task. In the case of classification problems, the model must learn how to split the points that represent different categories so that when new unknown data is introduced it can be classified in one of those classes.

Training an ML model requires a few steps until it is ready. Generally, these steps are similar to the ones described below (Aprendemachinelearning 2017):

1. **Collect the data**: We will have to find out how to obtain the data needed to feed the program and train it. For this you can use already existing databases or create one from zero. The model will be directly affected according to the quality of the data we have trained it with.

2. **Prepare the data**: This is the moment where we have to review the data we are going to use and separate them in different sets for training and validation generally.

3. **Choose the type of model we want to train:** There are different options to choose according to the problem we need to treat, some may be for binary classification, could be if a mail is spam or not, multiclass classification in order to determine what type is a product or regression models that help predict a numerical value (Amazon ML 2020a).

4. **Training**: This is the moment where the model will be generated. The learning algorithm must capture the patterns from the training data.

5. **Evaluation**: Once the training is finished, it should be checked that the performance of the training is as expected. This step, depending on the technology used to train the model, is carried out together with the training. It uses a validation dataset of which the algorithm should not know their correct answers. Normally we try to achieve over 90% accuracy to determine that the model is reliable.

6. **Parameter tuning**: Once the model is created, if the desired levels of accuracy are not obtained, we may have problems of overfitting or underfitting, which are explained later. There are several parameters which can be adjusted to solve these problems such as the learning rate or the epochs, which is the number of times we iterate our training data. In the case of neural networks, an important parameter may be the number of hidden layers and the number of neurons in each layer. This is the step that requires the most effort and with which we can obtain considerable improvement.

3.5.1 Neural networks

Neural networks are an effort to emulate the way the human brain works. They are a powerful tool for many applications and a challenging field of research.

In contrast to other models, neural networks are useful for solving non-linear problems such as computer vision, speech recognition and recommendation systems.

According to current neuroscience, a **neuron** functions somewhat similar to a **transistor**, in terms that a neuron can be very active or not active at all, which comes close to the fact that a neuron can be active or not active at all. For the transmission of information, two neurotransmitters come into play, dendrites and axons. The basic structure of a neuron is shown in Figure 4.

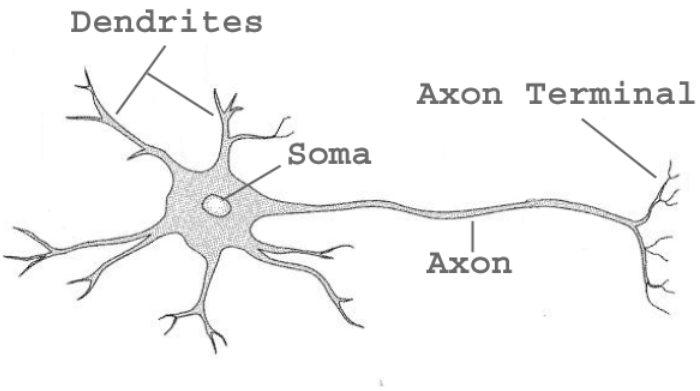

Figure 4 - Main structure of neuron (Salman 2009)

Neurons are activated by dendrites, which we can determine as the inputs to these. When the number of dendrites sending signals reaches a certain limit called the activation threshold, the neuron is activated and sends electrical signals to other neurons through the axons, which can be determined as outputs.

Before the neural networks appear, simple perceptron came into play. **Perceptron** is a model of a single neuron which simulates a single neuron. It receives input signals, simulating dendrites, and returns an output, simulating an axon. These inputs are limited to zeros or ones which will determine whether the perception is active or not. In addition, the weights (w) enter into play, each of the entries (x) is assigned a weight by which the input value is multiplied. The weights are values that the algorithm adjusts to give more or less importance to certain connections. The function of the output is shown below.

$$Z = \sum_{i} Xi \times Wi$$

If the resulting value is higher than the activation limit, then the neuron will be activated. The function of the weights is to adjust them through training to obtain the result that is nearest to the correct answer (Serrano 2012, 210-213).

As shown in Figure 5, the output will depend on the input, its weights and the activation limit of the neuron. This is one way of representing it, although there are other possible variations.

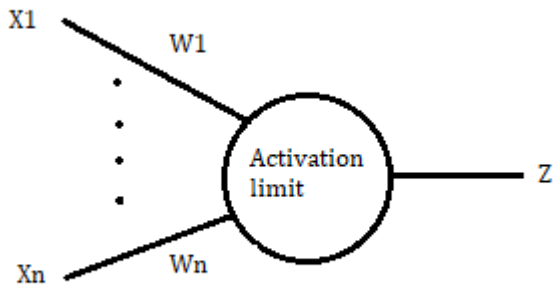

Figure 5 - Diagram of artificial neuron

The perceptron has a very limited use as it can only solve linearly separable problems. This is solved by using sets of neurons to achieve different levels of complexity when partitioning the plane, which are called hidden layers of neurons. Therefore, without the use of hidden layers it is only possible to solve linearly separable problems, with a single hidden layer, separable problems with a curve and with more than one layer, problems in which arbitrary separations appear (Serrano 2012, 208-211).

The diagram of Figure 6 shows where a layer would have to be introduced into the neural network to break the linearity and thus be able to treat more complex problems. We have the input connected to the next hidden layer of neurons which will be connected to a layer with which we will introduce a non-linear function, thanks to this the next layer will learn a more complex function than the previous one.

This function is called the activation function. This function takes the weighted sum of all the inputs from the previous layer and then generates a generally non-linear output value to the next layer. Different layers can have different activation functions.

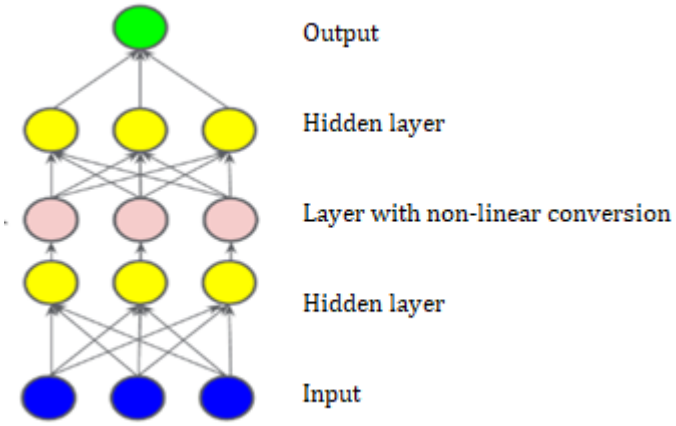

Figure 6 - Neural network diagram (Google ML 2020c)

Some of these commonly used activation functions are:

- **Sigmoid**: converts the weighted sum to probabilities by returning a value between 0 and 1.
- **ReLU:** Rectified linear units, usually gets better results than functions like sigmoid and is less expensive to calculate. Its operation is the following:
    - If the input is negative or zero, the result will be zero.
    - If the input is positive the result will be equal to the input.

Finally, focusing on the classification, we use a last layer with a SoftMax function, this function is used to assign decimal probabilities to each of the possible classes in our problem. When dealing with probabilities the sum of all of them must add up to one.

This function is implemented by a neural layer prior to the outcome layer and this layer should contain as many neurons or nodes as possible categories. It should be considered that this function is useful for problems where each example can only be a member of one class exactly (GoogleML 2020c).

3.5.2 Convolutional Neural Networks

CNNs are a class of artificial neural networks that are trained through supervised learning. CNNs contain several hidden layers that follow a hierarchy and specialize in detecting everything from simple shapes such as lines and curves to more complex shapes. For this it is

necessary to have a wide dataset of images with which to train in order to learn what attributes characterize an object and then be able to generalize. These input images are converted to an attribute map with the pixels of the raw image.

On the other hand, it is convenient to normalize the values of the colours, normally composed by three channels, RGB, of the pixels have values that go from 0 to 255 so the pixels are transformed using $\frac{Valor}{255}$ in order to obtain a value between 0 and 1.

A CNN is made up of a set of operations (Google ML 2020d):

- **Convolution**: This operation allows the network to extract new attributes or patterns from the images by applying different filters. The result of this operation is an attribute map that is lower than the attribute map of the original image and has only one dimension. The filters are arrays of dimensions smaller than the image that are sliding through the array of input attributes pixel by pixel to perform that extraction. The filters applied in the first layers will extract characteristics of a lower complexity, such as lines or shapes, while the filters applied in subsequent layers, may come to extract faces or bodies among other forms.
  As it can see in Figure 7, a colour image is represented by three planes where each one represents an RGB colour. The filter to be applied should also have one plane for each RGB. Once the multiplication to extract the attributes is done, the value of each pixel of each plane will be added to give a single plane output. During this process, CNN performs multiplications between the filter array and the corresponding part of the original one. In the case of the Figure 7, a filter size of 3x3 has been chosen so the first element of the output array will be the multiplication of the first 3x3 pixels of the input image.

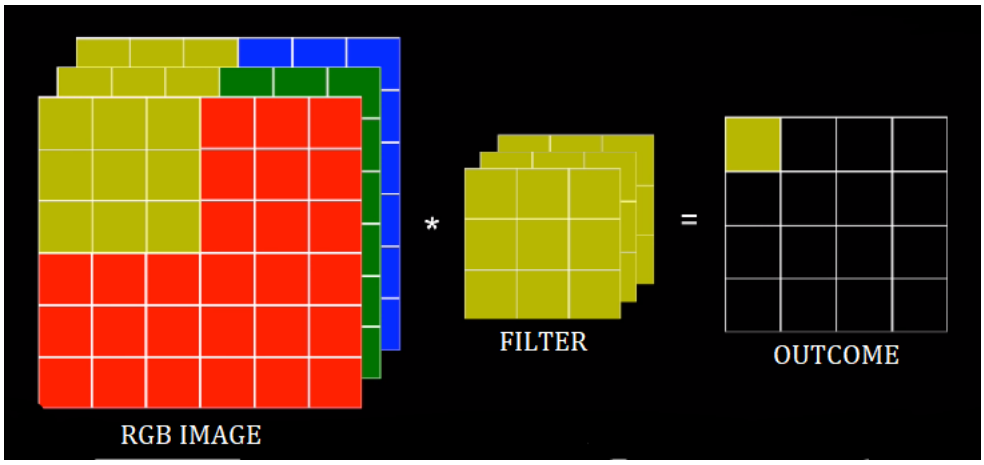

Figure 7 – Convolution (codificandobits 2019).

You have to indicate to the CNN the number of filters you want to apply and the size of the filters you want to use. This is an iterative operation and once it is finished the network will learn which values are the most optimal to extract significant attributes such as shapes or textures.

The computation of the filters is one of the most expensive operations of this type of network, so a greater number of filters to apply will cause the time of training to increase significantly.

- **Activation function**: A conversion will be applied to each attribute resulting from the convolution using a function to introduce a non-linearity to the model as discussed in the previous point. One of the most used functions is still ReLU which was also explained in the previous point.
- **Pooling**: The intention of this operation is to reduce the processing time by reducing the number of attributes to be processed. This is achieved with an algorithm called max pooling and works in a similar way to convolution. As you can see in Figure 8, from the resultant attribute map of a filter is extracted again another map of smaller attributes where only the attribute with higher value is extracted, the rest of the values will be discarded. For this algorithm it should indicate the size parameters of the pooling filter, commonly 2x2 and the segmentation, which represents the distance of pixels that will slide in each iteration.

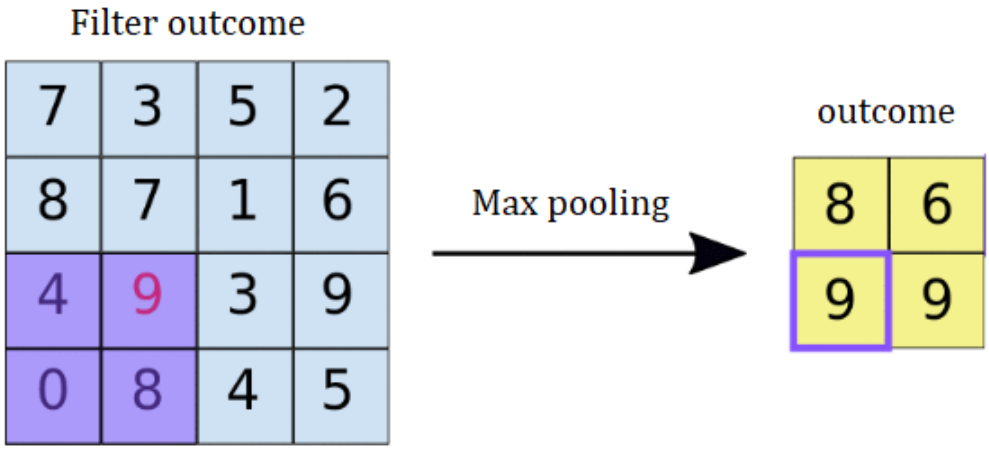

Figure 8 - Max pooling (Google ML 2020d).

This model follows the fully connected layer neural network model, which means that all nodes in one layer are connected to all nodes in the next layer. Where there may also be one or more layers connected to each other.

Finally, with the attributes extracted during the convolutions they will go to a last SoftMax layer where the classification will be done, which was also explained in the previous point. In Figure 9, the diagram of a CNN is shown from the beginning until the result is obtained.

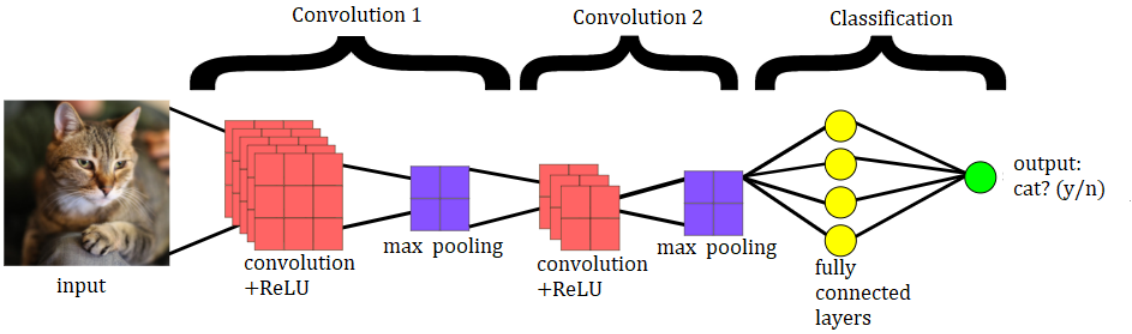

Figure 9 - Convolutional neural network (Google ML 2020d).

3.6 Datasets

Artificial Intelligence is trained using prepared data that we have gathered from a source such as sensor, a camera, files, etc. This set of data needs to suit for each concrete AI model. Thus, we need to prepare that dataset in order to suit it. Furthermore, this set usually has to be divided in three subsets: training dataset, validation dataset and test dataset with the following percentages of the total set 60%, 20%, 20%.

There are public datasets for anyone's use in case of cannot gather your own dataset. One of them is ImageNet, is one of the best datasets for ML, it can be used in computer vision research field (ImageNet 2016), in this case part of the dataset has been gathered.

It's important keep in mind that gather useful dataset takes a lot of time because it's needed thousands of rows to teach AI correctly. There are different types of sets such as: Training data and Validation data. It is also recommendable to take into account that it would be a good idea to have some extra data in order to make some test predictions once we have the model.

Once we have the data to feed the supervised model, we must indicate in some way what appears in the image. This is done through labelling data that will be explained in the following chapters along with the different datasets.

3.6.1 Labelling images

Labelling data is a data that has been tagged giving it a description about its properties. An example could be what type of clothing there is or what type of animal appears in each of the images, depending on the case we are trying to solve.

The task of labelling for AI training usually requires a human to do it, so this task is repetitive but it is important and required because if the labelling is not done correctly, the AI model could learn the wrong way and the result will be predictions with less accuracy.

There are different ways to label data, depending on how you want to train the model, in classification tasks where the object to be classified is clearly seen, something like I show in the Figure 10, the object should be in the center of the image, with a background that does not get confused with the object.

Figure 10 - One label example

For other tasks it may be interesting to be able to label more than one object in an image or to want to tell the model where the object is in the image to label it. For this there are some useful tools like **LabelImg,** which allows you to set a rectangular area where the object is and save the label together with its coordinates.

This labelling is often used for object detection problems where what you need to know is how many objects of a class there are and where they are located, which is more complex than just classifying. An example of this labelling could be similar to the one shown in Figure 11.

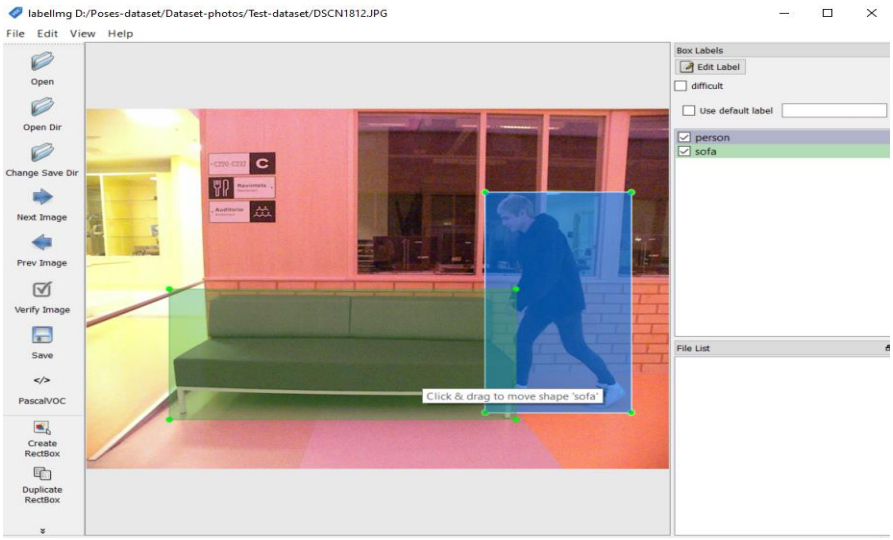

Figure 11 - Labelling with labelImg

3.6.2 Training data

Training data is the data belonging to a training set which is used for learning and fits the parameters of the classifier (Ripley 2005, 354). These parameters are variables that the model trains for itself, for example the weights, gradually learning through successive training iterations.

3.6.3 Validation data

The validation data is the data belonging to the validation dataset that is used to adjust the parameters of a classifier. These parameters are called hyperparameters and they are adjusted during the successive training repetitions.

One example could be deciding how many hidden layers to use between the input layer and the output layer or the number of hidden units in a neural network (Ripley 2005, 354).

3.6.4 Test data

Test data is the data belonging to the test set. It is the last subset used in the process and its main objective is to assess the performance of a trained model (Ripley 2005, 354).

This data set must not have been seen previously by the model since it will be used as a test to check if it is really predicting correctly.

3.6.5 K-Fold cross validation

When the amount of data is relatively small, a dilemma arises, which is how we should split the dataset to be able to train the model without using the same data in training and in validation since this would be a mistake. On the other hand, if we divide them in the way proposed above there will be a very small amount of data in one of the sets, which may cause the model to be insufficiently trained or the validation to be unreliable.

One of the solutions in these cases is to use a technique called cross validation. This technique, as can be seen in diagram 12 below, is based on dividing the set into K-subsets and using K-1 subsets for training while the remaining subset is left for validation. This step will be iterated K times until all layers have been used for training and validation. Once this is done, the model is trained one last time with all the data (Amazon ML 2020b).

The necessary measures such as the error or the accuracy rate of the model during the validation will be calculated K times to finally calculate the average of the set and this will be the measure for the final model.

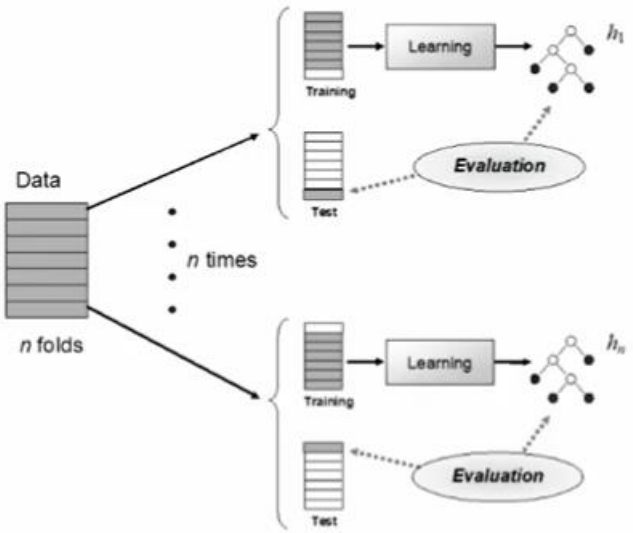

Figure 12 - K-fold cross validation (UPV 2019).

4 TECHNOLOGIES

"Design is not just what it looks like and feels like.

Design is how it works." – Steve Jobs

In this chapter, I will talk about **Java**, **SceneBuider** and a useful library for programming a multiplatform graphics application. On the other hand, I will talk about one of the most used relational database management systems, **MySQL,** which is useful to incorporate an SQL database into a program.

4.1 Java and JavaFX

Java is a programming language and computing platform first released by Sun Microsystems in 1995 (Oracle 2020), it is object oriented, multiplatform, and designed to have as few implementation dependencies as possible. As of 2019, Java was one of the most popular programming languages.

Some examples of leading tech companies using Java can be Twitter, Uber or Netflix (Leiva 2016).

Java is made up of these parts and as it can see in the Figure 13 each part belongs to another of them:

- **JDK**: The JDK is a development environment for building applications, applets and components using the java programming language (Oracle 2020a).

  It includes the Java Runtime Environment, an interpreter/loader, a compiler, an archiver (jar), a documentation generator.

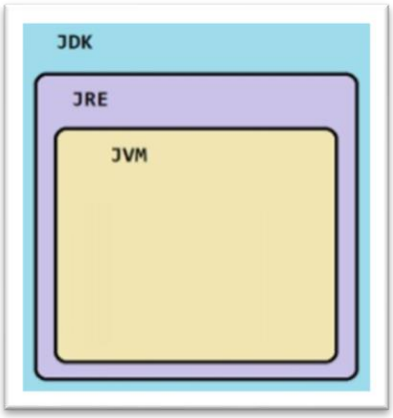

Figure 13 - JDK includes

- **JRE**: The Java Runtime Environment, The JRE consists of the Java Virtual Machine (JVM), Java platform core classes, and supporting Java platform libraries (Oracle 2020b).

Having seen what Java is, next is to explain what JavaFX is, because this library has been used to create the UI of the labeler tool.

JavaFX is a set of graphics and media packages that enables developers to design, create, test, debug and deploy rich client application that operate consistently across diverse platforms (Oracle 2020c). The way to personalize the UI is based on Cascading Style Sheets (CSS) that separate the appearance and style from implementation.

This allows developers to focus on coding, while the graphic designer can work on the user interface.

Creating the UI is easier with a tool that helps you do it. The IDE **Scene Builder** is a visual layout tool for JavaFX application as you can see in the Figure 14 that it lets users quickly design UI without coding, it is a drag and drop components environment in addition to modify their properties (Scene Builder 2020).

On the other hand, the file with the CSS is called FXML file, this is automatically generated by Scene Builder once the UI is saved, only that is needed at the end is link the Java application with the corresponding FXML file. This file can be modified directly by coding but is not recommended if it is designing the UI from the IDE.

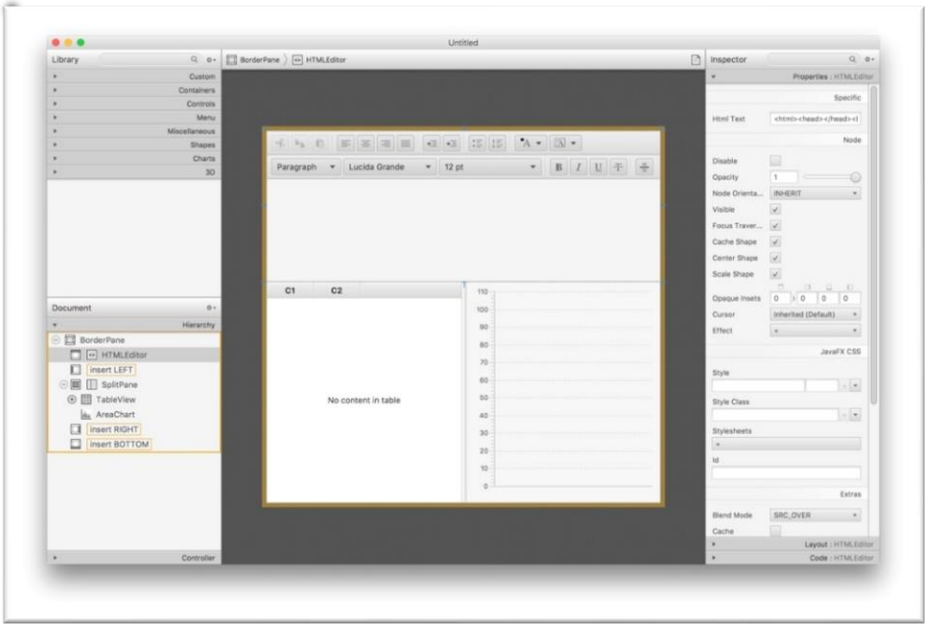

Figure 14 - Scene Builder IDE

The following to explaing is CSS library for JavaFX, called **JFoenix**.

JFoenix is an open source java library, that implements Google Material Design using java components. To use it you only need to download and import it like any other library in Java, this is free available from GitHub where it can find the sources files and compiled jar ready to be imported (Jfoenix 2017).

4.2 SQL and MySQL

SQL means **Structured Query Language**, it is the standard programming language to interact with DB and allows us to access and manipulate DB doing SQL sentences such as executes, inserts, deletes and updates which are the SQL basic operations.

Relational DB query operations are based on relational algebra to work with datasets represented in tables with many rows and columns. It consists of a set of operators that act on the relationships between tables in the database.

Some of the most common **operators** are: (Segura 2018, 16-17)**:**

- **Union**: Given two similar relationships the joining of them results in a table where all the rows appearing in one or both are.
- **Intersection**: Given two similar relationships the intersection of them results in a table in which all the rows that appear in both are in.
- **Difference**: Given two similar relationships the difference or subtraction of them results in a table in which all the rows that appear in the first one is in and not in the second one.
- **Cartesian product**: Given two relationships that do not have equal column names, the Cartesian product results in a relationship with as many columns as the tables have and with all the rows that can be built with one row from the first and one from the second.
- **Select**: Show the rows in a table that meet a condition.
- **Concatenate**: Combines the information contained in two tables by linking the rows that have the same value in some attribute.

Relational DBs are based on relationships between entities or tables. Relationships are associations between tables that are created using join statements to retrieve data. The diagram of a relationship between two tables could be as shown in Figure 15.

The following relationship would be read so that a professor gives one or more courses and a course is given by a single professor.

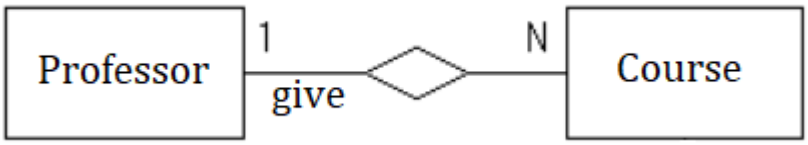

Figure 15 - Example of a relationship between two tables

The objectives of a database are basically to store the information that is required and to consult it in an efficient way. For this, there are database management systems that simplify both data queries and the creation of DBs.

There are many implementations of this standard such as MySQL, Microsoft, SQL Server, SQLite or Oracle. I decided to use MySQL for a couple of reasons, first, it is a well knowing relational database management system based on SQL as I said before and is free and open source (MySQL 2020). Secondly, it is compatible with Java using a MySQL connector to connect both. Moreover, as both technologies are widely known there are reliable documentation to know how to use it together.

To start to use MySQL is required install MySQL Server and MySQL Client (MySQL 2020).

5 CASE: LABELLING TOOL FOR AI

“When building a dataset, it’s important to know

what features to look for, so that the model

can get the most benefit out of it. “(Agic 2018)

In this section, I will explain what technologies and what versions and libraries have been used during the project. Java has been used for coding the labelled tool and SQL for creating the database which stores the dataset.

The objective of this dataset was to make larger the poses database that LAB university of Applied Sciences has, to be used as training data for a machine learning model that would predict poses matching the tagged pose. The pictures were to be used in a fall detection algorithm in order to improve the model that detects if a human has fallen down or not.

After creating a large enough dataset for different poses, about two thousand pictures in this case, we can use this data to train an AI model to accurately predict the pose of a human in new pictures.

The pictures were stored in a relational database created with MySQL and as I said in previous sections the tag task is repetitive, so I created a tool in order to make the task more bearable.

Also, the dataset can be used as training data for other machine learning models, such as pose analysis straight from the picture or detecting where a human being is in the picture.

5.1 Photos taken

An important issue to consider is that the images should be taken from different angles and rotation. This is because the entries in the photo cases could be taken horizontally and vertically and then neural network should be learned as well, similarly with the different angles, when more points of view of the same element get to know the neural network they are more likely to get correct predictions.

The requirements for the pictures were:

- At least over a thousand rows of data with their tags correctly added.
- Images in a normal image format such as jpg or png.
- The quality of the picture should be good enough so that you can make out its content.
- Exactly one human being is presented in the picture with at least some of the body parts visible.
- Types of poses required: standing, sitting, falling, and lying.

As the images had to be tagged by myself, the evaluation of what kind of pose appears in each image has been my responsibility. To take the pictures I was helped by some coworkers I worked with for a few days to make the whole dataset that was needed.

Number of pictures in each pose:

- Standing: 488
- Sitting: 816
- Falling: 400
- Lying: 512

The images shown in Figure 16 are a few examples of the photos captured in each pose to show what type of photo was needed:

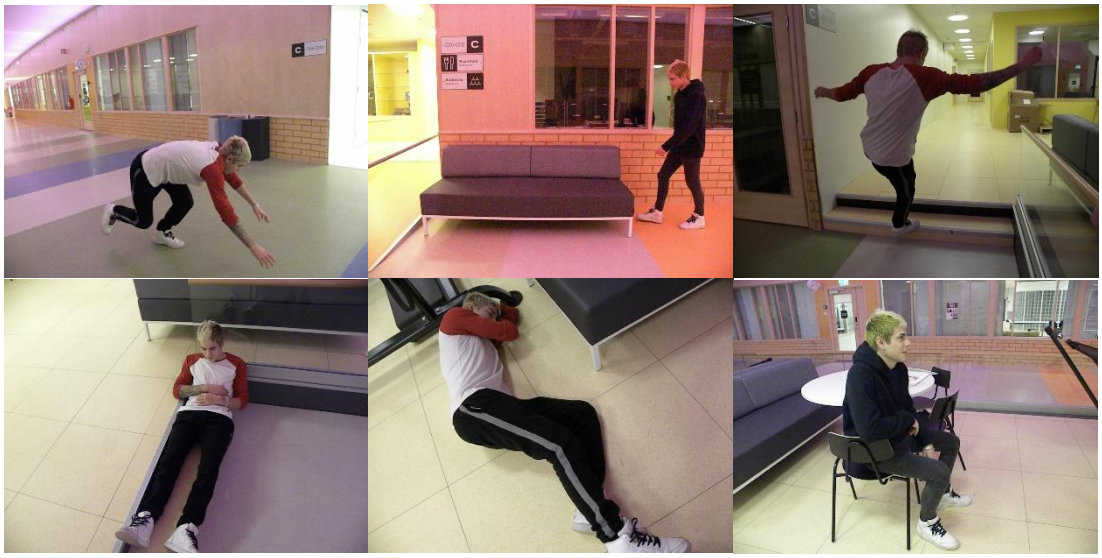

Figure 16 – Examples of the pictures taken

5.2 Database

The dataset was stored in a SQL-based database with MySQL server version 8.0. This was needed because this collection was going to be merged with the main SQL database. As can be seen in Figure 17, the database only consists of one table, so the design of the database has no relevance since it has no relationships.

The fields of the database are the following:

- ID: As identifier of each picture. It must be unique.
- FileName: I decided to use original picture name.
- Pose: The pose you have determined to appear in the picture, in other words, the **label**. It must be string format and lowercase.
- Image: The image in a BLOB format.

Using the BLOB format, the image is stored entirely in the database. This way, the image is uploaded to the database and it is not necessary to keep the image file separately.

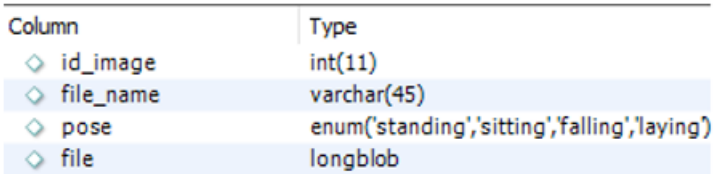

Figure 17 - Rows of poses table

The last step is connecting the labeller tool with the DB. For this, you have to download the MySQL Connector, which is available at the official MySQL website. The connector version used has been 8.0 because it is highly recommended for use with MySQL server 8.0 (Oracle 2020d). MySQL provides connectivity for client applications developed in the Java programming language with MySQL Connector/J, a driver that implements the Java Database Connectivity (JDBC) API (Oracle 2020d).

5.3 Labeller tool

The version used to code the labeller tool has been JDK 8.2. This was decided because to save time doing the UI, I found a JavaFX material design library, which has less bugs in the version compatible with Java 8.x.x.

The SceneBuilder version used has been the latest version available for Java 8, the version 8.5.0. It was released on Jun 5, 2018 though the latest version is 11.0. This decision was taken because version 8.5.0 has less bugs than the latest version and it some compatibility problems appears with the CSS library used later.

The tool consists of a simple UI with two tabs, one with the upload DB functions and another with read and modify DB functions. Figure 18 shows the tab to upload the data.

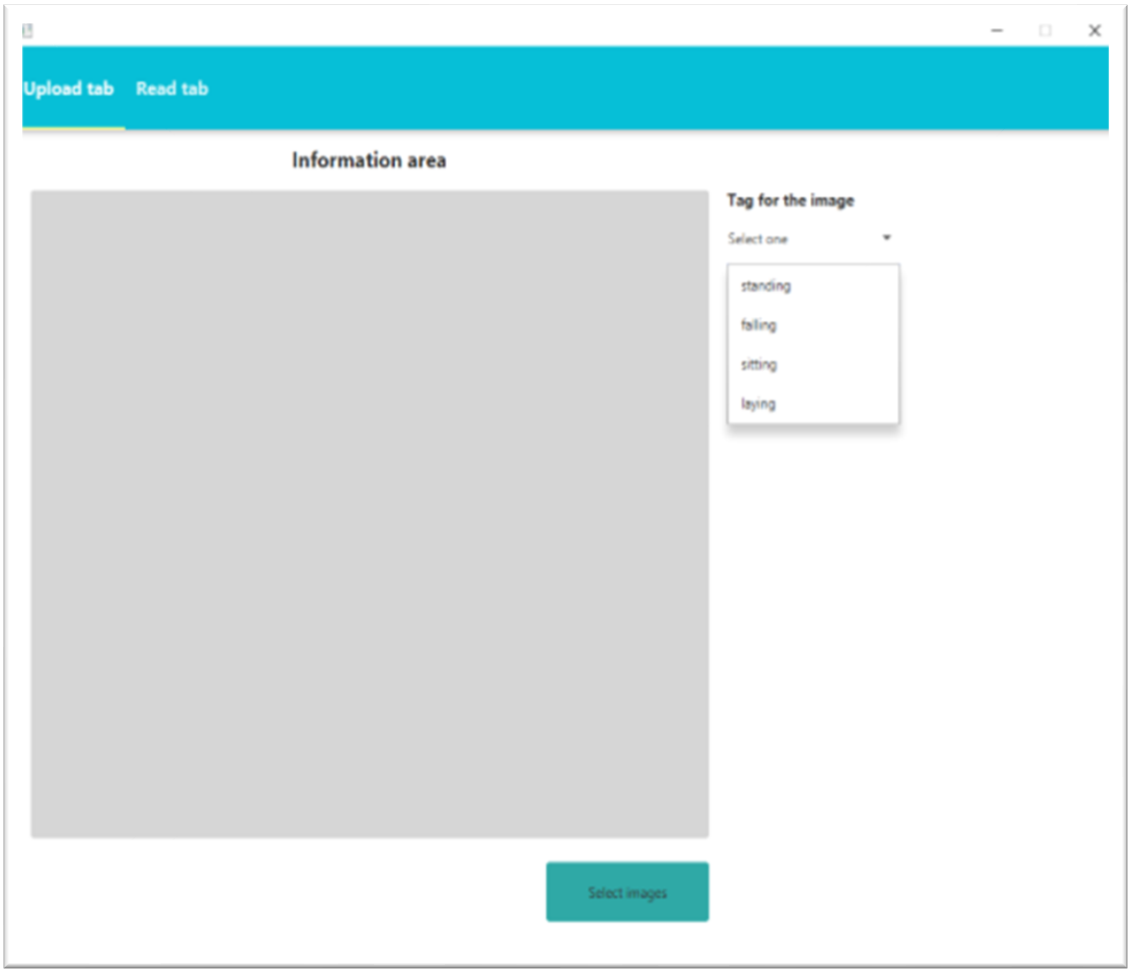

Figure 18 - Upload tab

The idea of this tool is to allow all images belonging to a label to be uploaded at once. To make this possible, I first needed to classify the pictures into the folders, so it is faster than uploading them one by one to the database.

The id field is added automatically using an auto increment option. The name of the file is getting from the original name of the image whilst pose is getting from the selected option in pose combo box and file is the file itself. Figure 19 shows it can see a snippet code about the function to upload images to the DB.

```
@FXML
/**
 * Function to execute the 'Insert' SQL statement. Insert the image file
 * into the DB.
 * @param file it's a File type parameter.
 */
private void insert_image(File file) throws SQLException, FileNotFoundException
{
    String stmt = "INSERT INTO image (file_name, pose, file) VALUES (?,?,?);";
    String tag = cb_uptag.getSelectionModel().getSelectedItem().toString();
    PreparedStatement pstmt;
    FileInputStream input = new FileInputStream(file);
    //DB statements
    pstmt = conn.prepareStatement(stmt);
    pstmt.setString(1, file.getName());
    pstmt.setString(2, tag);
    pstmt.setBinaryStream(3, input);
    pstmt.execute();
}
```

Figure 19 - Upload function

On the other hand, it is possible to display the pictures from the DB with their labels so that it makes it easier to check if the label added is correct. In case the label was wrong, the modify function can be used to get the correct one. This tab is shown in the Figure 20.

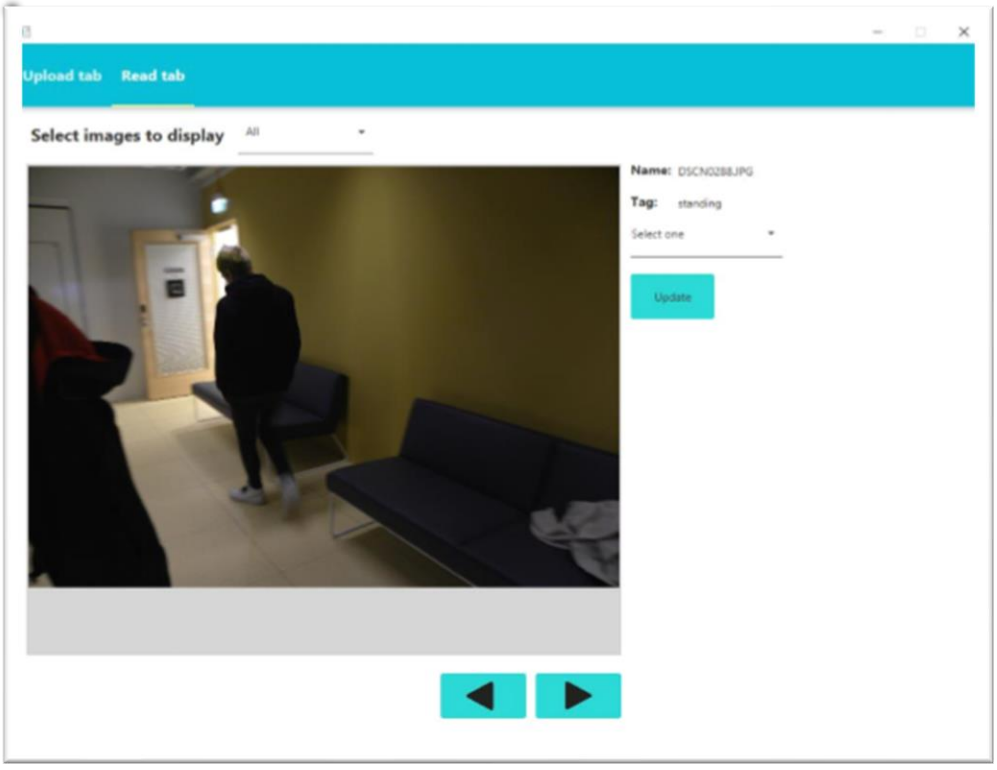

Figure 20 - Read tab

5.4 Case conclusion

The gathering of photos did not have any complexity although it took a significant amount of time since they had to be with specific requirements and in different ways. Once the photos were taken, I created the database along with the application to upload the labelled photos to the database.

The database did not turn out to be a problem since the only requirement was that it was an SQL database and it was only made up of one table. The only doubt that appeared was how to store the images in the database since a couple of options I had were to upload a link to the location of the photos or to upload the photo directly to the database. To avoid future problems with the links I decided to store them in the second way although the database was larger.

Finally, most of the time was spent on the tool and the research part. Although the application was only for personal use to facilitate the labelling, I spent some time on its design and error handling.

Once all tasks were made, I just had to export the database with all the images labelled for delivery.

Finally, some improvements have been left pending for the labelling application to give it a more generic use so anyone can use it with any SQL database since right now it is implemented in a way that it was only useful for the needs I had in the project.

6 CONCLUSION

During the elaboration of this project I have only had minor problems related to incompatibilities between versions, which were solved easily. On the other hand, regarding the theme of the project, it was completely new to me and I realized the task of collecting and labelling data in order to train an ML model requires a significant amount of time. Finally, the scope of this project has been limited to the collection and labelling of the collected data. Leaving aside the implementation of a program that makes use of them. Although it was not possible to implement a program, some future project might be able to do that.

On the other hand, during the research I have realized that really the most complicated part is not programming a neural network and training it since there are many tools and libraries that carry out most of this work, but that the most difficult part is to know how to apply a correct methodology in the treatment of the data so that this way the conclusions that are obtained are not mistaken.

LIST OF REFERENCES

Agic N. Bpugoldings. Blog. 24 December 2018 [Accessed 10 November 2019]. Available at: https://www.bpuholdings.com/the-importance-of-having-a-good-dataset/.

Amazon ML a. Documentation. 2020 [Accessed 10 April 2020]. Types of ML models. Available at: https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/types-of-ml-models.html.

Amazon ML b. Documentation. 2020 [Accessed 10 April 2020]. Cross validation. Available at: https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/cross-validation.html.

Aprendemachinelearning. Blog. 11 September 2017 [Accessed 20 April 2020]. 7 pasos del Machine Learning para construir tu máquina. Available at: https://www.aprendemachinelearning.com/7-pasos-machine-learning-construir-maquina/.

Bejerano, P.G. Blogthinkbig. Blog. 8 February 2017 [Accessed 15 November 2019]. Diferencias entre machine learning y deep learning. Available at: https://blogthinkbig.com/diferencias-entre-machine-learning-y-deep-learning.

Blanco E. Blogthinkbig. Blog. 9 April 2019 [Accessed 17 March 2020]. Que es el overfitting y como evitarlo. Available at: https://empresas.blogthinkbig.com/que-es-overfitting-y-como-evitarlo-html-2/.

Codificandobits. Video. 30 March 2019 [Accessed 21 April 2020]. Las redes convolucionales #2: la convolución. Available at: https://www.youtube.com/watch?v=ySbmdeqR0-4.

Dot CSV. Video. 10 March 2019 [Accessed 17 March 2020]. Cómo identificar el OVERFITTING en tu RED NEURAL - Parte 2. Available at: https://www.youtube.com/watch?v=ZmLKqZYlYUI.

ExpericIA Oracle. Medium. Blog. 14 September 2018 [Accessed 15 November 2019]. Diferencias entre la inteligencia artificial y el machine learning. Available at: https://medium.com/@experiencIA18/diferencias-entre-la-inteligencia-artificial-y-el-machine-learning-f0448c503cd4

Foote K. D. Dataversity. Blog. 26 March 2019 [Accessed 2 March 2020]. A Brief History of Machine Learning. Available at: https://www.dataversity.net/a-brief-history-of-machine-learning/#

Google ML d. Google. Course. 2020 [Accessed 1 april 2020]. Convolutional nerworks. Available at: https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks?hl=es.

GoogleML a. Google. Glossary. 2020 [Accessed 7 January 2020]. Machine Learning glossary. Available at: https://developers.google.com/machine-learning/glossary#unsupervised_machine_learning.

GoogleML c. Google. Course. 10 February 2020 [Accessed 7 January 2020]. Introduction to neural networks. Available at: https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/video-lecture.

ImageNet. Website. 2016 [Accessed 20 November 2019]. Available at: http://www.image-net.org/download-faq

Jfoenix. Website. 2017 [Accessed 27 November 2019]. Jfoenix JavaFX Material Design Library. Available at: http://www.jfoenix.com/.

Leiva A. GenBeta. Blog. 30 May 2016 [Accessed 27 November 2019]. ¿Por qué las empresas vuelven a Java?. Available at: https://www.genbeta.com/desarrollo/por-que-empresas-que-empiezan-con-lenguajes-modernos-se-vuelven-a-java.

Mauri F. Wildentrepreneur. Blog. 8 February 2020 [Accessed 15 November 2019]. Lo que usted puede esperar de la IA. Available at: https://wildentrepreneur.org/lo-que-usted-puede-esperar-de-la-inteligencia-artificial-para-2020/.

MySQL. Oracle. Documentation. 2020 [Accessed 29 November 2019]. MySQL Getting Started. Documentation. Oracle. Available at: https://dev.mysql.com/doc/mysql-getting-started/en/#mysql-getting-started-installing.

María José Segura. Universidad politécnica de Valencia. Lecture. 2018 [Accessed 3 April 2020]. Bases de datos y sistemas de información.

NVIDIA. 18 May 2016 [Accessed 15 November 2019]. Adobe Adds Shazam for Fonts Tool to Photoshop. Available at: https://news.developer.nvidia.com/adobe-adds-shazam-for-fonts-tool-to-photoshop/

Oracle a. Documentation. 2020 [Accessed 26 November 2019]. Java Development Kit 8. Available at: https://www.oracle.com/technetwork/java/javase/jdk-8-readme-2095712.html

Oracle b. Documentation. 2020 [Accessed 26 November 2019]. What is java?. Available at: https://java.com/en/download/faq/whatis_java.xml

Oracle c. Documentation. 2020 [Accessed 26 November 2019]. JavaFX Overview. Available at: https://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm.

Oracle d. Documentation. 2020 [Accessed 26 November 2019]. Java JDBC API. Available at: https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/index.html.

Oracle e. Documentation. 2020 [Accessed 26 November 2019]. Java programming environment. Available at: https://docs.oracle.com/cd/E19455-01/806-3461/6jck06gqd/index.html.

OpenAI. Blog. 30 July 2018 [Accessed 14 April 2020]. Learning Dexterity. Available at: https://openai.com/blog/learning-dexterity/

Oscar Vega. Lecture. 2018. Introducción a la inteligencia artificial. Universidad politécnica de Valencia.

PRWEB. 13 July 2015 [Accessed 15 November 2019]. Insilico Medicine to utilize deep learning for drug repurposing and discovery in cancer and age-related diseases. Available at: http://www.prweb.com/releases/2015/06/prweb12797010.htm.

Rich & Knight. 1991. Artificial intelligence. McGraw Hill. P.5 [referenced 5 December 2019]

Ripley, B. 2005. Pattern Recognition and Neural Networks. Cambridge University Press. P. 354 [referenced 5 December 2019]

Robert Moni. Medium. Blog. 18 February 2019 [Accessed 14 April 2020].. Reinforcement Learning algorithms an intuitive overview. Available at: https://medium.com/@SmartLabAI/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc

Salman, A. Research. May 2009 [Accessed 14 April 2020]. The main structure of a neuron. Available at: https://www.researchgate.net/Figure/The-main-structur-of-a-neuron-consists-of-soma-dendrites-and-axon_fig3_220856618.

Santos P. R. Blogthinkbig. Blog. 2017 November 2016 [Accessed 12 March 2020]. Tipos de aprendizaje en Machine Learning: supervisado y no supervisado. Available at: https://empresas.blogthinkbig.com/que-algoritmo-elegir-en-ml-aprendizaje/

Scene Builder. Oracle. Website. 2020 [Accessed 28 November 2019]. JavaFX Scene Builder. Available at: https://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html

Serrano A. G. 2012. Inteligencia artificial: Fundamentos, prácticas y aplicaciones. RC Libros. P. 2 [referenced 10 November 2019]

Serrano A. G. 2012. Inteligencia artificial: Fundamentos, prácticas y aplicaciones. RC Libros. P. 210-213 [referenced 20 March 2020]

Serrano A. G. 2012. Inteligencia artificial: Fundamentos, prácticas y aplicaciones. RC Libros. P. 208-211 [referenced 20 March 2020]

Techworld. Techworld. Blog. 8 November 2019 [Accessed 15 November 2019]. Tech giants investing in artificial intelligence. Available at: https://www.techworld.com/picture-gallery/data/tech-giants-investing-in-artificial-intelligence-3629737/

UPV Universidad politecnica de valencia. Video. 16 January 2019 [23 March 2020]. Validación cruzada. Available at: https://www.youtube.com/watch?v=jzcD1iT3-iQ&list=PL6kQim6ljTJsmPM_v_b2p8p-GPQhtoekI&index=27.

Zaforas M. Paradigmadigital. Blog. 2018 [Accessed 23 February 2020]. Inteligencia artificial como servicio: Reconocimiento de imágenes. Available at: https://www.paradigmadigital.com/techbiz/inteligencia-artificial-servicio-reconocimiento-imagenes/.

APPENDICES

APPENDIX 1

In Appendix 1 I have added the main part of the code used for the implementation of the labelling application.

```
package com.readuploaddb;
import com.jfoenix.controls.JFXButton;import com.jfoenix.controls.JFXComboBox;
import com.jfoenix.controls.JFXTextArea;import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;import java.io.File;import java.io.FileInputStream;
import java.io.FileNotFoundException;import java.io.IOException;import java.net.URL;
import java.util.List;import java.util.ResourceBundle;import javafx.event.ActionEvent;
import javafx.fxml.FXML;import javafx.fxml.Initializable;import javafx.stage.FileChooser;
import javafx.stage.Stage;import java.sql.*;import java.util.logging.Level;
import java.util.logging.Logger;import javafx.collections.FXCollections;
import javafx.collections.ObservableList;import javafx.embed.swing.SwingFXUtils;
import javafx.scene.control.Alert;import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.Label;import javafx.scene.image.Image;
import javafx.scene.image.ImageView;import javax.imageio.ImageIO;
/**
 * @author Rafael Esparza Tortosa
 */
public class FXMLDocumentController implements Initializable{
    //Visual components
    @FXML
    final FileChooser fileChooser = new FileChooser();
    @FXML
    private Stage stage;
    @FXML
    private JFXTextArea ta_report;
    @FXML
    private ImageView img_view;
    @FXML
    private JFXComboBox cb_uptag, cb_uptag2, cb_filter;
    @FXML
    private JFXButton b_uprun, b_next, b_prev, b_update;
    @FXML
    private Label lbl_imgname, lbl_imgtag;
    //Variables
    private List<File> list = null;
```

```
private Connection conn = null;
private PreparedStatement stmt = null;
private String sURL = "jdbc:mysql://localhost:3306/poses";
private Boolean clickNext = false;
private ResultSet rs = null;
@FXML
/**
 * Takes the images selected with the file chooser,
 * save them in the List 'list' and write their absolute path in the
 * text area 'ta' of the window update.
 * Function executed when the images are selected.
 */
private void up_file(ActionEvent event){
    ta_report.setText("");
    list = fileChooser.showOpenMultipleDialog(stage);
    if (!list.isEmpty()){
        b_uprun.setDisable(false);
    }
    if (list != null){
        for (File file : list){
            ta_report.appendText(file.getAbsolutePath() + "\n\n");
        }
    }
}
@FXML
/**
 * Upload the images previous selected into the DB. Executed with the button
 * 'Run' in the upload window.
 */
private void run_sql_statement(ActionEvent event) throws FileNotFoundException{
    Alert alert = new Alert(AlertType.INFORMATION);
    alert.setTitle("Information Dialog");
    alert.setHeaderText(null);
    try{
        conn = open_conn_db();

        for (File file : list){
            insert_image(file);
        }
        conn.close();
```

```
            alert.setContentText("Upload done!");
        } catch (SQLException ex){
            alert.setContentText("Error has ocurred!" + "\n" + "Check DB log for more information");
            Logger.getLogger(FXMLDocumentController.class.getName()).log(Level.SEVERE, null, ex);
        } finally{
            alert.showAndWait();
        }
    }
    @FXML
    /**
     * Function to execute the 'Insert' SQL statement. Insert the image file
     * into the DB.
     * @param file it's a File type parameter.
     */
    private void insert_image(File file) throws SQLException, FileNotFoundException{
        String stmt = "INSERT INTO image (file_name, pose, file) VALUES (?,?,?);";
        String tag = cb_uptag.getSelectionModel().getSelectedItem().toString();
        PreparedStatement pstmt;
        FileInputStream input = new FileInputStream(file);
        //DB statements
        pstmt = conn.prepareStatement(stmt);
        pstmt.setString(1, file.getName());
        pstmt.setString(2, tag);
        pstmt.setBinaryStream(3, input);
        pstmt.execute();
    }
    @FXML
    /**
     * Function to read the images inserted into the DB, save them in a
     * ResultSet object named 'rs' and display in the
     * ImageView of the 'Read Window'.
     * This is executed when it's selected one value in the
     * 'Select image to...' comboBox.
     * NOTE! This function is not intended to read huge amount of images.
    */
    private void read_images_table() throws IOException, NoSuchFieldException{
        //Alert info dialog
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setTitle("Information Dialog");
        alert.setHeaderText(null);
```

```
    String stmt = null;
    if (cb_filter.getValue().toString().equals("All")){
        stmt = "SELECT * FROM image";
    } else{
        stmt = "SELECT * FROM image WHERE pose IS NULL";
    }
    try{
        conn = open_conn_db();
        Statement ppstmt = conn.createStatement(0, ResultSet.CONCUR_UPDATABLE);
        ppstmt.setFetchSize(1);
        rs = ppstmt.executeQuery(stmt);
        if (rs.first()){
            display_image(null);
            //Enable action buttons
            b_next.setDisable(false);
            b_prev.setDisable(false);
            b_update.setDisable(false);
            //Information Alert window
            alert.setContentText("Images read correctly!");
        }else{
            alert.setContentText("There are no images with that filter.");
        }
        //conn.close();
    } catch (SQLException ex){
        alert.setContentText("Error has ocurred reading the images!");
        Logger.getLogger(FXMLDocumentController.class.getName()).log(Level.SEVERE, null, ex);
    } finally{
        alert.showAndWait();
    }
}
@FXML
/**
 * Take one image from the ResultSet and displays in the ImageView
 * of the 'Read Window'.
 * If some of the arrow buttons are pressed this method will be executed
 * to display the next or the previous image.
 */
private void display_image(ActionEvent event) throws IOException, SQLException{
    if (event != null){
        JFXButton btn = (JFXButton) event.getSource();
```

```
        if (btn.getId().equals("b_next")){
            if (rs.isLast()); else{
                rs.next();
            }
        } else if (btn.getId().equals("b_prev")){
            if (rs.isFirst()); else{
                rs.previous();
            }
        }
    }
    //img to imgView
    ByteArrayInputStream bis = new ByteArrayInputStream(rs.getBytes(4));
    BufferedImage read = ImageIO.read(bis);
    Image image = SwingFXUtils.toFXImage(read, null);
    img_view.setImage(image);

    //Inform Labels
    lbl_imgname.setText(rs.getString(2));
    lbl_imgtag.setText(rs.getString(3));
}
@FXML
/**
 * Function to update the tag of the image. It can used choosing an
 * option from the tag ComboBox in the 'Read Window'.
 */
private void update_image(){
    //Alert info dialog
    Alert alert = new Alert(AlertType.INFORMATION);
    alert.setTitle("Information Dialog");
    alert.setHeaderText(null);
    try{
        String stmt = "UPDATE image SET pose = \"" + cb_uptag2.getValue().toString()
                + "\" WHERE id_image= " + rs.getInt(1) + ";";
        Statement pstmt = conn.createStatement();
        pstmt.executeUpdate(stmt);
        rs.refreshRow();
        System.out.println(rs.getString(3));
        lbl_imgtag.setText(rs.getString(3));
        //Information Alert window
        alert.setContentText("Image update correctly!");
```

```
        } catch (SQLException ex){
            alert.setContentText("Error has ocurred updating the images!");
            Logger.getLogger(FXMLDocumentController.class.getName()).log(Level.SEVERE, null, ex);
        } finally{
            alert.showAndWait();
        }
    }
    /**
     * Create the connection with the DB.
     * @return Connection
     * @throws SQLException
     */
    private Connection open_conn_db() throws SQLException{
        return DriverManager.getConnection(sURL, "raestor", "bcn34");
    }
    @Override
    public void initialize(URL url, ResourceBundle rb){
        //Text Area
        ta_report.setText("");//ComboBox
        ObservableList<String> type_tags = FXCollections.observableArrayList("standing", "falling",
"sitting", "lying");
        ObservableList<String> filter_opt = FXCollections.observableArrayList("All", "pose NULL");
        cb_uptag.setItems(type_tags);
        cb_uptag2.setItems(type_tags);
        cb_filter.setItems(filter_opt);
        //Buttons
        b_next.setDisable(true);
        b_prev.setDisable(true);
        b_update.setDisable(true);
    }
}
```