Embedded Deep Learning to Aid the Mobility of Individuals with Disabilities: A Solution for In-house Bus Line Recognition

Tashko Pavlov[0000−0001−7689−4475], Nikola Stanojkovski[0000−0001−9555−963X], Mario Stojchevski[0000−0002−0662−6421], Kostadin Mishev[0000−0003−3982−3330], and Monika Simjanoska[0000−0002−5028−3841]

Ss. Cyril and Methodius University
Faculty of Computer Science and Engineering,
Rugjer Boshkovikj, 16, 1000 Skopje, N. Macedonia
{tashko.pavlov, nikola.stanojkovski,
mario.stojchevski}@students.finki.ukim.mk,
{kostadin.mishev,monika.simjanoska}@finki.ukim.mk
http://www.finki.ukim.mk

Abstract. The mobility of individuals with visual impairments is a significant challenge as the cities are becoming more and more crowded each day. The technology is rapidly developing, offering novel high-tech smart white canes to aid the mobility of individuals with partial or total blindness. However, they are hardly affordable due to the high prices. Even more, they are impractical for in-vivo usage as they depend on third-party technologies and services, which require an Internet connection for data transfer and data processing on Cloud services. In this paper, we offer a novel methodology that aids the transportation of blind individuals, which is entirely integrated into the chip, thus avoiding the need for an Internet connection. Our methodology embeds three intelligent Deep learning models on a single smart mobile device, one model to localize the position of the bus line number approaching the individual, the second model to recognize the bus number, and the third is a text-to-speech model, which synthesizes speech to notify the individual in a pleasant and human-like manner about the number of the approaching bus. Our work presents one step closer to the completely independent embedded intelligent models that simplify the transportation of visually impaired persons using cutting-edge tools from AI.

Keywords: Intelligent embedded systems · Deep learning · Disabilities · Assistive technology · Computer vision · Image processing · Speech technologies · Mobile technologies.

1 Introduction

According to the World Health Organization (WHO) [17] 285 million people are estimated to be visually impaired worldwide in 2014. The later statistics provided

by WHO in 2020 show that this number has reached approximately 1.3 billion people [14]. The number of blind and visually impaired people is expected to increase over the coming years, and it is estimated to triple by the end of 2050, which is quite alarming. In our country, North Macedonia, where we conducted the experiment, there are nearly 2500 people with more than 90% damaged sight according to the statistics of the Union of Blind Persons [16]. The most common causes are cataracts, glaucoma, uncorrected refractive errors, diabetes, retinal displacement, ocular trauma, and aging-related diseases [5, 3, 25].

Daily activities still represent real challenges for visually impaired individuals due to the lack of affordable, appropriate assistive devices. The absence of assistive tools triggers an infinite loop of inappropriate education at its basics, followed by limited lifestyle development that leads to frustration, low confidence, reduced autonomy, and often physical safety risks [9]. There are several methods and devices that are used to guide visually impaired persons, and all of them have their advantages and disadvantages [26]:

- White cane: A blind person using swing-like movements scans the path in 1m distance in front. The cane requires about 100 hours of training for skilful use, like detecting drop-offs, walking up and down the stairs. Some of the advantages are: foldable and adjustable, effectively informs of shorelines, landmarks and obstacles at ground-level, notifies others about visual disability of its user, and they are cheap and light-weighted if not in a form of smart white cane with additional electronics. The main disadvantage is that it does not protect from obstacles at torso and face level.
- Guidance of dog: A specially trained dog assisting the blind in obstacle avoidance. It does not aid in finding a way (only a familiar path). It is trained to stop before obstacles, and reacts to commands on walking directions. Guide dogs are a rarely used aid - only about 1% of the visually impaired use it. The advantages of this method is that a dog is good in following familiar paths, good overall obstacle avoidance, and can be trained for selective disobedience when sensing danger to his owner. The disadvantage is that it is very costly, sometimes the training can cost up to $40k; the guide dog service period is on average 6 years, and there are regular dog up-keeping costs and lifestyle changes.
- Human Guide: A blind person walks hand in hand with a sighted guide. This is not a permanent solution. Even though there are many obvious advantages, the lack of privacy and feeling of being a burden to his or her guide are disadvantages that cannot be ignored.

Among the presented methods, probably the most commonly used device for visually impaired persons is the white cane. Since its initial appearance in the 1930s, the long cane's design has not undergone many innovations. These canes for the blind can detect objects immediately in front of them, but they cannot detect objects on all sides, such as low-hanging branches, nor can they help a person navigate a city. This might put users in physical danger [10]. According to a research [13] 40% of 300 blind participants suffer head accidents at least

once a year, and 23% reported medical consequences. This has led the research in introducing novel smart white canes as the technology has had its expansion in the recent years. Several research works are being performed to offer the best navigational electronic (smart) white cane in terms of cost effectiveness, however, unfortunately this feature is not among the pros of those devices. There are few models of electronic smart white canes commercially available. Among the most popular commercial smart white cane is WeWalk [27], but also very expensive ($500 in the US), while a traditional white cane typically costs from $20 to $60. A cheaper device up to our knowledge is MiniGuide [8] that costs $330 and comes in a form of an additional device that should be attached to the traditional white cane. Beside the cost, one of the main disadvantage is the weight that all the electronics puts on to the cane, making it hard to use. There are some smart canes trying to solve this problem by introducing novel lightweight materials. One representative is UltraCane [28] which is a carbon graphite collapsible cane. However the price is far from affordable (around $800).

Even though smart white canes are proven to be useful tools for the blind, they are still especially satisfied with the traditional white canes since their weight is much less than the weight of the smart white canes that include additional electronic components. A survey on the best traditional white canes [4] show that they characterize as lightweight (usually aluminum), affordable, and can be folded up easily for compact storage when not in use. Also they feature white and red reflective colors for visibility at night time (VISONU). Some of them are designed to absorb the shock when a sharp object is hit, allowing for hand and wrist comfort. Traditional white canes are durable and therefore can be used even in snow conditions (NO JAB). Some are even designed for hiking, featuring a wrist strap and a non-slip handle (TIMISHON). Often the plastic used is textured for easy gripping (Lumex).

Considering the fact that all the developed smart white canes require a smart phone device connection, a question is raised on the problem why there is a need of overwhelming the traditional white canes with electronics, when none of them process the data in-house. We believe the add-on electronics can be avoided for some problems and smart phones's power can be used to collect and process data without the use of Internet connection, acting as completely independent and intelligent embedded device. In this paper we propose a proof-of-concept that solves an image processing problem that is in front coming bus line recognition, and a speech generation problem of providing a sound notification to the blind person about the bus line approaching to him/her, by embedding three Deep learning models in a mobile device. Thus, our intention is to consider the satisfaction of the traditional white canes that are not overwhelmed with high-tech equipment adding on weight to the cane, and to provide a solution that can be embedded on a blind person mobile device.

The main contributions of this work are as follows:

– Development of a Deep learning model for bus line number recognition.

- Development of a high-quality dataset BusLineDetection of photos that contain bus(es), taken on bus stops in Skopje, which consists of 862 images. The dataset is publicly available.
- The first integration of an advanced-level Text-to-Speech (TTS) model on a mobile device.
- Integration of two Deep learning models for bus line number position assessment and number recognition on a mobile device.

The rest of the paper is organized as follows. In Section 2 we explore some solutions that are close to the problem at hand. The dataset we have created is presented in Section 3. In Section 4 the methodology and the architecture of our solution are presented. The experiments and the results are discussed in Section 5. In the final Section 6 we conclude the work and present directions for future research.

2 Related Work

The vOICe [15] is an assistive visual technology system for the totally blind. This system uses a head-mounted camera to scan the surroundings in front of the user, process the image, and convert that image data into sound mappings. Using the occipital lobe of the brain, the user can then make a mental picture of the environment.

Tyflos [2] initially developed by Dr. Bourbakis, makes use of a portable computer, cameras, GPS sensors, microphones, language processor, a text-to-speech module, and a vibrating belt to aid the visually impaired during navigation. The system scans the environment and creates a depth map of the three-dimensional space. This depth map is then translated into a tactile language and fed to the actuators of the belt. The belt vibrates on different parts of the body to inform the user of the objects around them.

Shankar Sivan and Gopu Darsan from Sree Buddha College Of Engineering in India have proposed a system for the visually impaired, which states to be cheaper and more accessible than the systems mentioned above. The system [23] consists of an object detection module, which uses BRISK [11], a text detection module, a door detection module, and a security module for detecting intruders in front of the visually impaired person. It is implemented using C++, OpenCV for object detection, and Tesseract OCR [24] for the text recognition module. The user is alerted to the output using a Speech System as a separate module. Since their system uses open-source software, the estimated cost would be 50$ - 60$.

FingerReader [22] is a small device that can be worn on a finger. This system uses the Tesseract OCR to detect the finger, making sense of where the horizontal line is to start processing the text. Right after that FingerReader uses the same OCR system to process the text image and extract words from it, ultimately feeding them into a text-to-speech module in the end. Despite some limitations of the system discussed in the paper due to auto focusing of the camera, it has proven to be useful.

The smart glass [18] is a product in the form of glasses that visually impaired people can use. By designing the glasses in a way standard glasses are designed, the smart glass ensures that the users are not differentiated from ordinary people. The smart glass has embedded ultrasonic sensors which reflect sound waves and calculate the distance of fast-moving objects. This information is presented via vibration on hand bands that the user wears.

Similar work to the problem at hand is presented in [7]. Three main models are installed on a mobile phone and detect bus line numbers to aid the visually impaired. The first model locates the bus from an image, the second model locates the bus line number from the ROI (Region of interest) of the output from the previous model. Lastly, bus line number extraction model is used to detect the number. They use several image processing techniques for the OCR model to recognize the number detected from the object detector model. The novelty we introduce in our approach is the state-of-the-art methods for image processing that was not available at the time the research was conducted, and the text-to-speech model, without which such a system would be useless for the blind. Considering the image processing methodology they proposed, our approach optimizes the recognition by omitting the part where the bus is recognized.

3 Methodology

This section presents the key components of our methodology described in the following subsections. The proposed methodology is developed and verified for the public transportation service in Skopje, N.Macedonia, but it allows facile scale-up and adoption in any city that provides public transportation.

3.1 Dataset

The BusLineDetection dataset comprises 862 images taken with an iPhone 7 in the city of Skopje, North Macedonia. Each image initially is 6Mb in size and has pixel dimensions of 4032x3024. We used two preprocessing steps. Auto-orienting the images and scaling or resizing them down to 416x416 pixels led to a size of around 16-40kB per image. After preprocessing, an augmentation method was applied to the dataset using the RoboFlow [1] tools. In this way, each image produces three new images with a random type of image modification. These modifications were:

- rotation from -10° and +10°;
- saturation tweak between -40% and +40%;
- brightness tweak between -40% and +40%, and
- noise was applied to images for up to 2% of the image pixels

The augmentation is depicted in Figure 1 where four images are shown from the training set. This version of the dataset is comprised of 2331 images in total.

[1] https://roboflow.ai/

The training set portion has 2200 images, the testing set has 43 images, and 86 images are used for validation. By observing a single photo, it is clear that it represents the frontal view of a bus heading towards the camera. Mostly, two important features should be identified on the bus. The digital number and the non-digital number. A single bus can have both or just one of the features, e.g., the bus may only have a digital number.

After annotating the initial dataset of 862 images with those two features as labels, the dataset showed 1142 annotations, 956 of them were in the class of digital numbers, and 186 were non-digital annotations. The non-digital class is underrepresented, however, we are more interested in the digital number annotations, which will be further classified using appropriate bus line numbers.

Fig. 1: Training set sample images.

The dataset is briefly summarized within the two tables below. Table 1 gives an insight regarding the classes which represent the bus line numbers and the number of images for that particular class.

Table 1: Overview of bus lines and the corresponding number of images for that particular bus line

Class	2	3	4	5	7	9	11	12	15	21	22	23
Number of images	11	13	31	36	50	7	3	3	17	8	59	35
Class	24	25	27	31	33	41	44	45	47	50	53	57
Number of images	29	5	38	12	10	78	7	28	43	49	8	6
Class	58	59	60	62	63	65	66	67	70	71	74	/
Number of images	8	5	9	3	3	71	7	12	32	10	24	/

As seen in Table 1 some of the bus line numbers are more common in the dataset than others. For example we have composed a dataset of 59 images for the bus line number 22, but only 3 images for number 11. As we can see in the following sections such unbalanced dataset does not have an impact on the final methodology that we present in this paper, since the model only detects the position of the displayed bus line number rather than the number itself.

3.2 Image Processing

As mentioned in the previous subsection, the images that are used as an input to the model are augmented and resized appropriately to fit the requirements of the model. For the process of recognizing the bus line number of an incoming bus, two key models were used:

- Object Detection model
- Optical Character Recognition model.

The model used for detecting the bus line number for an incoming bus image is YOLOX [6] model. The YOLOX model is an improvement to the YOLO series being a new high-performance detector. YOLOX is a single-stage object detector that modifies YOLOv3 [19] and uses a DarkNet53 backbone.
The YOLOX model is trained on the COCO [12] dataset and can detect common objects, therefore is not fit for our problem of detecting a bus line number. For this reason we pre-trained the YOLOX-s model from scratch.

The other model used for recognizing the number from an image, i.e. the optical character recognition model that we used is EasyOCR [2]. The implementation of EasyOCR is based on several different papers and open-source repositories. The key detection execution model uses the CRAFT [1] algorithm. The CRAFT algorithm is a scene text detection method which is based on neural networks. The recognition model used is a CRNN [21] neural network model which is constructed of 3 main components: feature extractor, sequence labeling, and decoding. We used the pre-trained models provided by EasyOCR.

To achieve our goal of detecting bus line number, we constructed a pipeline combining the aforementioned models in a sequential order. Figure 2 depicts the path that a particular image must take when achieving its purpose. The first model that the image interacts with is the YOLOX model, this model detects the location of the bus line number from the image. Therefore it can detect two classes: a digital number and non digital number. After detection, the pipeline crops the initial image in the coordinates where the class was detected. Depending on the detected class, the pipeline does different preprocessing steps. When the detected class is non digital number, the cropped image is sent to the EasyOCR model and the output is returned. However, when it comes to the digital number class there are some preprocessing steps that must be done in order for the EasyOCR model to recognize the number. Since many different types of digitally displayed numbers on a bus are present, sometimes the preprocessing step that worked on one type of digitally displayed number does not work on another and vice versa. To overcome this issue we implemented two different preprocessing steps. The first one is by converting the initial image to HSV color space and second one is a simple threshold.

[2] https://github.com/JaidedAI/EasyOCR

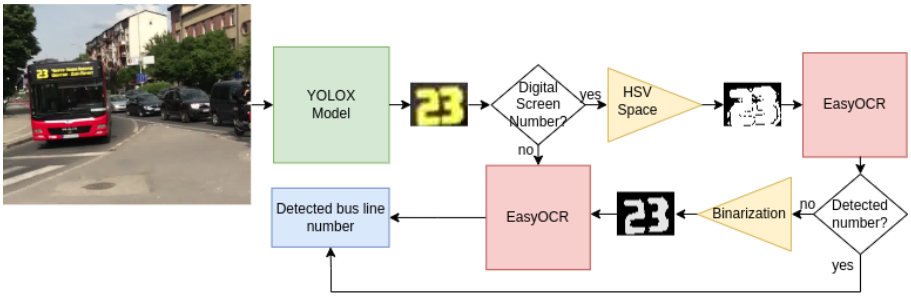


Fig. 2: Bus line number detection pipeline

3.3 Model Training

In this section we will examine the hyperparameters and training configurations for the YOLOX-s model. As stated in the previous section, the YOLOX-s model was pre-trained from scratch on detecting bus line number. For the purpose of training such model the scripts provided from Roboflow [3] were used and trained our model effectively utilizing Google Colab's free GPU provided resources.

Table 2 shows the hyperparameters used for training the YOLOX-s model. We trained the model for 40 epochs, used variable learning rate which changes during each iteration and a batch size of 16.

Table 2: YOLOX model training hyperparameters

Hyperparameters	**Value**
Number of classes	2
Input size	640x640
Batch size	16
Warmup epochs	5
Epochs	40
Basic learning rate per image	0.00015625
Minimum learning rate	0.05
Number of parameters for training	8.94M

3.4 Text-to-speech

The FastSpeech 2 [20] model for Text-to-Speech inference is the final deep learning model we implement in our method. The key role of such model is to read out loud the bus line number that is detected using the previously described

[3] https://models.roboflow.com/object-detection/yolox

models. The reason that FastSpeech 2 was chosen as TTS model is because it is a non-auto regressive model that makes synthesizing speech significantly faster and with comparative quality. The FastSpeech 2 model is an improvement of the FastSpeech model in generated voice quality, faster training time and inference time. The input to this non-auto regressive model is text only. The model is constructed of several layers including: Phoneme Embedding, Encoder, Variance Adaptor, Mel-spectrogram Decoder and Waveform Decoder. The encoder block converts the phoneme embedding sequence into the phoneme hidden sequence which is then passed to the variance adaptor. The variance adaptor has an important role in this architecture since it adds different variance information such as duration, pitch and energy to the hidden sequence. And finally the mel-spectrogram decoder model converts the transformed hidden sequence into mel-spectrogram sequence in parallel.

4 The Architecture

We opted to use Android Native as the platform for implementing our solution and developing an application that allows visually impaired individuals to interact with our tool in an engaging manner. This decision was based on the fact that mobile devices are the most user-friendly IT tools for any population, including the blind, and that Android is the most popular and widely used mobile platform[4] at the moment, implying that it will have a low to no learning curve for anyone.

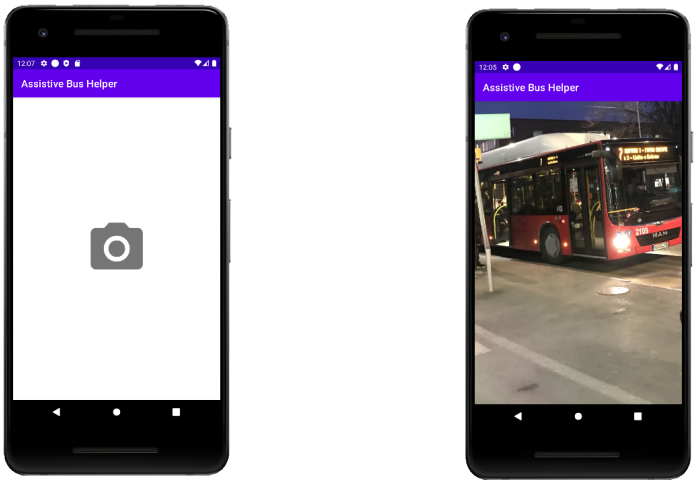
4.1 Android components

The core Android application components, such as Activities, Services, Coroutines, Notifications, Camera[5], and Broadcast Receivers[6], as well as their lifecycle methods and other built-in features, were used to design the whole application flow and user interface. Kotlin was the primary programming language used in the solution. Our application consists of two Android Activities:

- *M*ainActivity (Figure 3a) - system's main screen, which has a very minimal design and consists of the application's title and one camera button that takes you to the *C*ameraActivity. Once the necessary Text-to-Speech and Optical Character Recognition models are loaded, the user can engage with this activity.
- *C*ameraActivity (Figure 3b) - the screen that displays live feed of the device's front camera. The application checks whether the camera is currently collecting a bus number by capturing an image from the camera every 5 seconds, submitting it to the image processing pipeline for inference, and waiting for a result to know how to process the data. If the model accurately

[4] https://gs.statcounter.com/os-market-share/mobile/worldwide
[5] https://developer.android.com/guide/topics/media/camera
[6] https://developer.android.com/guide/components/broadcasts

detects a bus number, it is immediately passed to the Text-to-Speech model, which generates an audio output and plays it on the device's available output speaker.



(a) The Main Activity of the application (b) The Camera Activity of the application

Fig. 3: Screenshots of the application

4.2 Modules integration

Chaquopy[7] is a Python SDK for Android. It includes all of the components needed to run Python programs in an Android Native environment. Chaquopy supports Android Studio's standard Gradle build system, as well as a huge variety of third-party Python packages and simple APIs for accessing Python code from Java/Kotlin and vice versa. Chaquopy is a tremendously powerful tool as a result of all of this, bringing up a whole new world of possibilities for integrating and optimizing Machine Learning and Artificial Intelligence models in Android embedded systems. The Android application calls the scripts by specifying their location and name in the project structure, as well as the precise method that needs to be called, along with the necessary arguments of any suitable type.

Chaquopy was used for loading and inference of the Text-to-Speech model FastSpeech 2 and the Image Processing models such as YOLOX and EasyOCR, which were built and trained using the programming language Python, with the help of various third-party libraries that are compatible with it.

[7] https://chaquo.com/chaquopy/

4.3 Application flow

The entire design and aesthetics of the application were created in such a way that a visually impaired person may use it in the most basic way possible, with no difficulty engaging with it owing to his perceptual limitations.

When the user first launches the application, just one button is available: the camera button. Since it is the only visual element on the screen, the button click event will be triggered if the user clicks anywhere near the middle of the screen. The user is redirected to a screen that displays a live video feed from his camera when he taps that button. Permissions will be required if the program does not have access to the camera on the user's device. An image is captured from this screen every 5 seconds and submitted to the pipeline, consisting of the Object Detector model YOLOX and Optical Character Recognition model EasyOCR for inference and prediction of whether the currently captured scene in front of the user, has a bus line number.

If the image contains a bus line number, it is sent to the FastSpeech 2 Text-to-Speech model for speech synthesis, which is immediately played on the user device's available output speaker, allowing him to hear the number of the bus that is passing next to him. After that, the user is returned to the initial screen, which shows the camera button again. If the image does not contain a bus line number, the preview of the camera's video feed continues until the Optical Character Recognition models predict a number, or the user exits the camera screen and returns to the beginning screen manually.

Internet connection is required only the first time the application launches on the user's device to download all the necessary components and dependencies. All of the processing will take place locally on the device.

Figure 4 illustrates the entire user flow of the application, as well as the flow of activating its components.

The whole source code of the application is published on GitHub[8].

5 Experiments and Results

In this section we will examine the experimental results obtained from using our methodology.

First, the results from the YOLOX model evaluation will be examined, then we will take a look at how expensive such a solution is for a mobile device in terms of battery usage, time to inference, model loading and etc.

Table 3 shows the evaluation results obtained on the test set. The IoU metric, i.e. Intersection Over Union is a widely used object detection evaluation metric. As the name suggests the metric evaluates the degree of overlap between the ground truth and the prediction. The AP, i.e. Average Precision metric computes the average precision value for recall value over 0 to 1, 1 being the perfect model. And the Mean Average Precision (mAP) metric is the average of AP values over all classes.

[8] https://github.com/nikolaStanojkovski/Assistive_Bus_Helper

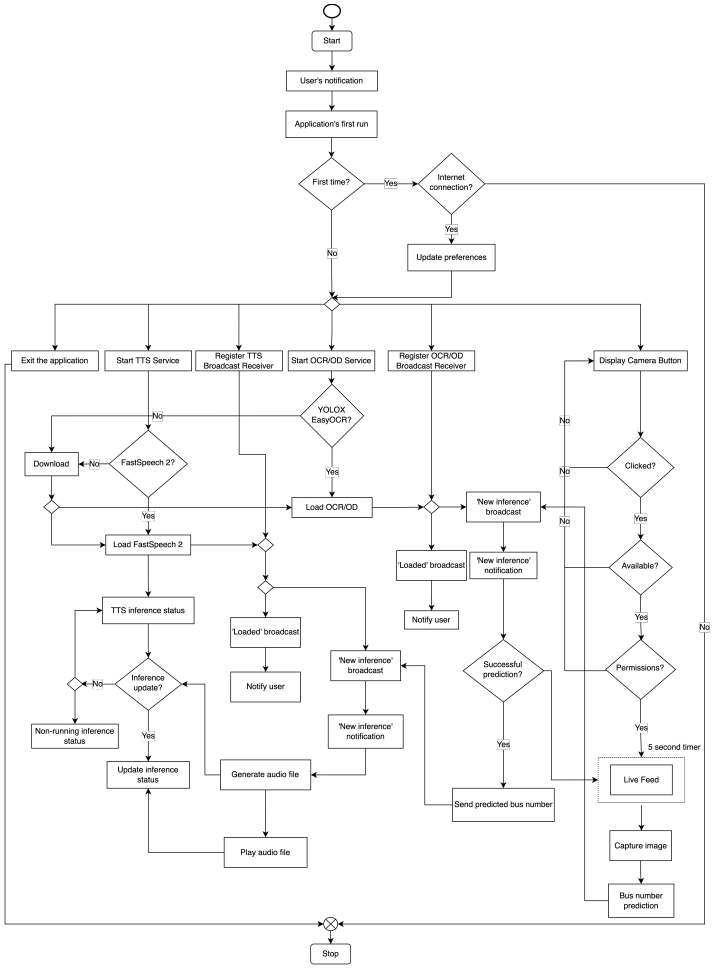


Fig. 4: The entire Android application's flow diagram

The average forward time and average inference time are computed using Tesla T4 GPU in Google Colab and can vary heavily on the type of processing unit as we will see in the next results obtained from an emulator mobile phone.

Table 4 shows the performance of the proposed solution executed on a mobile emulator device. We have done 5 identical experiments to evaluate the performance. As described in the introduction above, we know how much fast inference time is important for this problem. The detection of the bus line number of an incoming bus requires fast inference time on all models, thus the results show that such execution time can be achieved without using any external services nor executing such complex models on Cloud. Rather we packed the whole system on a mobile device with impressive execution times. The average inference time

Table 3: YOLOX evaluation

Metric	**Value**
Eval IoU	0.50
AP for digitalnumber	0.8300
AP for nondigitalnumber	0.5423
Mean AP	0.6862
Average forward time	27.58 ms
Average inference time	28.27 ms

Table 4: Mobile app performance

Metric	**1st try**	**2nd try**	**3rd try**	**4th try**	**5th try**	**Average**
App installation time	1m45s	1m30s	2m	2m10s	1m52s	**1m51.4s**
TTS & OD & OCR Models loading time(first time)	1m15s	1m2s	1m	1m20s	58s	**1m7s**
TTS & OD & OCR Models loading time(every other time)	11s	14.6s	14s	17s	16s	**14.52s**
OD & OCR inference time	0.5s	1s	0.9s	1.2s	1.8s	**1.08s**
TTS inference time	4.2s	4.8s	4s	6s	5.4s	**4.88s**
Final app size	1.02GB					

for the whole system is ∼5s-6s from the moment the camera captured an image, and the announced bus line number.

Such inference times could vary on different processing units used by the various mobile devices, the battery usage is significant and must be considered while using the application.

6 Conclusion and Future Work

In this research we took an initiative not only to implement Deep learning models for detecting bus line number and to announce the detected number in a human-like speech, but we also tackled the challenge of implementing and executing such complex Deep learning based models on a single mobile device. This novel approach could open new research opportunities where Deep learning models could be used in-house rather than be dependent on Internet connection all the time. Therefore, implementing and developing applications that use Deep learning models for facilitating everyday activities of the visually impaired persons could be realized.

In our future work the focus will be on optimizing the solution by working on models destilation and their implementation on a memory-constrained microcontroller that will end-up in sufficiently lower energy consumption. Eliminating the need of Bluetooth or WiFi modules will lead to a hardware architecture ready to be implemented in the form of a smart white cane, without using additional

electronic components that will add on additional weight and thus keeping the comfort of the traditional white canes.

References

1. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character region awareness for text detection (2019). https://doi.org/10.48550/ARXIV.1904.01941, https://arxiv.org/abs/1904.01941
2. Bourbakis, N., Keefer, R., Dakopoulos, D., Esposito, A.: A multimodal interaction scheme between a blind user and the tyflos assistive prototype. In: 2008 20th IEEE International Conference on Tools with Artificial Intelligence. vol. 2, pp. 487–494. IEEE (2008)
3. Bourne, R., Steinmetz, J.D., Flaxman, S., Briant, P.S., Taylor, H.R., Resnikoff, S., Casson, R.J., Abdoli, A., Abu-Gharbieh, E., Afshin, A., et al.: Trends in prevalence of blindness and distance and near vision impairment over 30 years: an analysis for the global burden of disease study. The Lancet global health **9**(2), e130–e143 (2021)
4. Everyday SIGHT: Best White Canes for the Blind, https://www.everydaysight.com/best-white-canes-for-the-blind/
5. Fricke, T.R., Tahhan, N., Resnikoff, S., Papas, E., Burnett, A., Ho, S.M., Naduvilath, T., Naidoo, K.S.: Global prevalence of presbyopia and vision impairment from uncorrected presbyopia: systematic review, meta-analysis, and modelling. Ophthalmology **125**(10), 1492–1499 (2018)
6. Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J.: Yolox: Exceeding yolo series in 2021 (2021). https://doi.org/10.48550/ARXIV.2107.08430, https://arxiv.org/abs/2107.08430
7. Guida, C., Comanducci, D., Colombo, C.: Automatic bus line number localization and recognition on mobile phones—a computer vision aid for the visually impaired. In: International Conference on Image Analysis and Processing. pp. 323–332. Springer (2011)
8. Hill, J., Black, J.: The miniguide: a new electronic travel device. Journal of Visual Impairment & Blindness **97**(10), 1–6 (2003)
9. Jafri, R., Khan, M.M.: User-centered design of a depth data based obstacle detection and avoidance system for the visually impaired. Human-centric Computing and Information Sciences **8**(1), 1–30 (2018)
10. Kim, S.Y., Cho, K.: Usability and design guidelines of smart canes for users with visual impairments. international Journal of Design **7**(1) (2013)
11. Leutenegger, S., Chli, M., Siegwart, R.Y.: Brisk: Binary robust invariant scalable keypoints. In: 2011 International conference on computer vision. pp. 2548–2555. Ieee (2011)
12. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2014). https://doi.org/10.48550/ARXIV.1405.0312, https://arxiv.org/abs/1405.0312
13. Manduchi, R., Kurniawan, S.: Mobility-related accidents experienced by people with visual impairment. AER Journal: Research and Practice in Visual Impairment and Blindness **4**(2), 44–54 (2011)
14. Messaoudi, M.D., Menelas, B.A.J., Mcheick, H.: Autonomous smart white cane navigation system for indoor usage. Technologies **8**(3), 37 (2020)

15. Modukuri, R., Morris, R.J.: Voice based web services–an assistive technology for visually impaired persons. Technology and Disability **16**(4), 195–200 (2004)
16. NSSRM: The National Union of the Blind of the Republic of Macedonia, https://nssrm.org.mk/za-nas/
17. Organization, W.H., et al.: Visual impairment and blindness fact sheet n 282. Retrieved June **21**, 2016 (2014)
18. Pardasani, A., Indi, P.N., Banerjee, S., Kamal, A., Garg, V.: Smart assistive navigation devices for visually impaired people. In: 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS). pp. 725–729. IEEE (2019)
19. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018). https://doi.org/10.48550/ARXIV.1804.02767, https://arxiv.org/abs/1804.02767
20. Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., Liu, T.Y.: Fastspeech 2: Fast and high-quality end-to-end text to speech (2020). https://doi.org/10.48550/ARXIV.2006.04558, https://arxiv.org/abs/2006.04558
21. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition (2015). https://doi.org/10.48550/ARXIV.1507.05717, https://arxiv.org/abs/1507.05717
22. Shilkrot, R., Huber, J., Meng Ee, W., Maes, P., Nanayakkara, S.C.: Fingerreader: a wearable device to explore printed text on the go. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. pp. 2363–2372 (2015)
23. Sivan, S., Darsan, G.: Computer vision based assistive technology for blind and visually impaired people. In: Proceedings of the 7th International Conference on Computing Communication and Networking Technologies. pp. 1–8 (2016)
24. Smith, R.: An overview of the tesseract ocr engine. In: Ninth international conference on document analysis and recognition (ICDAR 2007). vol. 2, pp. 629–633. IEEE (2007)
25. Steinmetz, J.D., Bourne, R.R., Briant, P.S., Flaxman, S.R., Taylor, H.R., Jonas, J.B., Abdoli, A.A., Abrha, W.A., Abualhasan, A., Abu-Gharbieh, E.G., et al.: Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to vision 2020: the right to sight: an analysis for the global burden of disease study. The Lancet Global Health **9**(2), e144–e160 (2021)
26. Subbiah, S., Ramya, S., Krishna, G.P., Nayagam, S.: Smart cane for visually impaired based on iot. In: 2019 3rd International Conference on Computing and Communications Technologies (ICCCT). pp. 50–53. IEEE (2019)
27. WeWalk: WeWalk, https://wewalk.io
28. Withington, D.J., Waters, D.A., Povey, M.J.W., Hoyle, B.S.: Spatial awareness device (Mar 23 2004), uS Patent 6,710,706