Pict2Text 2.0. Translating messages with pictograms into text

Final project
Course 2021–2022

Autors
Gasan Mohamad Nazer
and
Veronika Borislavova Yankova

Directors
Virginia Francisco Gilmartín
and
Susana Bautista Blasco

Bachelor's Degree in Software Engineering
Faculty of Informatics
Complutense University of Madrid

Pict2Text 2.0. Translating messages with pictograms into text

Bachelor's Degree Final Project in Software Engineering

Autors
Gasan Mohamad Nazer
and
Veronika Borislavova Yankova

Directors
Virginia Francisco Gilmartín
and
Susana Bautista Blasco

Convocation: *June 2022*

Bachelor's Degree in Software Engineering
Faculty of Informatics
Complutense University of Madrid

January 10, 2021

Contents

Chapter 1

Introduction

This chapter will display the motivation behind this project and its objectives. In the end, we present the document structure with the different sections within it and a brief description of them.

1.1. Motivation

Communication is a pillar in interpersonal relationships, a fundamental need in our society. However, for some people communication requires a lot of effort. Their differences create an uncrossable barrier and almost impossible human connection using the traditional way of communication. To remove this barrier an alternative approach should be used, for example the use of alternative ways of communication as pictograms. These graphic images, representing an object or a concept, have helped people with special needs to communicate. However, the majority of the population does not understand pictograms.

In order to include pictograms into the communication between people with disabilities and those without, we can use modern technology and create a software to be a mediator between the two sides.

Currently, multiple tools allow transforming natural language into pictograms, one of which is the first version of this project - Pict2Text, but there is no such tool that does the inverse - from an image of pictograms to text. Pict2Text translates pictograms into natural language, but to create the text with pictograms, the pictograms must be selected manually searching for every single one in a search engine by entering the words associated with the pictograms. This is not good enough as people with disabilities who need the pictograms, cannot think about the words that compose the sentence and search for them in the search engine. This problem can be solved, giving those people the option to upload or take a picture of a sentence

constructed with pictograms. By creating a new version of Pict2Text, we aim to reduce the exclusion of those people with special needs from the society and break the communication barrier, providing the aforementioned functionality. Another objective of ours is the improvement of the translation since currently only simple sentences can be translated.

The beneficiaries of this software are people who don't understand pictograms but want to communicate with people who are using them. This project will help people understand each other better, creating a more equal society, independently of the ability to use the natural language in our communication.

1.2. Goals

The main goal of our project is to improve Pict2Text. To do that, we have two objectives - to build an application that can recognize pictograms on uploaded or taken at the moment pictures and improve the translation that Pict2Text provides given a set of pictograms.

We use services-oriented application architecture, constructing web services, and microservices, to increase maintainability and scalability. For the same reasons, we also follow industry standards during the whole process.

Last but not least, we want to consolidate and expand the knowledge acquired during the Software Engineering Degree.

1.3. Document structure

The document is structured as follows. Chapter 2, State of the Art, covers the state of the first version of Pict2Text, the functionalities we will include to increase its usefulness, and the general idea of the machine learning algorithms and models we will be using. Chapter 3, Software development methodology, describes the methodology we have chosen to use, it's characteristics, and things specific to our application of it.

Chapter 2

State of the Art

In this chapter, we have briefly defined Augmentative and Alternative Systems of Communication (section 2.1) and pictograms (section 2.2). Also, in section 2.3, we review Pict2Text 1.0, which is the tool that serves as the basis for our work. In section 2.4. we present the machine learning models we have analyzed to solve our problem and their characteristics.

2.1. Augmentative and Alternative Systems of Communication (AASC)

The Augmentative and Alternative Systems of Communication (AASC)[1,2,3] are communication systems, alternative to the natural language, that don't use spoken or written words but can transmit information. They are used by people, who cannot use natural language, or it is not sufficient for them to express themselves. They are created to increase the communication capabilities of the people who use them.

The AASCs do not arise naturally, but they need previous knowledge. There are two types of AASCs - those who need additional equipment such as objects, pictures, pictograms, etcetera, and those that do not need any equipment, depending on the needs, as they are often personalized to match the needs of its user.

They include different systems of symbols: graphic and gestural. The gestural symbols can vary widely from mimicry to hand signs. The graphic symbols can be used by both people with an intellectual disability or with a motor disability. Examples of graphic symbols are drawings and pictures, as well as pictograms, which will be better explained in the next chapter.

[1] EDITOR PREDICTIVO DE MENSAJES EN PICTOGRAMAS
[2] PICTAR
[3] INTERSAACs

Those systems provide various benefits for their users. They prevent or decrease the isolation of people with disabilities, helping with the improvement of social and communication abilities. Also, AASCs are relatively easy to learn and apply, and adapted for modern technology.

2.2. Pictograms

Pictograms are written signs representing objects from the real world, as shown in Figure 2.1 - a pictogram of an icecream, ideas, actions, etcetera. In general, they represent anything that a person would want to express, without the need for verbal or written language. Pictograms are used in the day to day life at hospitals, malls, airports, etcetera because they are not strictly related to the spoken language. Pictograms are widely used by people with special needs, to help with communication and social integration.

As pictograms are not universal, various systems exist, such as Blissymbolics, CSUP, Minspeak, ARASAAC, and more.

Figure 2.1: Pictogram representing an icecream.

2.2.1. Pictogram systems

2.2.1.1. Blissymbolics

Blissymbolics[4], an example of which you can see in Figure 2.2, created in 1949, is an ideographic language consisting of several hundred basic symbols, each representing a concept. Each symbol is represented by basic forms (circles, triangles) and universal signs (numbers, punctuation signs) and uses colour codification to mark the grammar category. They can be combined to generate new symbols that represent new concepts. Blissymbolics characters

[4]PICTAR

do not correspond to the sounds of any spoken language and have their use in the education of people with communication difficulties.

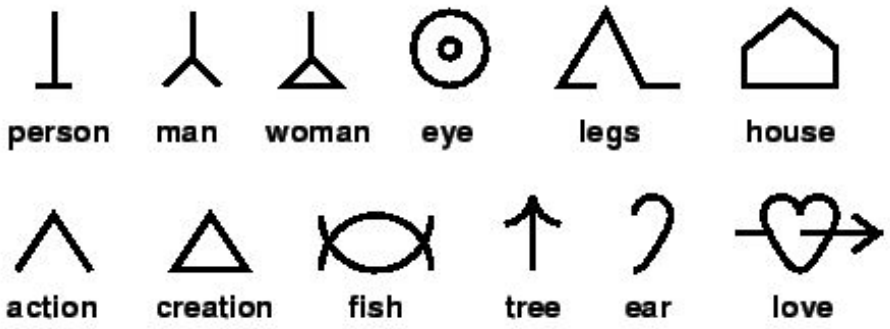

Figure 2.2: Example of Blissymbolics

2.2.1.2. CSUP

The Communication System Using Pictograms (CSP)[5], developed in 1981 by Mayer-Johnson, is one of the systems that use pictograms to support interactive non-verbal communication. This system uses pictograms not only for objects but with events as well. It is designed in a way that it can be used between a person with a disability and a non-disabled person, child and adult, people speaking different languages, and so on.

2.2.1.3. Minspeak

Minspeak (Semantic Compaction Systems, s.f.)[6] is a pictographic system created by Bruce Baker in 1992. Unlike other systems, this system is based on multi-meaning icons whose meaning is determined by the speech therapist and the user. As you in Figure 2.3, two or three icons are combined, determined by rule-driven patterns, to code vocabulary.

2.2.1.4. ARASAAC

The ARASAAC[7,8,9] system is the most used pictogram system in Spain. The ARASAAC project was created in 2007, and it currently consists of more than 30.000 pictograms, including complex pictograms with already constructed phrases, in more than 20 languages. The pictograms are separated

[5]PICTAR
[6]PICTAR
[7]https://arasaac.org/
[8]Traductor de Pictogramas a Texto
[9]EDITOR PREDICTIVO DE MENSAJES EN PICTOGRAMAS

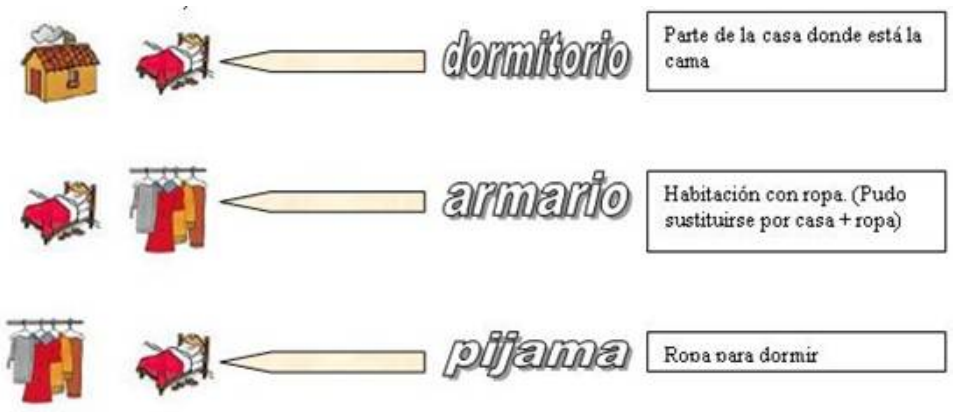

Figure 2.3: Construction of symbologic in Minspeak.

into five groups - coloured pictograms, black and white, photographs, and sign language videos and pictures. Unlike other pictogram systems, ARASAAC makes a difference between singular and plural and gender differentiation, an example in Figure 2.4 where you can see the difference between the pictograms representing male and female teacher.

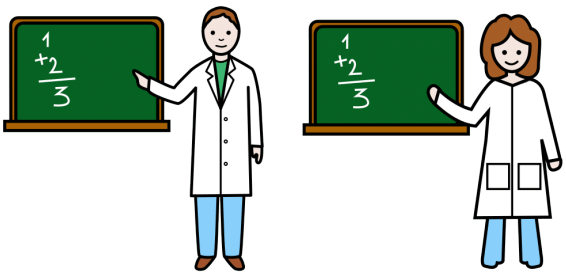

Figure 2.4: Example for gender diferenciation for the word 'teacher'.

As you can see in Figure 2.5, one word can be represented by various pictograms. I the current example we can observe the word 'teacher' having various representations.

Verbs come with a different pictogram for every conjugation, and the tense is determined by pictograms representing yesterday, today, and tomorrow.

ARASAAC is free to use, internationally recognized, and provides a wide variety of pictograms. For those reasons, we decided to use the pictograms we can obtain from their website.

Figure 2.5: Example for different pictograms representing the same word(teacher).

2.2.2. Communication based on pictograms

Communication via pictograms happens with the help of a board or a communication book, example of which you can see in Figure 2.6 - an ACC communication book, where the person points to the pictograms one by one to form a sentence. The complexity of the phrases is limited, usually consisting only of subject, verb, and object. Often, only the most significant words are used, although ARASAAC has pictograms for determinatives and prepositions.

Figure 2.6: Example of ACC communication book.

2.3. Pict2Text 1.0

As described previously, Pict2Text 1.0[10] is the initial state and the base of our project. The first version of this project is a web application that permits the translation of sentences written using pictograms to natural language (Spanish).

[10]Traductor de Pictogramas a Texto

The following images and descriptions present a simple flow of actions a user can do to translate a message written with pictograms into natural language. When entering the website[11] the user can see on the left part, a big panel, the pictogram sentence panel, with a caption 'Pictograms' above it, and a button 'Traducir' below it. On the right part, an input box with the caption 'Nombre del picto', and a button 'Buscar' on the left of it. We can write and search a specific word from the ARASAAC pictogram database and display it into a panel on the right part of the web page. After that, we can include the chosen one into the pictogram sentence panel, from where later the message is translated into natural language.

To search for a specific pictogram, the user should write the world they are looking for in the input box of the right side and click the button 'Buscar' as shown in Figure 2.7, where we can see the search of the word 'Hombre' and the corresponding pictogram.

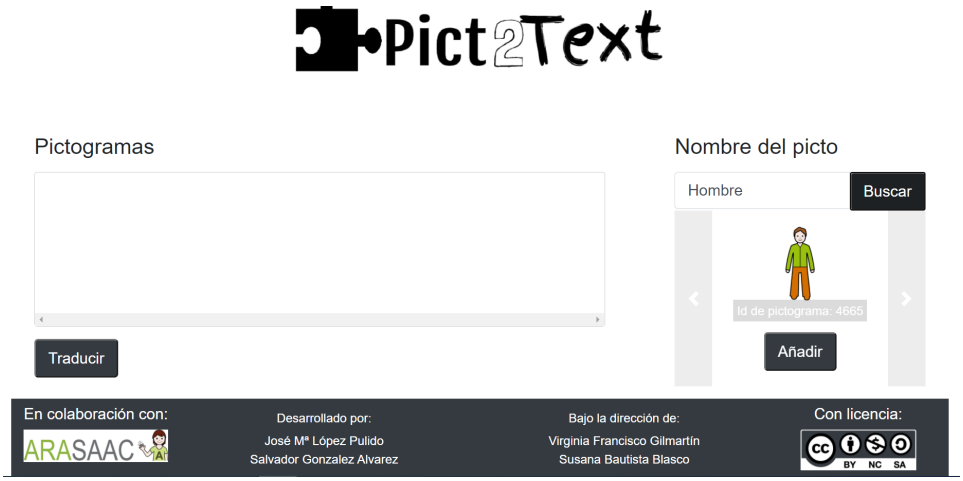

Figure 2.7: Searching the word "Hombre"

After searching the pictogram, the user needs to include it into the left panel with pictograms. This is done by clicking the button "Añadir". As we can observe in Figure 2.8, the pictogram corresponding to the word "Hombre" is included in the pictogram sentence panel.

We can form a sentence by repeating the previous step with other words. Figure 2.9 displays a translation of a sentence written with the pictograms corresponding to the words "Hombre", "Comer", "pizza". As we can see that is translated to "El hombre come una pizza" ("The man is eating a pizza").

[11] https://holstein.fdi.ucm.es/tfg-pict2text

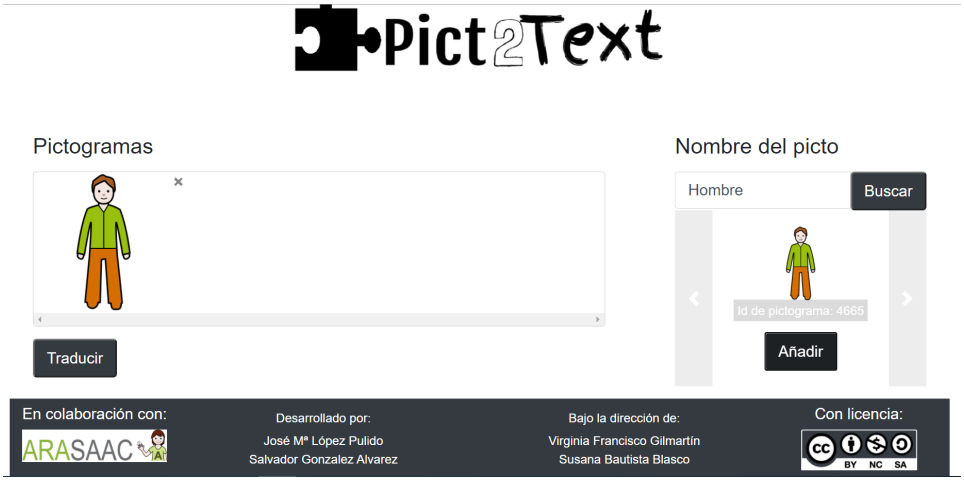

Figure 2.8: Adding the pictogram "Hombre" to the pictogram sentence panel

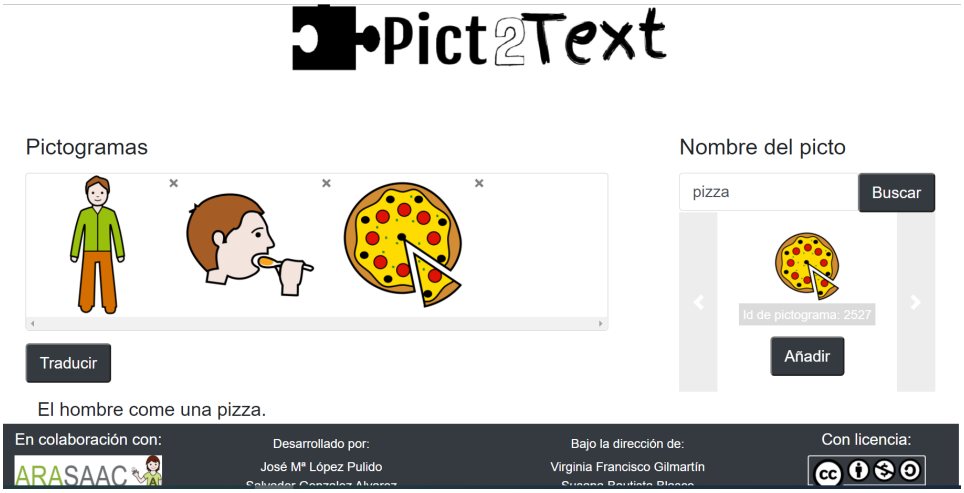

Figure 2.9: Translating the sentence "El hombre come una pizza."

2.3.1. Implementation

For the front-end of the project, Angular[12] was used. As the website itself is a SPA (Single-Page Applications), which needs to respond fast, a framework like Angular fulfills this performance requirement.

As all of the different functionalities from the project were implemented as web services, most of them in Python, the team of 'Pict2Text' version 1 have decided to use the framework Django[13] for integration and intercommunication

[12] Angular
[13] Django

between them.

The core of Pict2Text 1.0 is the API of ARASAAC[14]. It provides the searching mechanism used to match words to pictograms, the graphical images of pictograms, and additional information about them.

A generator of grammatically correct phrases in Spanish was needed. SimpleNLG[15] was used in Pict2Text 1.0, a Java library for natural language generation. This library permits the creation of simple and complex phrases. To do that, it requires sentence structure- subject, verb, adjectives, gender, and number (singular or plural) of every word in the formed sentence. With this information, SimpleNLG can generate grammatically correct sentences.

Spacy[16] is the tool that gives the previous word characteristics. It is a python library with a high accuracy used for advanced natural language processing.

2.3.2. Conclusions

Although the first version of Pict2Text translates the pictograms into natural language, it requires the user to manually select the pictograms they want to use in the construction of their sentence. Writing the words is impossible for people with disabilities who need pictograms to communicate if it was not, they would have used the natural language in the first place.

This problem can be solved, giving those people the option to upload a picture of a sentence constructed with pictograms. The functionality we are building will be able to separate the different pictograms from the original image and later translate the phrase using the implementation in Pict2Text 1.0.

Another issue that exists in this project is the translation of pictograms to natural language. The algorithm used allows the correct translation of simple phrases with one subject and one verb. One of our goals is to improve that performance and provide the opportunity to construct complex sentences with a high level of assertion.

2.4. Machine learning algorithms

2.4.1. Why a conventional algorithm is not going to resolve the problem?

As the problem we have is to detect pictograms from a given image, we cannot assume that a specific pictogram has one, and only one, graphical representation. Although being relatively close for a particular topic, the

[14] ARASAAC API
[15] SimpleNLG
[16] Spacy

images of the different pictograms are not the same. Their color and sometimes format varies.

In ARASAAC, searching for the word ‘man‘ returns several different pictograms. The first two results are shown in the Figure 2.10, and they are an exact match to the searched word. The first is a colored pictogram, and the second a grayscale[17] one.

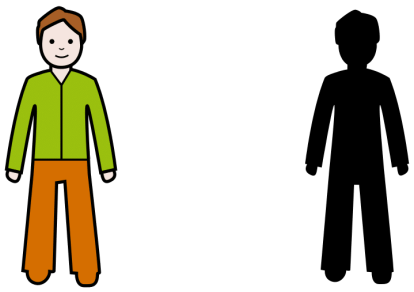

Figure 2.10: Colored and grayscale pictograms representing the word "man"

But the different colors and formats are not the only problem why we cannot compare them pixel by pixel. Both of the pictograms above are of a man. As we can observe, in the first, the man is represented with a mouth, a nose, eyes, and lines separating his shirt from his trousers, and everything is in colour. The second one has the same form, but all filled in black. Despite the similarities, a comparison between the two pictograms pixel by pixel will indicate that the images are not the same. Other pictograms differ in shape, as well as in colour. Considering that, we determined that without a machine learning model, capable of comparing images, we will not be able to solve our problem.

2.4.2. What is a Machine Learning and Machine Learning model?

Machine learning (ML)[18] is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A machine learning model[19] is a file that has been trained to recognize

[17] Grayscale
[18] Machine learning
[19] What is a machine learning model?

certain types of patterns. You train a model over a set of data, providing it an algorithm that it can use to reason over and learn from those data. Once you have trained the model, you can use it to reason over data that it hasn't seen before, and make predictions about those data

2.4.3. Why no other tool or ML model is helpful to resolve our problem?

After analyzing the ARASAAC Spanish pictograms - our dataset, we realized there are several difficulties and problems we have to consider. The main one is that there are above 8000 pictograms and more than 6000 represent different concepts. Having a dataset with as few representatives of the different classes as ours does not apply to most of the ML models. To train a model from scratch a dataset with a lot of representatives of the different classes is needed. In our case with a single or less than 3 applying even the simplest shallow Neural Network, Logistic Regression will overfit to the data and gives a high but unrealistic accuracy. Providing images not included in the original dataset to an overfitted model will show poor accuracy and incorrect results. With a more complex Neural Network model, the situation described above will be the same, even worse, because they need even more data as they find better functions and more distinct features of the given images.

2.4.4. What is a Neural Network?

Artificial neural networks (ANNs)[20], usually called neural networks (NNs), are computing systems vaguely inspired by the biological neural networks that constitute animal brains. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, as presented in Figure 2.11, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.

[20] Artificial neural network

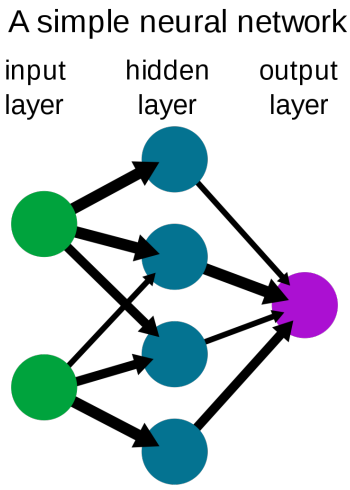

Figure 2.11: A simple neural network architecture.

Supervised learning is learning through pre-labelled inputs, which act as targets. For each training example there will be a set of input values (vectors) and one or more associated designated output values. The goal of this form of training is to reduce the models overall classification error, through correct calculation of the output value of training example by training. Unsupervised learning differs in that the training set does not include any labels. Success is usually determined by whether the network is able to reduce or increase an associated cost function. However, it is important to note that most image-focused pattern-recognition tasks usually depend on classification using supervised learning[21].

2.4.5. Convolutional Neural Network

Convolutional Neural Networks (CNNs)[22] are analogous to traditional ANNs in that they are composed of neurons that self-optimize through learning. Each neuron will still receive input and perform an operation, such as a scalar product followed by a nonlinear function - the basis of countless ANNs. From the input raw image vectors to the final output of the class score, the entire network will still express a single perceptive score function (the weight). The last layer will contain loss functions associated with the classes. The only notable difference between CNNs and traditional ANNs is that CNNs are primarily used in the field of pattern recognition within images. This allows us to encode image-specific features into the architecture, making the network more suited for image-focused tasks - whilst further

[21] An Introduction to Convolutional Neural Networks
[22] An Introduction to Convolutional Neural Networks

reducing the parameters required to set up the model. One of the largest limitations of traditional forms of ANN is that they tend to struggle with the computational complexity required to compute image data. Common machine learning benchmarking datasets such as the MNIST[23] database of handwritten digits are suitable for most forms of ANN, due to its relatively small image dimensionality of just 28 x 28. With this dataset, a single neuron in the first hidden layer will contain 784 weights (28x28x1 where 1 bare in mind that MNIST is normalised to just black and white values), which is manageable for most forms of ANN. If you consider a more substantial coloured image input of 64 x 64, the number of weights on just a single neuron of the first layer increases substantially to 12288(64x64x3- Height, Width, RGB[24] channels). Also, take into account that to deal with this scale of input, the network will also need to be a lot larger than one used to classify colour-normalised MNIST digits, then you will understand the drawbacks of using such models.

2.4.5.1. Convolution and Convolution operation

The convolution operation[25] is one of the fundamental building blocks of a Convolutional Neural Network. Although in mathematics (in particular, functional analysis), convolution is a mathematical operation on two functions (f and g) that produces a third function that expresses how the shape of one is modified by the other, in computer vision, this term has a slightly different meaning (the convolution as described in the use of convolutional neural networks, bellow, is a 'cross-correlation').

In the context of a convolutional neural network, convolution is a linear operation that involves the multiplication of a set of weights with the input, much like in a traditional neural network. Given that the technique was designed for two-dimensional input, the multiplication is performed between an array of input data and a two-dimensional array of weights, called a filter or a kernel. The filter is smaller than the input data, and the type of multiplication is applied between a filter-sized patch of the input, and the filter is a dot product. A dot product is an element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value. Because it results in a single value, the operation is often referred to as the 'scalar product'. Using a filter smaller than the input is intentional as it allows the same filter (set of weights) to be multiplied by the input array multiple times at different points on the input. Specifically, the filter is applied systematically to each overlapping part or filter-sized patch of the input data, left to right, top to bottom. If the filter is designed to detect a specific type of feature in the input, then the

[23] MNIST
[24] RGB
[25] Convolution

application of that filter systematically across the entire input image allows the filter an opportunity to discover that feature anywhere in the image. The output from multiplying the filter with the input array one time is a single value. As the filter is applied multiple times to the input array, the result is a two-dimensional array of output values that represent the filtering of the input. As such, the two-dimensional output array from this operation is called a 'feature map'[26]- Figure 2.12.

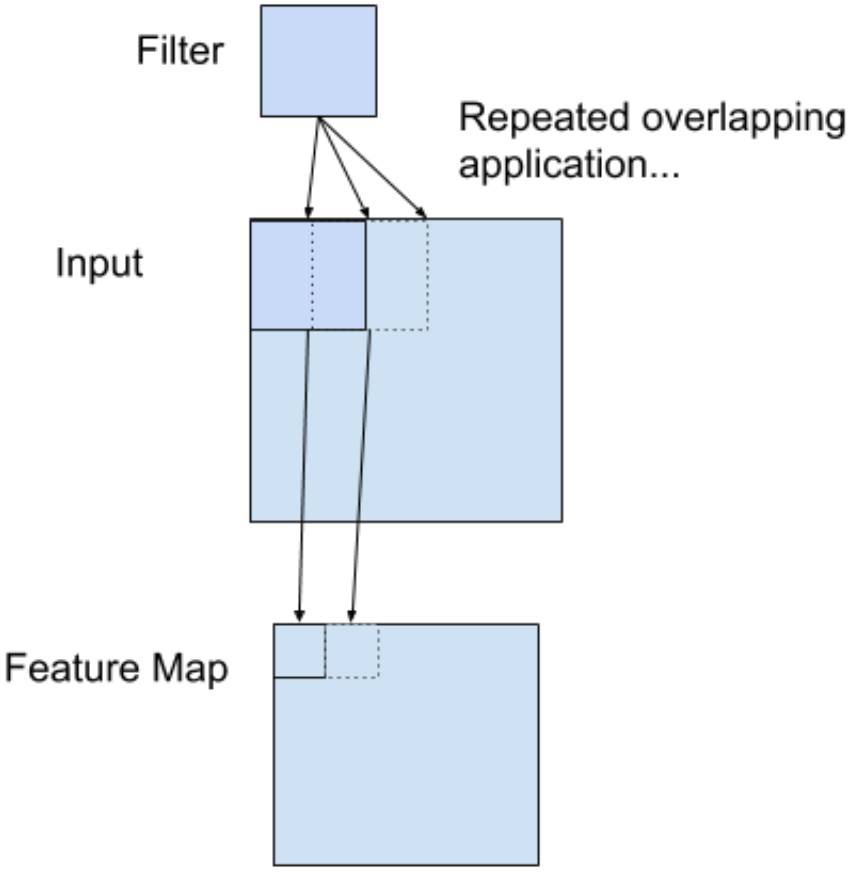

Figure 2.12: Filtering over the input image and constructing the feature map.

As you can see in the Figure 2.13, 3 by 3 filter for vertical edge detection applied to an image with height 6 and width 6 with stride 1 and no padding.

[26]How Do Convolutional Layers Work in Deep Learning Neural Networks?

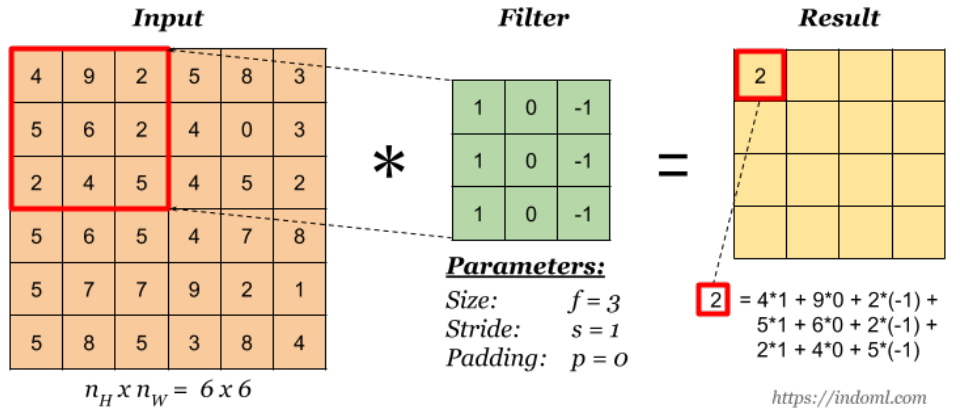

Figure 2.13: Filtering over an input image, representing the convolution operation in CNN.

2.4.5.2. Architecture

A typical CNN starts with the input image, sent to a network, formed by convolutional and pooling layers where the learning of the features of the image is performed. Later the final result of these is flattened, forming the input of the fully connected layers (neural network layers) with a classification (activation) function in the end. This way, the characteristics of the input image are detected in the beginning, and later, the picture is classified according to them. IN Figure **??** you can see a CNN architecture.

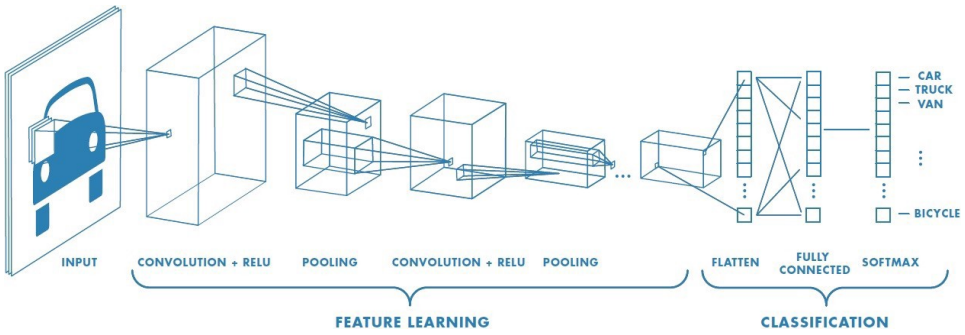

Figure 2.14: A convolutional neural network formed by several features layers followed by classifications.

Chapter 3

Software development methodology

Software development methodologies[1] are used to make software development more efficient and predictable. The two types that exist today are traditional and agile. Traditional methodologies, also known as heavyweight methodologies, follow a linear approach going through those steps - requirements definition, solution building, testing, and deployment. They require defining and documenting all the requirements at the beginning of the process. Some examples of heavyweight methodologies, among many others, are Waterfall, Spiral Model, and Unified Process. Agile methodologies are centered around the idea of iterative development. They are more flexible and focus less on initial planning. Some of the most popular being Scrum, Kanban, and Extreme Programming (XP).

3.1. Kanban

Kanban[2] is a system, oriented towards visualizing the work, making it flow, reducing waste, and maximizing the product value. The primary practices are visualization, usually via a dashboard, and limit the work in progress (WIP). We chose it as a methodology for our project because the product is delivered continuously, and changes are allowed during the whole process, and no estimation of the tasks is needed. That will increase our flexibility and productivity and will decrease the effect of our lack of experience in estimating.

During every meeting, the tutors will give us the tasks - code and memory- that we have to finish until the next one, including their priority. The date for the next meeting also will be decided during the current one, as the sprint may vary between two and four weeks, depending on the given tasks.

[1] Comparison between Agile and Traditional software development methodologies
[2] A Review of Lean-Kanban Approaches in the Software Development

We decided to have the following columns:

- To do. Tasks given to us by the tutors created according to the priority given by the tutors to each task (from higher to lower priority).
- In progress. Tasks we are currently working on. All tasks in that column will have a particular person assigned. As we are using Kanban, we have limited the WIP (Work In Progress) of the column to 4 assignments to avoid doing more tasks than we can reasonably manage. The completed tasks will be moved to 'Testing'.
- On hold. Activities that we can't continue at the moment. The reason behind this is that they are waiting for another task to finish. When the task could be continued, it is returned to the column 'In progress'.
- Testing. In the case of a coding task, the testing will include code review and automated and/or manual testing. In the case of a memory task, the person who didn't write the particular part will read and correct it. If a problem is spotted, the task will be moved to the column 'In progress'. If not, it will go to the column 'Ready for review'.
- Ready for review. Tasks we have finished but are not yet reviewed by our tutors. The tasks in that column will be reviewed during the next meeting and the tutors will decide which of them will be moved to the column "Done" and which ones are not finished and must be remade.
- Done. Tasks that are finished, reviewed and approved by the tutors.

If the WIP of any of the columns does not allow more tasks, any additional ones will be moved to the column 'Hold on'.