SUP'COM
Higher School of Communication of Tunis

P2M REPORT

Mobile application for Tunisian sign language recognition

Carried out by

Rihab Jerbi

Wassim Chakroun

Supervised by

Mr. Riadh Abdelfatteh

2021/2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

TunSL	Tunisian Sign Language
SLR	Sign Language Recognition
ArSL	Arabic Sign Language
CNN	Convolutional Neural Networks
ASL	American Sign Language

GENERAL INTRODUCTION

Deafness is defined as the condition of lacking the power of hearing or having impaired hearing. According to the World Health Organization (WHO), 1.5 billion people worldwide have some degree of hearing loss [1]. By 2050, the number is expected to rise to nearly 2.5 billion people with at least 700 million requiring hearing rehabilitation. In Tunisia, the number of deaf people exceeds 200 thousand and 95% of them are illiterate and suffer from many problems and obstacles [2].

Sign language is the primary means of visual communication between people who are deaf or have hearing impairments or have trouble with spoken language due to a disability, and their societies. It is a mode of communication using movements and hand gestures, facial expressions and body language instead of normal words. Akin to spoken languages, sign languages are not universal, but there are some similarities among them. There are approximately 72 million deaf people worldwide, using more than 300 different sign languages, according to the World Federation of the Deaf (WFD).

Several sign languages have gotten a lot of interest in the research community because it is crucial to increase equality in a society that involves both deaf and hearing people. In this context, this project targets the Tunisian deaf community and aims to provide a smart recognition system for Tunisian Sign Language (TunSL) integrated in an accessible mobile application.

CHAPTER1: STATE-OF-THE-ART

I. Sign language recognition:

Sign Language Recognition (SLR) is a discovery based on new technologies such as artificial intelligence and deep learning to help deaf people break down the communication barriers with other communities. It has been researched for many years but there is still no high-performance solution. Nowadays, researchers are receiving more attention for developing commercially viable solutions. They conduct their studies in various ways. The methods used in each study differ as well. Each strategy has its own strength over others and researchers are still using diverse approaches to build their own solutions [3].

The recent progress in the technological field has geared us towards the further exploration of hand signs recognition with the aid of deep neural networks. SLR provides a deep learning model trained to detect and recognize hand gestures and translate them into corresponding texts or voices as depicted in the figure below.

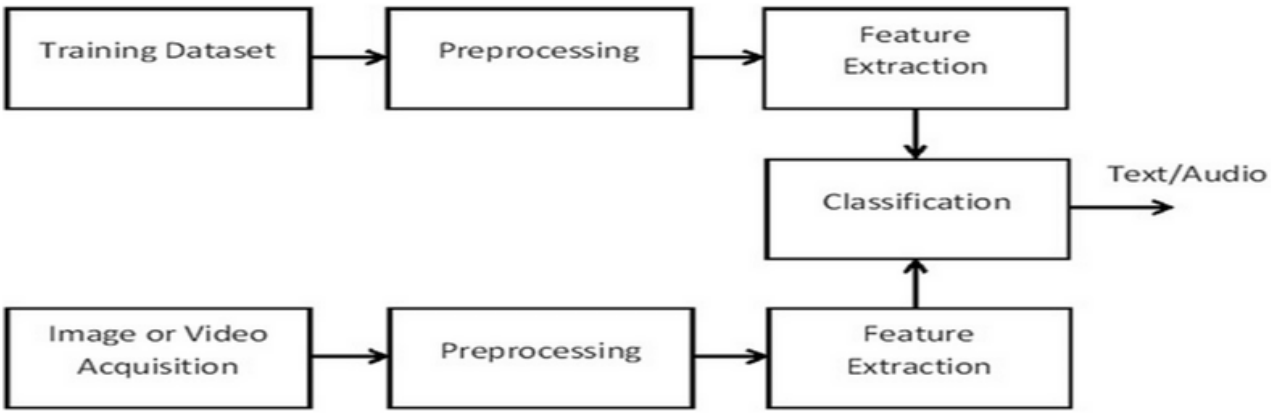

Figure 1.1: Sign language recognition flow chart (ResearchGate)

There are two types of recognition: static and dynamic. Static recognition deals with the detection of gestures from a 2D image. While dynamic recognition is a real-time capture of gestures using the camera. The first alternative can be referred especially if it is difficult to capture frames in a live camera preview due to poor lighting, weather conditions or camera quality.
Research has revealed acceptable solutions. For example, Intelligent models for SLR detects signs from a camera or images.

This solution added tremendous benefits to the world of deaf people. However, its usage is so limited since it was seldom transferred to an accessible and easy-to-use alternative. On the other hand, smart gloves, as shown in the following figure, that translate sign language in real time include a pair of gloves with thin and stretchable sensors that run the length of each of the five fingers. These sensors pick up finger placements and hand motions that stand for individual letters, numbers, words and phrases. This solution is not practical since it contains many sensors and cables and they may not work properly together. It is only developed for western sign languages.

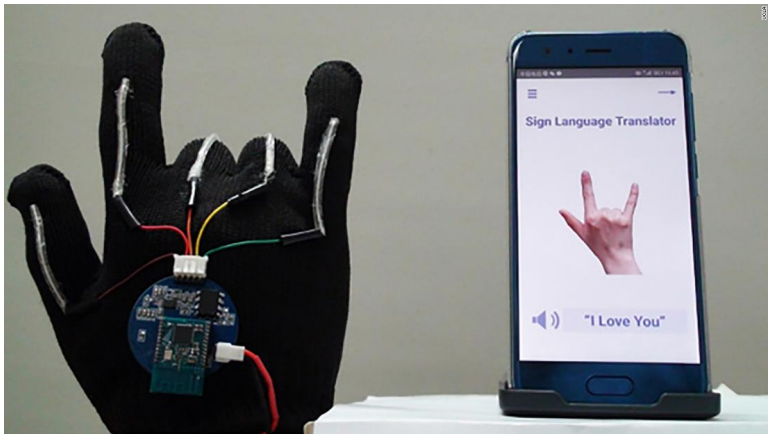

Figure 1.2: Sign language recognition glove (Daily Mail)

II. Arabic sign language recognition:

Arabic Sign Language (ArSL) is a full natural language used by Arab deaf people to communicate within their communities. Deaf people's isolation from society is exacerbated by their unfamiliarity with this language. ArSL is definitely not standardized in all Arab countries. Several sign languages have been developed and evolved depending on the country and can even vary from region to region within the same state like spoken languages and dialects.

ArSL has witnessed remarkable research activities to recognize hand signs and gestures applying deep learning techniques. Most research focuses on sign languages of the Arabic-speaking Middle East. As of that, Saudi sign language has received great attention with attempts to develop a recognition system based on Convolutional Neural Networks (CNN) [4]. One of the CNN architectures used to build the sign language recognition model contains multiple convolutional layers, max pooling layers, dropout layers and ends with flatten and dense layers.

Exploiting the Saudi Dictionary that was prepared and published by the Saudi Association for Hearing Disability in 2018, a dataset of the sign images was constructed and used to train the model. The accuracies of training and testing were over 95% in this research example.

III. Tunisian sign language and problem identification:

In Tunisia, deaf people use TunSL which derives from Italian and French sign languages, combined with indigenous sign. There are unfortunately no official resources gathering the required information of the TunSL.

As mentioned earlier, the number of people having impaired hearing is constantly increasing and the situation in Tunisia is no exception. Tunisan deaf people face enormous difficulties communicating with normal people since most of them do not understand or even know sign language. They frequently find themselves in contexts where verbal communication is the norm. Furthermore, the access to expert interpretation services is not possible in many circumstances, which can lead to underemployment, public health issues and perilously a definitive gate between deaf individuals and society.

An intelligent solution can be adapted from applications working for other sign languages: if TunSL can be accurately interpreted via a deep learning application, even if it only starts with the alphabet and numbers, we can give a prime advantage to our deaf and hard of hearing community in the absence of any resources. In fact, our solution can serve as a baseline reference for a future attempt to document TunSL and a way to unite the vocabulary used by deaf people and sign language interpreters.

To offer our deaf community a greater voice, we attempted to answer this question: Can computer vision and deep learning bridge the gap for the deaf and hard of hearing people by learning TunSL? For the first time in our country, we can give a major benefit to deaf people.

The primary target of this project is building a sign language recognition model using the TensorFlow object detection API and transfer learning and integrating it in a Flutter mobile application.

CHAPTER2: DATASET

I. Existing datasets:

Western sign languages generally have good documentation and dataset availability. For example, American Sign Language (ASL) has many datasets serving for object detection of each ASL letter with a bounding box. Whereas, the lack of datasets, especially for ArSL and related languages, is one of the most significant issues for research.
The figure below summarizes the existing datasets used in some previous works [5]. Several studies based on standard Arabic have been conducted, with the majority of them focusing on the recognition of alphabets, numbers and a limited number of words [6]. Besides, TunSL has never been the main topic of a research or a study and it lacks an official dictionary.

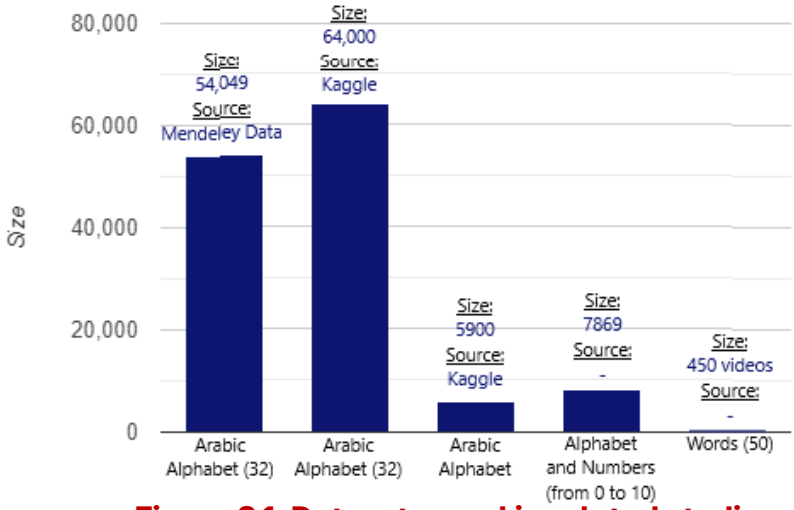

Figure 2.1: Datasets used in related studies

II. Building the very first dataset of TunSL:

1. Data Collection:

Since there is no official documentation of TunSL, we decided to collaborate with Tunisian Association of Sign Language Interpreters (ATILS) to better cover the main words being part of the vital fields of the deaf in Tunisia. The words were selected by the members of ATILS according to their degree of importance in daily life. As we approved them, we filmed videos of 54 signs belonging to the following categories: requests, destinations, family, transport and days : عم, عسلامة, عائلة, نعاونك, الخميس, أخ, إنت, تحب, طلب, اليوم, برنامجك, شنوا أحوالك, أصم, تعرف, تقرا, نكتبلك, مرحبا, الصحة, لباس, أب, ابن, جد, مرأة, أم, طفل, السبت, الأحد, الاثنين, الثلاثاء, الأربعاء, الجمعة, بلدية, بانكة, بوسطة, دار, الثقافة , محكمة ...

2. Data preparation:

We extended the dataset to the signs that contain not only one gesture but also multiple gestures and we enriched it with the contribution of some of our friends and relatives. OpenCV, which provides real-time optimized computer vision tools, was used to capture the images with computer cameras so that the system could detect later any images taken with primary cameras (computer, phone…). Then, the processing plan was as follows:

a. Key Frame extraction:

We extracted frames from videos containing signs using OpenCV. We acquired sets of frames that have a good representation of the shots. They preserve the salient feature of the shots, while eliminating most of the repeated frames (data filtering according to brightness, clarity and so on)

Figure 2.2: Extracting frames from a video

b. Data augmentation:

As we want to avoid overfitting, one way to do this is data augmentation. The dataset size can be artificially enhanced by modifying the images in such a way that preserves their real meaning.

Cropping:

The images were cropped to better fit the model with respect to the input size of the pretrained model. They now focus on the positioning of the hands more than the entire upper body, slashing the confusion generated by different faces for the same sign.

Flipping horizontally:

Flipping rearranges the pixels while protecting the features of the image. Vertical flipping is not meaningful in our case since most of the signs would lose their meaning and the confusion level increases. While horizontal flipping ensures that both hands are used to perform the signs originally done with one of the hands.

Converting to grayscale:

The color of images is changed by new pixel values. It is converted to grayscale because it renders flawless skin tones, smoother shadows and gradients and brings out more intricate details.

c. Labeling:

The images were finally labeled using « LabelImg » Python tool which is a graphical image annotation tool. It's an easy and free way to label a few hundred images to try out an object detection project and teach a model to recognize different objects [7]. As an output, we obtain XML files containing each the name of the image, the corresponding label and the coordinates limiting the sign to be detected.

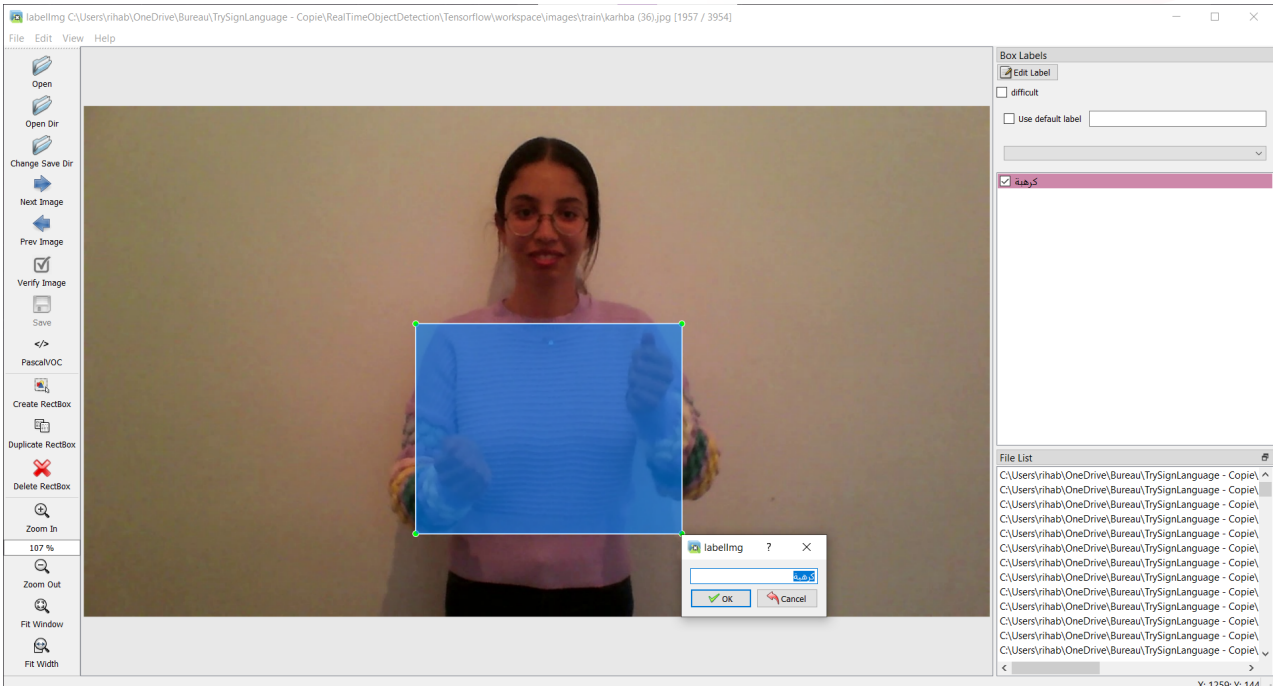

Figure 2.3: Labeling images with LabelImg

d. Dataset splitting:

The resulting images along with their correspondent XML files (total size=9510) are split into two sets:

- Training: about 80% of the data, used for training the model.

- Test: rest of the data, used to evaluate the performance of the model.

CHAPTER3: TUNISIAN SIGN LANGUAGE RECOGNITION BASED ON DEEP LEARNING

I. Inspiration : why a deep learning model?

Deep learning attains great flexibility and power by: its ability to deal with large volume of data, multiple labels which have a variety of representations and learning to represent the world as a nested hierarchy of concepts without losing on the quality of models.

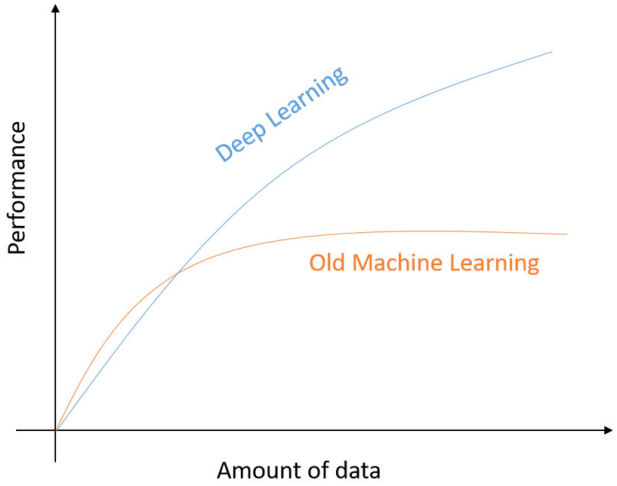

Figure 3.1: Performance and dataset size (ResearchGate)

The selected deep learning framework TensorFlow is the leading open-source library for developing and training machine learning and deep learning models. It was originally built by the Google Brain team and the reason why a lot of data scientists tend to use TensorFlow is that it makes constructing models easier and faster by providing a rich collection of tools that includes data pre-processing, model evaluation..

As we mentioned before, several sign language models have been implemented since a lot of people worldwide choose to work on this topic. In fact, there are several SLR techniques used in precedent studies. Many of them were a source of inspiration and helped us gain a great insight of the scope of the required work:

- The study of Hayani et al. [8] developed a system using CNN. The proposed model was inspired by fbyLeNet-5. They used four layers to extract deep features and three layers to classify them.
- The work of Ibrahim et al. [9] used Euclidean distance for feature extraction for the ArSL. Hand segmentation was applied to the different frames of a video-based dataset.

- All those studies consolidated the work of Al-Obodi et al. for Saudi Sign Language [10]. Otherwise, the study of Steve Daniels for Indonesian sign language recognition used a state-of-the-art real-time object detection system called YOLOv3. Only a single CNN is used to predict what objects will be in the image [11]. This can happen because YOLO divides images into cells that predict the number of bounding boxes, the confidence level, etc…

II. Implementation:

The implementation process was structured in five important steps aiming to provide a consistent model at the end.

a. Data preprocessing:

The work process of the recognition model begins with creating a Label Map which is an ordered representation of the different objects or labels to be detected by the model. Then, we created TensorFlow Records (TFRecords) from the Label Map to store data and images as a sequence of binary records used by the object detection API for a better performance on the training time of the model.

b. Pretrained model selection:

Next, it is not possible to use the existing model during development when taking on the object detection API. That's why we chose from a list of models in the TF Detection Model Zoo as depicted below.
The pretrained model is « SSD MobileNet V2 FPNLite 320x320 ». Its speed is 22ms and its accuracy is 22.2. In fact, a higher COCO mAP or accuracy and a lower speed are preferable because it is the inference speed, or more specifically the time needed for the network to provide an output according to the inputs.


TensorFlow 2.2 Python 3.6

We provide a collection of detection models pre-trained on the COCO 2017 dataset. These models can be useful for out-of-the-box inference if you are interested in categories already in those datasets. You can try it in our inference colab

They are also useful for initializing your models when training on novel datasets. You can try this out on our few-shot training colab.

Please look at this guide for mobile inference.

Finally, if you would like to train these models from scratch, you can find the model configs in this directory (also in the linked `tar.gz`s).

EfficientDet D6 1280x1280	268	50.5	Boxes
EfficientDet D7 1536x1536	325	51.2	Boxes
SSD MobileNet v2 320x320	19	20.2	Boxes
SSD MobileNet V1 FPN 640x640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320x320	22	22.2	Boxes
SSD MobileNet V2 FPNLite 640x640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640x640 (RetinaNet50)	46	34.3	Boxes

Figure 3.2: TensorFlow Detection Model Zoo

c. Transfer Learning:

Our approach is to exploit transfer learning which consists in using a neural network that has already been trained on another dataset, modify the decision part of the network to fit our need and retrain it. This way reduces the amount of data needed, as long as weights are already trained on something else. In our case, transfer learning was fulfilled by updating the configuration file of the pretrained model according to the characteristics of the model in process.

d. Model Training:

It is based on the execution of the python file “model_main_tf2” belonging to the object detection API allowing to create and run TF2 object detection models. It is required to specify the number of training steps to be performed. A first trial with 10000 steps wasn’t really efficient, that’s why the number was set to 20000 since there are multiple complex signs and the dataset is pretty large.

e. Checkpointing:

After training our model on the collected data, we are able to load the trained model from checkpoints produced automatically during the training process. These checkpoints save the exact values of all involved parameters and weights. The last one notably conserves the least amount of loss and serves to tune the model, restore it for evaluation purposes and set the real time detection from there.

III. Result:

The outcoming model is composed of 70 layers except the input and output layers. This result was verified by the « Netron » tool used to visualize neural network, deep learning and machine learning models.

In order to evaluate the performance of the model, we tested it with a computer camera using the open-source library OpenCV. We made a prototype that can recognize the gestures performed by the person in front of the camera, delimit the sign with a frame and display the corresponding word accompanied by its score which must be greater than the minimum score threshold (= 0.8). This can be shown in the following figure.

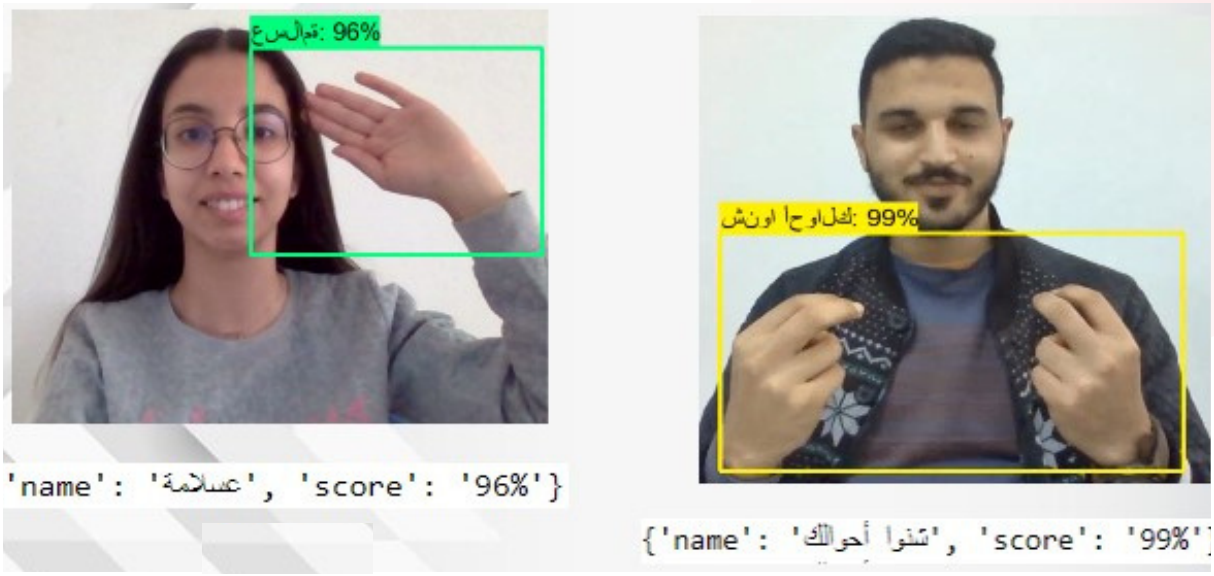

Figure 3.3: Words detected by the model

Since the model encounters many confusions and errors and focuses excessively on faces, we decided to work temporarily with a smaller model able to properly recognize 7 of the 54 signs, as mentioned below, and use it to complete the mobile application on time.

CLASS	ACCURACY	# SAMPLES
تحب	1.00	7
شنوا أحوالك	1.00	9
دار	1.00	12
وزارة	1.00	13
نعاونك	1.00	14
برنامجك	1.00	14
أم	1.00	7

Figure 3.4: Final model limited to 7 signs

CHAPTER4: TUNSL MOBILE APPLICATION

I. Conception:

1. UML diagrams:

UML serves to clearly define the needs of the target population, to specify all the functionalities of an application and thus to avoid possible over-costs or delays. In addition, UML modeling provides a quick understanding of the program to other external developers in case of enhancement of the software and facilitates its maintenance.

For all of its benefits, we decided to use UML modeling. It helped us realize the scope of our application through use case and sequence diagrams. We have foremost fixed the target population: people who are deaf or have hearing impairments or have trouble with spoken language due to a disability. Then, we have set the main features of the application as mentioned in the use case diagram below.

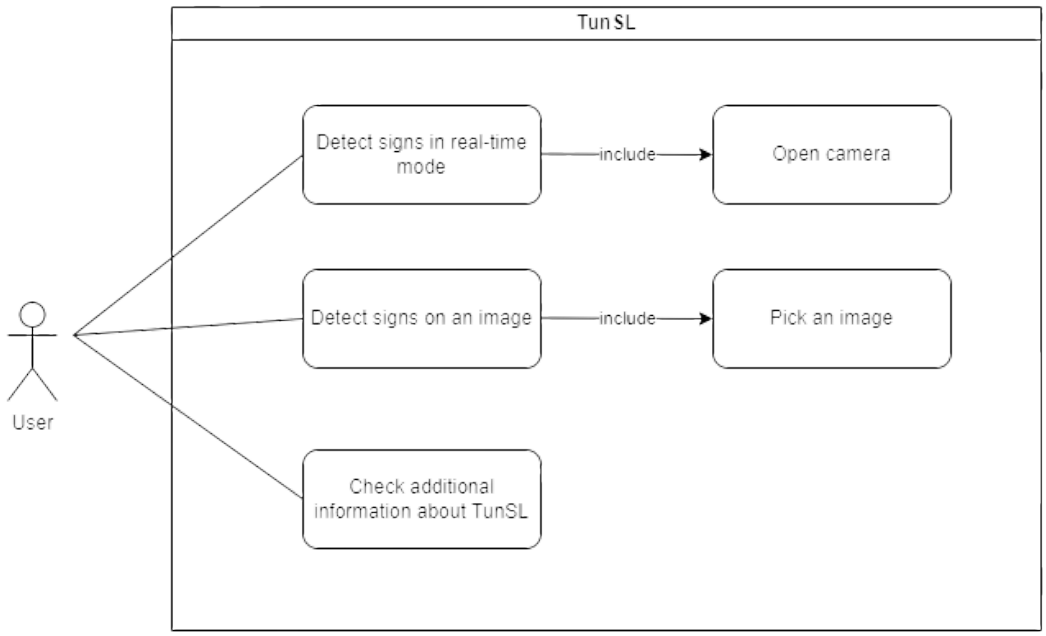

Figure 4.1: Use case diagram of TunSL

In fact, users are able to either open the phone's camera to detect signs in real-time mode, or pick an image from the phone's storage to detect signs on it. Otherwise, they can check additional information such as tips on how to deal with deaf people and a description of the application and its features.

On the other hand, the sequence diagram below details a bit the use cases included in the previous diagram. It is designed as a fragment of interaction with the alternatives operator "alt": a conditional operator with several operands. This is the equivalent of a multiple-choice process. Only the sub-fragment whose condition is true is executed.

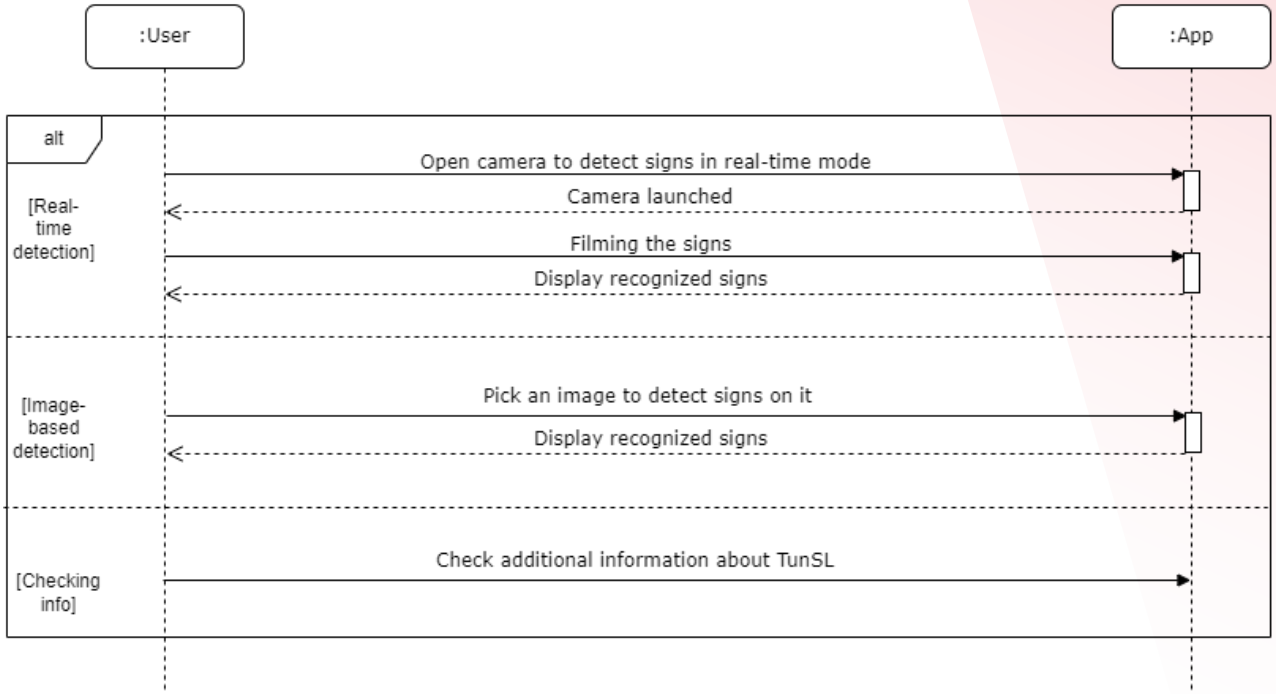

Figure 4.2: Sequence diagram of TunSL

2. Design:

It's evident to ask the following question “how did the design come to be?”. Since the application has two primary features about sign language detection and recognition, we have focused on the aesthetic side of the user interfaces.

The logo represents the unified sign of love in almost all sign languages reflecting our care and support for discriminated communities and the necessity of harnessing advanced technologies for their sake. The choice of the color palette respects the tip of preference for cold colors when dealing with deaf people since they are peaceful colors and give a sense of calm and relaxation anywhere.

Figure 4.3: Logo and color palette

We divided the user interfaces into approximately seven different interfaces:

- Splash screen: the welcome interface
- Home page: the two basic features of the app
- Real time detection page: detection on frames in a live camera preview
- Static detection page: detection on images picked from the gallery
- Help page: some advice on how to use the app's two primary features
- About page: the description and the developers of the app
- Tips page: the key elements to properly deal with deaf people

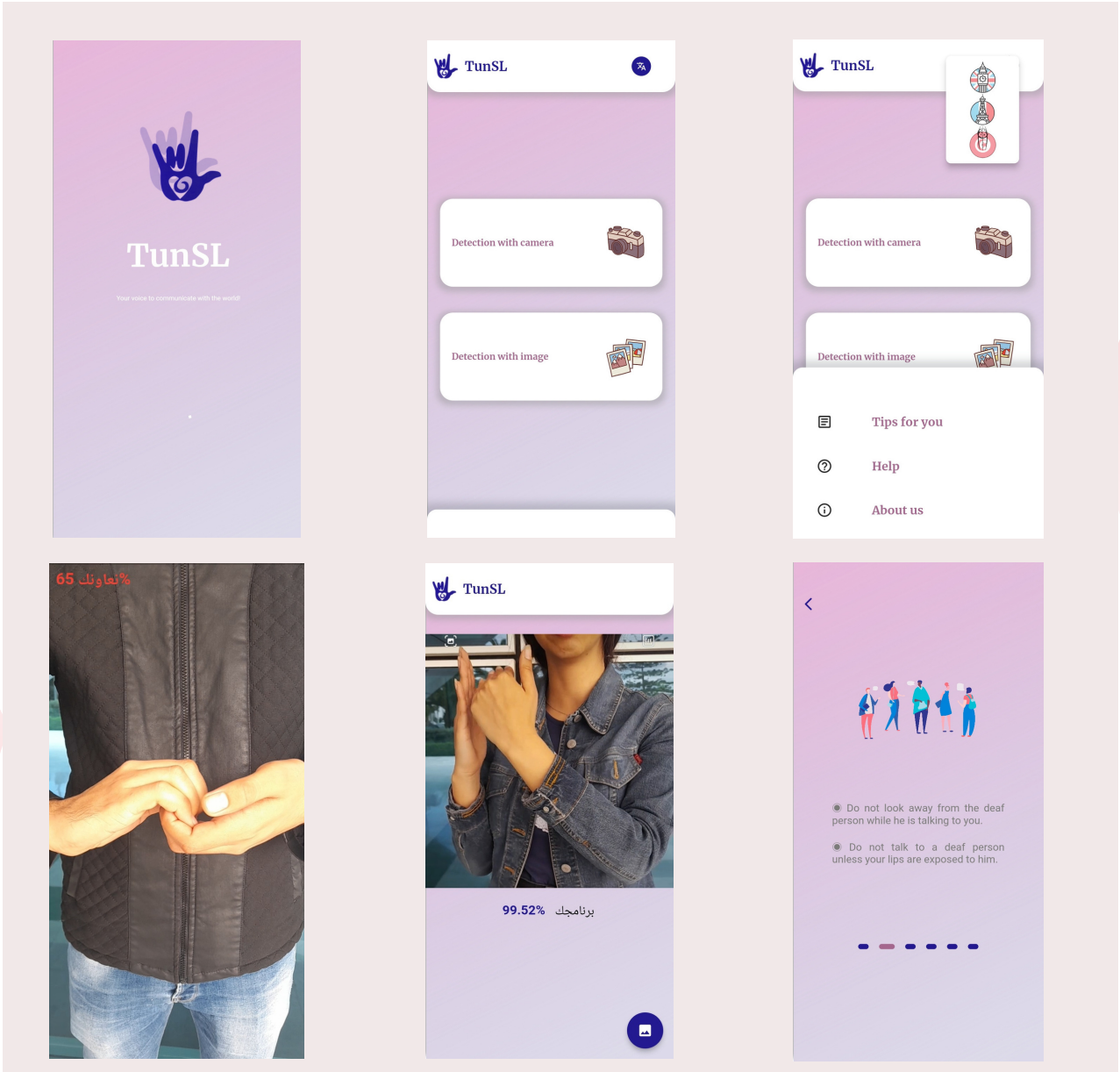

Figure 4.4: Some interfaces of TunSL

II. Development:

1. Model integration:

If we want a tensorflow model to be used in a mobile app, it needs to be converted to a TensorFlow Lite (TFLite) object. This can be fulfilled by the open-source « tflite » package which allows the model's integration within a Flutter app [12]. It is a plugin for accessing tensorflow lite API. It supports image classification, object detection and other options on both iOS and Android platforms.
In order to ensure both types of sign language recognition, we also need to exploit the open-source camera plugin. As image streaming capability was added in this plugin, it has allowed capturing frames in a live camera preview. Therefore, it is possible to create an app that uses image streaming with tflite plugin to achieve real-time object detection in Flutter.
We started afterwards by generating a TFLite-friendly intermediate Saved Model and then use the TensorFlow Lite Converter to convert this generated model to TFLite object. It must be accompanied by a text file serving as a store that contains the labels of the signs recognized by the model to extract them at the time of detection.

2. Extra features:

As it is important to incorporate the choice of language regarding the user's preference, we added the translation feature to the app guaranteeing a wider range of users of different intellectual levels. This was achieved with the « translator » package based on Google Translate API for Dart programming language. Besides, communication can be a struggle for some people when dealing with deaf people since there are several behaviours considered rude by them. As of that, we selected the key elements to establishe good communication with the deaf and hearing impaired people in the form of easy-to-assimilate illustrative pages. Eventually, we provided some help instructions about the two main functionalities of the app (real time detection and static detection on images) to simplify their use.

CONCLUSION

This report is elaborated to highlight all the details related to our career preparation project « Mobile application for real time translation of Tunisian sign language ». This project provided an opportunity to apply our acquired knowledge, learn and practice new skills and cope with the challenges that come with working on a real-life project. Furthermore, working on a topic such as sign languages is really interesting as it combines different technologies ranging from deep learning to computer vision and mobile development.

We covered the broad spectrum of real-time Tunisian sign language recognition using TensorFlow starting with the process of data collection using sign language capturing methods to build the very first dataset of TunSL, moving on to the implementation process of TunSL recognition model based on deep learning and ending with the work done to design and develop TunSL mobile application. We were able to implement a prototype that allows the translation of some gestures of the sign language used by deaf and dumb people in Tunisia and therefore offer this community a greater voice even if we only starts with a limited number of signs (7 out of 54 collected signs) due to the limitation of hardware resources.

Some critics of technologies related to sign languages have argued that the potential of intelligent models is commonly confused or degraded, because many sign languages have a complex grammar that includes use of the sign space, movements and facial expressions (non-manual elements) that confuse the model and increase its variance.

For future improvement of our project, we can add text-to-voice translation of the words recognized by the model so that it could help normal people get the meaning aimed by deaf individuals. In addition, we may share the constructed dataset as a beneficial open source for other researchers. We can also think of another complementary project that achieves the translation of speech into sign language by using speech detection and recognition.

BIBLIOGRAPHY

- [1] World Health Organization, Deafness and hearing loss, 1 April 2021. [CrossRef]
- [2] Ramzi Al-Ayari, journalist from Tunisia. Sign language in Tunisia..a cruel society and a neglected country, Ultratunisia ultrasawt, 27 February 2020. [CrossRef]
- [3] T.-Y. Pan, L.-Y. Lo, C.-W. Yeh, J.-W. Li, H.-T. Liu, and M.-C. Hu, "Real-time sign language recognition in complex background scenes based on a hierarchical clustering classification method", IEEE Second International Conference on Multimedia Big Data (BigMM). IEEE, 2016, pp. 64–67
- [4] Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network [CrossRef]
- [5] Latif, Ghazanfar; Alghazo, Jaafar; Mohammad, Nazeeruddin; AlKhalaf, Roaa; AlKhalaf, Rawan (2018), “Arabic Alphabets Sign Language Dataset (ArASL)”, Mendeley Data, V1, doi: 10.17632/y7pckrw6z2.1 [CrossRef]
- [6] Muhammad Usman, Arabic sign language binary image dataset, Binary image dataset for hand gesture classification. [CrossRef]
- [7] Joseph Nelson, Getting Started with LabelImg for Labeling Object Detection Data, Roboflow, MAR 16, 2020. [CrossRef]
- [8] S.Hayani, M.Benaddy, O.ElMeslouhi, and M.Kardouchi, "Arab sign language recognition with convolutional neural networks," in 2019 International Conference Of ComputerScience and Renewable Energies (ICCSR), July 2019, pp. 1–4.
- [9] N. B. Ibrahim, M. M. Selim, and H. H. Zayed, "An automatic arabic sign language recognition system (arslrs)," Journal of King Saud University - Computer and Information Sciences, vol. 30, no. 4, pp. 470 – 477, 2018.
- [10] Alaa H Al-Obodi, Ameerh M Al-Hanine, Khalda N Al-Harbi, Maryam S Al-Dawas, and Amal A. Al-Shargabi. "A Saudi Sign Language Recognition System based on Convolutional Neural Networks", Department of Information Technology, College of Computer, Qassim University, Buraydah, Saudi Arabia.
- [11] Steve Daniels et al, "Indonesian Sign Language Recognition using YOLO Method", 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1077 012029.
- [12] Miqudo, How To Build A Computer Vision Mobile App In Flutter, Digital Knights, 04 Feb 2020. [CrossRef]