SceneRecog: A Deep Learning Scene Recognition Model for Assisting Blind and Visually Impaired Navigate using Smartphones

Bineeth Kuriakose[1], Raju Shrestha[2] and Frode Eika Sandnes[3]

***Abstract*— Deep learning models have recently gained popularity in the research community due to their high classification success rates. In this paper, we proposed an EfficientNet-Lite based scene recognition model for scene recognition as a part of the smartphone-based navigation support application for the blind and visually impaired. We created a custom dataset with both indoor and outdoor scenes for training and testing of the model. The main objective of this work is to support people with visual impairments navigate by providing information about the scene via a smartphone application. The results from the experiment show encouraging performance from the proposed model. As a proof of concept, a prototype app was developed on the Android platform. However, the model can be implemented and deployed in any modern smartphone with good processing power.**

***Index Terms*— scene recognition; visual impairment; navigation; deep learning; smartphone; assistive technology.**

I. INTRODUCTION

A scene is a representation of incidents, actions, events, or a landscape that consists of numerous objects. Humans can recognize and classify scenes using their visual senses. For example, we can recognize visual scenes such as the kitchen or bathroom by analyzing the environment and objects present in their field of view. This scene recognition capability helps us to avoid danger such as a 'fire' in the vicinity. However, this could be challenging for people with a vision problem, such as blindness and other forms of visual impairment (VI).

Scene recognition is a fundamental part when describing visual scenery. An overall recognition of a scene contrasts with a simple listing of the visible objects in the scene. Scene recognition concerns the existence of objects and the semantic relations between them and the contextual information concerning the background [1]. Automatic scene classification is an active avenue of research within computer vision, which involves assigning a label to an image presented as input based on its overall contents. Visual scene recognition and understanding can be achieved either using static input images or dynamic input videos. The application of scene understanding is used in diversified domains in computer vision and artificial intelligence, such as in autonomous vehicle navigation [2], robot navigation [3], and mobile photography [4]. Scene recognition is also considered a prerequisite for other advanced computer vision tasks such as image retrieval and object detection [1].

There are mainly two key challenges associated with scene recognition that concern the nature of the images depicting a scene context [5]. The first is related to the visual inconsistency or low interclass variance. Images belonging to different classes may be confused with each other. We may end up with class overlap. The second challenge relates to annotation ambiguity or intraclass variance. The distinction of scene categories is a subjective process that is highly dependent on the experience of the annotators. Therefore, images from the same class can exhibit significantly different appearances. One method for dealing with the visual inconsistency problem [5] is to use a multiresolution convolutional neural network (CNN) framework. In multiresolution CNNs, two levels of resolution are implemented. At a coarse resolution level, global structures or large-scale objects are captured. And at a fine resolution level, the detailed local information of fine-scale objects is captured. Annotation ambiguity can be handled by merging similar categories into a supercategory after learning the correlation between different classes.

According to the World Health Organization, it is estimated that at least 2.2 thousand million people worldwide have vision impairment or blindness[1]. Technological solutions that focus to help and support people in tasks such as navigation, reading, etc. are generally called as *Assistive Technology*. A navigation system integrated with a scene recognition module could support people with visual impairments when there is an emergency such as 'fire' or 'heavy rainfall'. Besides, portability is one of the vital requirements in navigation systems for people with visual impairments [6]. When prioritizing portability, smartphones are becoming a viable technological platform for a navigation support solution [7].

In this work, we propose a deep learning-based scene recognition model as a part of the smartphone-based navigation support application. The model is based on the EfficientNet-Lite from the family of EfficientNet [8], which is currently one of the most accurate models that are also easily deployable on smartphones. To our knowledge, this is the first attempt to use the EfficientNet-Lite model for scene recognition in a navigation support application. A custom dataset was created to train and test the model. In addition,

[1]Bineeth Kuriakose is with the Department of Computer Science, Oslo Metropolitan University (OsloMet), Oslo, Norway. Corresponding author: bineethk@oslomet.no

[2]Raju Shrestha is with the Department of Computer Science, Oslo Metropolitan University (OsloMet), Oslo, Norway. raju.shrestha@oslomet.no

[3]Frode Eika Sandnes is with the Department of Computer Science, Oslo Metropolitan University (OsloMet), Oslo, Norway. frodes@oslomet.no

[1]https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment

 DOI: https://doi.org/10.1109/SMC52423.2021.9658913

a prototype app is developed to demonstrate a portable and real-time scene recognition solution based on the proposed model, which outputs both text and audio output from the recognized scenes.

The paper is organized as follows: Section 2 gives a brief review of navigation systems proposed for people with visual impairments and developments in scene recognition. Section 3 describes the EfficientNet-Lite model, which is used as the base model in the proposed system. Section 4 presents the proposed system, including the details about the dataset and the prototype app. Section 5 describes the experiments and presents the results. The results are discussed in Section 6. Section 7 concludes the paper.

II. RELATED WORKS

A. Navigation System for people with visual impairments

Several navigation support systems for people with visual impairments have been proposed in the literature. NavCog is a mobile navigation app which uses low energy Bluetooth beacons to give indoor navigation directions to users [9]. PERCEPT is another indoor navigation mobile app where the user can obtain navigation directions to the destination when touching specific landmarks tagged with Near Field Communication tags [10]. Another notable system with similar indoor navigation functionality is GuideBeacon [11]. The GuideBeacon is also a smartphone application that uses Bluetooth beacons deployed in the indoor space to navigate surroundings. The main limitations associated with all these systems are they need an infrastructure or additional hardware such as beacons and NFC tags for wayfinding or navigation.

Researchers also explored the application of computer vision and deep learning for the design of navigation systems for the visually impaired. Systems such as [12]–[14] used various deep learning-based object recognition systems to help the user navigate in unfamiliar environments. But some of these systems requires a data connectivity to do operations which are executed in a cloud environment.

Common to most of the current navigation support systems is that they focused on how a user can avoid obstacles and reach a destination without informing the user about the environment context. Hence, it is relevant to include the scene recognition feature in a navigation system since it can help the users to be more aware about the environment. It can also support users to traverse different environments using different strategies and ways of moving. Thus, a system with a scene recognition module can support people with visual impairments during navigation, both indoor and outdoor, to learn about the surroundings and help the user make safe navigation decisions.

B. Scene Recognition

Early scene image representation and recognition mainly relied on global attribute descriptors, which are formed by some low-level visual properties to model the perception of human beings. CENsus TRansform hISTogram (CENTRIST) [15] and Local Difference Binary Pattern (LDBP) [16] are some examples of global attribute descriptors. The performance of these global attribute descriptors is limited by the complex visual constitutions of scene images [1].

Later, the patch feature encoding was introduced to improve the recognition performance. Examples of local visual descriptors that have been widely used in the patch feature extraction include Local Binary Patterns (LBP) [17], Scale Invariant Feature Transform (SIFT) [18], Speeded Up Robust Features (SURF) [19], Histogram of Oriented Gradients (HOG) [20], and Oriented Texture Curves (OTC) [21]. Limitations of local visual descriptors includes their high dimensionality and computational effort. Moreover, they exhibit poor performance when numerous similar local features and complex background exist in the matching image. The Bag-of-Visual-Words (BoVW) framework was introduced to integrate many local visual descriptors into an image representation [22]. It was one of the most required feature transformation methods that were extensively used for image classification.

Recent developments in deep learning and deep convolutional neural networks, their high classification performance, and the significant number of annotated datasets have sparked a substantial interest in addressing image classification and scene recognition. Scene-centered datasets are created, such as the MIT Indoor67[2], SUN397[3], and Places365[4]. Several works on scene recognition have been reported. Zhou Bolei et al. reported one well-known work on scene recognition which used the Places365 dataset [23]. The authors trained three popular convolutional neural network (CNN) architectures AlexNet [24], GoogLeNet [25], and VGG16 [26].

Recently, research attention has been drawn towards scene recognition from moving videos. CNNs, which have shown promising results for the general task of scene recognition in images, can also be generalized to video data. The T-ResNet architecture alongside the YUP++ dataset, established a new benchmark in the subfield of dynamic scene recognition [27]. Even though the T-ResNet model showed strong performance for classes with linear motion patterns such as ‘elevator’, ‘ocean’, the performance was negatively impacted for scene categories that have irregular or mixed defining motion patterns such as ‘snowing’ and ‘fireworks’. Moreover, the model exhibits performance degradation when the camera is in motion.

Due to the proliferation and increasing computational power of mobile and smartphone devices, much research on deep learning focused on applications using these devices. At present, there exist several deep learning architectures or models for mobile devices, called *lite* models. MobileNet [28] is an example of mobile architecture that is used to build lightweight deep neural networks. Moreover, TensorFlow Lite provides an option to convert and run TensorFlow models on mobile, embedded, and IoT devices. Many of the existing deep learning models such as MobileNet [28],

[2]http://web.mit.edu/torralba/www/indoor.html
[3]https://vision.princeton.edu/projects/2010/SUN/
[4]http://places2.csail.mit.edu/

YOLO [29], EfficientNet [8], and Inception [30] can be converted into the *lite* format with some compromise in accuracy but with better inference time and deployed in mobile devices using *TensorFlow Lite Converters*[5].

The next section briefly describes the EfficientNet lite model.

III. EFFICIENTNET-LITE MODEL

EfficientNet is a convolutional neural network (CNN) introduced by Google, which achieves state-of-the-art accuracy with lesser computations and parameters than similar models [8]. A convolutional neural network can be scaled in three dimensions: *depth*, *width*, and *resolution*. The *depth* of the network relates to the number of layers in the network. The *width* is related to the number of filters in a convolutional layer. And the *resolution* is associated with the height and width of the input image. EfficientNet uses a compound coefficient ϕ to scale the network width, depth, and resolution in a more structured and uniform manner [8].

EfficientNet provides an optimized version called EfficientNet-Lite that is designed for mobile CPU, GPU, and EdgeTPU and runs on TensorFlow Lite. Due to the unique nature of mobile and edge devices, several challenges are raised when a CNN model is deployed. Since many CNN models have limited floating-point support, quantization is widely used to overcome the limitation when the same model is used in mobile/edge devices. However, a complicated quantization-aware training procedure is required in most cases. Besides, the model accuracy could be comprised after the posttraining quantization. Using the TensorFlow Model Optimization Toolkit[6], the model can be quantized easily via integer-only post-training quantization without losing much accuracy.

Hardware heterogeneity is one of the challenging issues when the same model is run on an extensive variety of accelerators, such as mobile GPU or EdgeTPU. Due to the hardware specialization, these accelerators often perform well for only a limited set of operations. This limitation is crucial for EfficientNet since some of the operations are not well supported by specific accelerators. To overcome this limitation, the original EfficientNet is tailored with some modifications in EfficientNet Lite model. The squeeze-and-excitation blocks are removed, and ReLU6 activation functions are introduced in the lite version, which claims to improve the quality of posttraining quantization[7].

EfficientNet-Lite comes in five variants, allowing users to select from the low latency/model size option, the EfficientNet-Lite0, to the high accuracy option, the EfficientNet-Lite4[7]. Fig. 1. shows how EfficientNet-Lite models perform compared to some popular image classification models in terms of accuracy, latency, and model size.

[5]https://www.tensorflow.org/lite/convert

[6]https://www.tensorflow.org/model_optimization/guide/get_started

[7]https://blog.tensorflow.org/2020/03/higher-accuracy-on-vision-models-with-efficientnet-lite.html

Even though several research have been reported in the scene recognition domain, our work mainly focuses on how a model can be trained to be deployed on smartphone devices with limited memory and computational power, thereby requiring a comparatively small model size and low computational demand. Taking these factors into account, we have chosen EfficientNet Lite4 architecture as the base model in our proposed system. The selection of the EfficientNet Lite4 is due to the following considerations.

- The EfficientNet Lite4 model has more *Top 1* accuracy compared to the other models (see Fig. 1). *Top 1* accuracy is the accuracy where the true class matches the most probable class predicted by the model, which is the same as our standard accuracy. This is an essential factor to be considered in our application domain. When there is a trade-off between accuracy and latency, we are concerned and more interested in accuracy.
- Even though this model has a comparatively large model size, that is not a constraint for our problem domain. Our primary concern is the model being optimized for mobile deployment.
- The low inference time is another important criterion for our problem domain for real-time scene recognition.

IV. PROPOSED SYSTEM

This section gives an overview of the proposed model for scene recognition. Furthermore, the section provides a brief description of the custom dataset we created for training and testing the model. The prototype app developed based on the model is also described.

A. The proposed scene recognition model

We used the EfficientNet-Lite4 model and employed a transfer learning technique to adapt to our scene recognition use case and accelerate the learning process. Transfer learning allows us to deal with these scenarios by leveraging the already existing models of some related task or domain. The knowledge gained from the first task applies to a different problem domain, such as scene recognition. TensorFlow offers *TensorFlow Lite Model Maker*[8] for *lite* models to apply transfer learning on custom datasets and export the resulting model to a TensorFlow Lite format. A classifier head is added to the pre-trained model of EfficientNetLite4 using the *TensorFlow Lite Model Maker*. An overview of the model architecture is shown in Fig.2.

B. Dataset

As a first step and as a proof of concept for an initial prototype, we selected 15 scene categories (classes) that can occur in indoor and outdoor environments and have relevance in the navigation scenario of people with visual impairments. They are: *Bathroom, Bedroom, Bridge, Cafeteria, Classroom, Computer room, Dining hall, Hospital room, Kitchen, Library indoor, Parking lot, Playground, River, Shopfront, and Supermarket*.

[8]https://www.tensorflow.org/lite/guide/model_maker

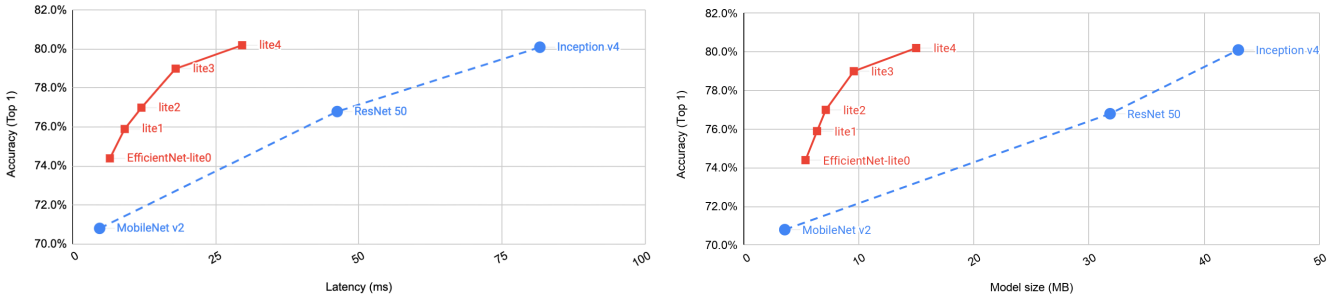

Fig. 1. Comparison of EfficientNet lite versions and 3 other popular deep neural network models: MobileNet v2, ResNet 50 and Inception v4 in terms of (a) accuracy vs latency, and (b) accuracy vs model size. (inspired from TensorFlow Blog[6])

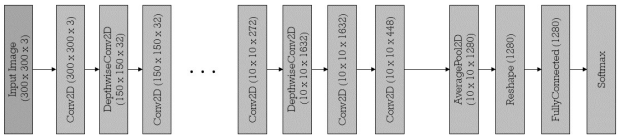

Fig. 2. Architecture of EfficientNet-Lite4 model

The dataset is created by collecting images collected from four different sources, MIT's Places365[4], Google Open Images V6[9], Flicker[10], and our own images. Places365 is one of the well-known datasets used in the scene recognition domain. The Google Open Images V6 dataset is mainly used for object detection or segmentation-related research and not for scene recognition. However, it contains relevant images for our use case. Flicker contains a large number of images, and it has API[11] support to download images and given permission to use them for non-commercial purposes. Relevant images for the 15 scene categories are extracted from these three data sources. After examining the extracted images, we found that many of the images were disordered, and some pre-processing such as re-labeling and image removal of irrelevant and irregular were needed. Moreover, additional images from the own collection are added in order to have some real images from the locality to increase the dataset and, in turn, improve the performance of the model. Around 600 images were captured from different scenes such as 'supermarket', 'storefront', 'river', etc., from Oslo. Altogether, the dataset consisting of approximately 5000 images per category is created and used for training and testing of the proposed model. Details of the training and testing process are described below in section 5.

C. Prototype App

A prototype Android app is developed using the trained model, converted to *tflite* (TensorFlowLite) format and deployed in a smartphone. The app can capture and recognize scenes belonging to the 15 scene categories in real-time. The app displays the recognized category as a text. It also gives audio output with synthetic voice when the user touches the screen. This feature of one-touch-based output is enabled to avoid random continuous voice prompts, which can disturb the user during navigation. Fig. 3. shows a screenshot of the prototype app in action, which recognized the scene as 'kitchen'. The app shows the prediction based on the highest probability.

To minimize false positives with unknown scenes, we incorporated a threshold. When the probability of a scene recognized by the model is above the threshold and holds the highest probability among other scene categories, the app will report the recognition result. Otherwise, it will output as 'unknown scene'. Based on trial and error, the threshold was set to 60%. This helps filter out the scenes other than the 15 categories used in our test use case as 'unknown scene.'

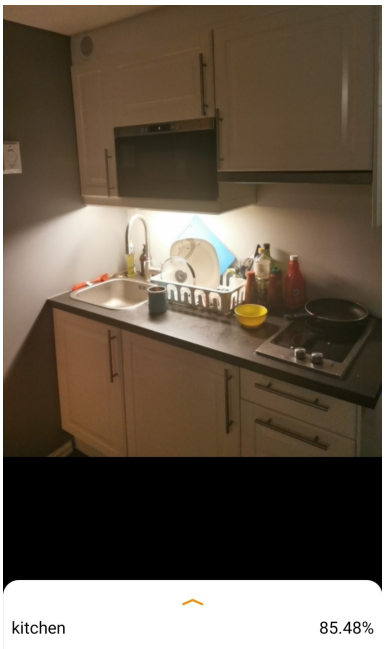

Fig. 3. A screenshot of the real-time scene recognition from the prototype

V. EXPERIMENT AND RESULTS

This section gives the details of the experiment and the results which we got from our model.

A. Experimental Setup

The proposed model is implemented and runs an HP Omen Intel core i7 processor with 32GB RAM and NVIDIA

[9]https://storage.googleapis.com/openimages/web/index.html
[10]https://www.flickr.com/
[11]https://www.flickr.com/services/developer/api/

GeForce GTX 1070 GPU. The platform settings of the experiment are TensorFlow-GPU 2.4.1, NVIDIA CUDA toolkit 11.0, and CUDNN 8.1. The model is trained, validated, and tested by randomly shuffling and splitting the dataset in the ratio of 80:10:10, respectively.

The training hyperparameter settings used in the experiment are as follows: To avoid the generalization issue with the larger batch size and the convergence issue with the smaller batch size, we decided to go with the default batch size of 32. The learning rate was 0.2 that was the default rate provided in the *TensorFlow Lite Model Maker*. At the end of 10 epochs, when the validation loss began to rise, we stopped training. Finally, the training accuracy of the model was at 93.43%, and validation accuracy was at 83.12%.

B. Results

To assess the performance of our scene recognition model, we used various evaluation metrics. The model was evaluated for *precision, recall, F1 score,* and *accuracy* on all scene categories. The different performance metrics results are given in Fig. 4.

Selected results of some test images for all 15 scene categories that are correctly and incorrectly classified by the model are shown in Fig. 5. The incorrect results are given in the figure to analyze which category is predicted by the model instead of its actual category.

The proposed model is compared with a state-of-the-art model *Inception v4 Lite* after training with a similar environment using the same dataset. The results listed in Table I. show that our model exhibits higher accuracy, precision, and recall. The Inceptionv4 Lite model size is also larger compared to our model, which makes our model more suitable to deploy on a smartphone device.

TABLE I
PERFORMANCE COMPARISON OF INCEPTION V4 LITE AND EFFICIENTNET LITE4 ON THE TEST DATASET

Model	Params	Accuracy	Precision	Recall	F1 Score	Model Size
Inception v4 Lite	23M	82.92	0.82	0.73	0.83	51MB
SceneRecog (our model)	11M	83.33	0.85	0.80	0.81	46MB

VI. DISCUSSIONS

The results show that the proposed model exhibits good performance in scene recognition even when deployed in a smartphone with less computational power compared to a PC or laptop. But at the same time, there exist some misclassifications associated with the proposed scene recognition lite model, as seen from the results. The three possible scenarios are discussed below.

(a) Misclassification due to labeling ambiguity: As seen in Fig. 5., there are various occurrences where the model fails due to the misclassification. The misclassification may have occurred when there were two different categories in a similar scene. For example, a 'river' and 'bridge' can appear in the same scene. During the labeling stage of a dataset, it is the annotator's decision to categorize the images into any of those categories. Therefore, there is a chance of misinterpretation based on the annotator's decision during scene labeling. This issue could be resolved by giving an intimation to the various scene categories presence during the labeling stage itself by the annotator. Since the primary aim of this research is to demonstrate how scene recognition is possible in a smartphone device using EfficientNetLite4, the focus is more given to the proof of concept. However, in future work, more effort will be given to solve the labeling ambiguity issues while creating the dataset.

(b) Misclassification due to the similarity between images: This case occurs when the model fails to recognize a scene with another one that shares common features. The similarity between scenes makes it challenging for the trained model to differentiate successfully. For example, consider the case between a 'cafeteria' and a 'dining hall'. The model could learn the presence of furniture and other food items in the scene to categorize it into any of those scene categories. Because of the similarity between the two scenes, there could also be a potential chance of misclassification reported from the model. A similar issue may occur in other pairs of scenes such as 'playground' – 'parking lot', 'shopfront' – 'supermarket', etc. This issue is due to the comparatively poor accuracy of a lite model deployed in a smartphone. When a highly accurate general model with a larger model size is used, this issue should be solved. However, we used a lite model with the constraint of accuracy to demonstrate how a smartphone device could be used for scene recognition.

(c) Overlap between two categories in the same scene: This case occurs when there is an overlap between two scenes with the same input image. The model can predict incorrect results in such a situation. This is similar to the first case, but here, the accuracy of the model is prominent. For example, consider the case of 'river'- 'bridge' or 'hospital room'- 'bedroom'. In each of those cases, there might be chances when both scenes at present together. One of the solutions that can be employed here is the use of multilabel classification on the same image. That is, categorizing the same image into two different classes by giving equal weight to both. The accuracy of the model can only be increased in such a situation by training with more images with multilabel.

Fig. 4. shows that two of the scene categories, 'cafeteria', and 'computer room', achieves less than 70% recognition accuracy. At the same time, five scene categories have an accuracy of more than 90%. The confusion matrix reveals that there was a high misclassification between the 'dining hall' and 'cafeteria'. The model founds it challenging to categorize both classes—the same holds for combinations of the 'parking lot' – 'bridge' and 'river' – 'bridge'.

The proposed scene recognition model is implemented by considering both the processing and power limitations of a portable mobile device. There might be models which yield better scene recognition performance with high computational requirements, such as general EfficientNet and Inceptionv4 architectures. However, according to our problem domain, the model needs to be used in real-time and should use computation and memory resources sparingly. The model

Predicted Categories \ True Categories	Bathroom	Bedroom	Bridge	Cafeteria	Classroom	Computer Room	Dining Hall	Hospital Room	Kitchen	Library Indoor	Parking Lot	Playground	River	Shopfront	Supermarket	ACCURACY
Bathroom	455	1		2	4		4	4	27			1	1	1		91.00
Bedroom	5	450			3	1	10	7	11	8				1	4	90.00
Bridge			443	1			1	2		2	6	8	35		2	88.60
Cafeteria	1			269	28		216		8	33	1	2	2	6	11	59.30
Classroom	1			17	438		21	5	3	14		1				87.60
Computer Room	3	3	1	4	90	338	5	1	29	25					1	69.88
Dining Hall				78	20		372	1	7	20		2				75.40
Hospital Room	6	21	1	5	33	1	9	373	26	5	4	6	1	6	3	75.60
Kitchen	3			10	4	1	19	3	447	10					3	89.40
Library Indoor		1		15	38		31	2	9	399				2	3	79.80
Parking Lot			170	6	3			17	1	3	417	16	9	11		83.40
Playground			9	3	2		1	1	1	2	1	473	3	1	3	94.60
River			101	3			1	3			3	1	388			77.60
Shopfront				4			1	5	4	2	1			477	6	95.40
Supermarket				9	5		1		2	7	1	1		10	462	92.40
Precision	0.97	0.95	0.78	0.67	0.69	0.99	0.58	0.92	0.80	0.80	0.96	0.93	0.90	0.94	0.94	
Recall	0.90	0.87	0.86	0.37	0.83	0.65	0.67	0.78	0.88	0.77	0.82	0.93	0.74	0.94	0.96	83.33
F1 Score	0.93	0.91	0.82	0.47	0.76	0.78	0.62	0.80	0.84	0.79	0.88	0.93	0.81	0.94	0.93	

Fig. 4. Confusion matrix of the scene recognition results from the SceneRecog model. The figure also shows the Precision, Recall, F1 Scores, and accuracy metrics values of each class.

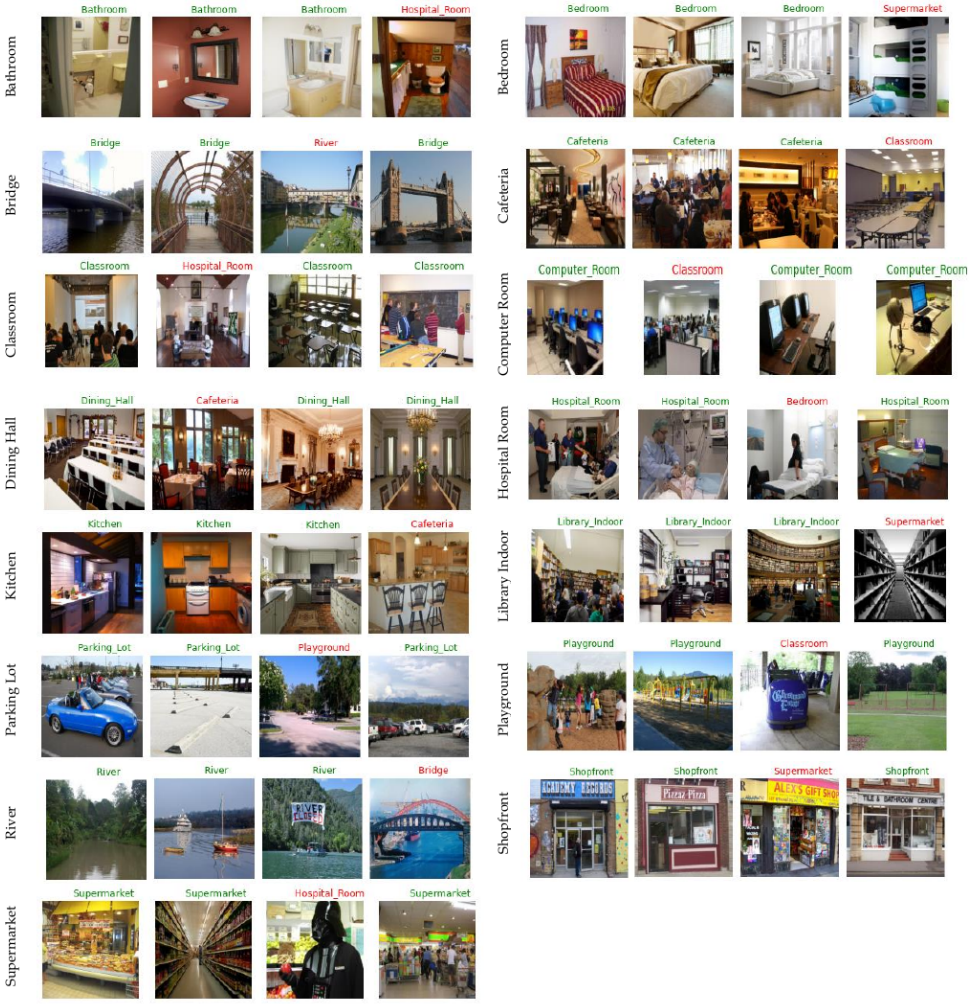

Fig. 5. Example prediction results from the model (red: incorrect, green: correct).

size and accuracy trade-offs are also some critical design decisions in a smartphone-based application. In this scenario, EfficientNet Lite provides reasonably good performance in summing up all those factors contributing to its overall performance. The work presented in this paper is only a proof of concept. The model can be improved by increasing the scene categories and training the model with more data to incorporate it into a full-fledged application. We think that the result is encouraging, and the proposed model hence provides a pragmatic solution for providing scene information in a navigation support application for people with visual impairments.

VII. CONCLUSIONS

Developing robust and reliable models for the automatic recognition of scenes is vital in intelligent systems and artificial intelligence since it directly supports real-life applications. We have shown how an EfficientNetLite4 model can be used for scene recognition and deployed in a smartphone through this work. The solution outputs the recognized scene using synthetic speech and can help the users with visual impairments to get an overview of their environment during navigation. The results achieved from the trained model using the custom dataset demonstrate its potential for navigation support systems. The system can be further enhanced through training using larger datasets with more classes and images.

REFERENCES

[1] L. Xie, F. Lee, L. Liu, K. Kotani, and Q. Chen, "Scene recognition: A comprehensive survey," Pattern Recognition, vol. 102, p. 107205, 2020.

[2] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, "Deep learning-based image recognition for autonomous driving," IATSS Research, vol. 43, no. 4. Elsevier B.V., pp. 244–252, Dec. 01, 2019.

[3] P. Espinace, T. Kollar, N. Roy, and A. Soto, "Indoor scene recognition by a mobile robot through adaptive object detection," Robotics and Autonomous Systems, vol. 61, no. 9, pp. 932–947, Sep. 2013.

[4] J. Li and Y. Qian, "Automatic scene recognition for digital camera by semantic features," in Proceedings of the 2008 International Conference on Wavelet Analysis and Pattern Recognition, ICWAPR, 2008, vol. 1, pp. 327–332.

[5] L. Wang, S. Guo, W. Huang, Y. Xiong, and Y. Qiao, "Knowledge Guided Disambiguation for Large-Scale Scene Classification with Multi-Resolution CNNs," IEEE Transactions on Image Processing, vol. 26, no. 4, pp. 2055–2068, Apr. 2017.

[6] B. Kuriakose, R. Shrestha, and F. E. Sandnes, "Tools and Technologies for Blind and Visually Impaired Navigation Support: A Review," IETE Technical Review, pp. 1–16, Sep. 2020.

[7] B. Kuriakose, R. Shrestha, and F. E. Sandnes, "Smartphone navigation support for blind and visually impaired people - a comprehensive analysis of potentials and opportunities," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Jul. 2020, vol. 12189 LNCS, pp. 568–583.

[8] M. Tan and Q. v. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in 36th International Conference on Machine Learning, May 2019, vol. 2019-June, pp. 6105–6114.

[9] D. Ahmetovic, C. Gleason, C. Ruan, K. Kitani, H. Takagi, and C. Asakawa, "NavCog: a navigational cognitive assistant for the blind." Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services. 2016.

[10] A. Ganz, J. M. Schafer, Y. Tao, C. Wilson and M. Robertson, "PERCEPT-II: Smartphone based indoor navigation system for the blind," 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2014, pp. 3662-3665.

[11] S. A. Cheraghi, V. Namboodiri and L. Walker, "GuideBeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented," 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom), 2017, pp. 121-130.

[12] J. Bai, D. Liu, G. Su, and Z. Fu. 2017. A Cloud and Vision-based Navigation System Used for Blind People. In Proceedings of the 2017 International Conference on Artificial Intelligence, Automation and Control Technologies (AIACT '17). Association for Computing Machinery, New York, NY, USA, Article 22, 1–6.

[13] B. Kim, H. Seo, and J.D. Kim. "Design and implementation of a wearable device for the blind by using deep learning based object recognition." Advances in Computer Science and Ubiquitous Computing. Springer, Singapore, 2017. 1008-1013.

[14] M. Poggi, and S. Mattoccia. "A wearable mobility aid for the visually impaired based on embedded 3D vision and deep learning." 2016 IEEE Symposium on Computers and Communication (ISCC). IEEE, 2016.

[15] J. Wu, J. R.-I. transactions on pattern analysis and, and undefined 2010, "Centrist: A visual descriptor for scene categorization," IEEE transactions on pattern analysis and machine intelligence, vol. 33, no. 8, pp. 1489–1501, 2010.

[16] X. Meng, Z. Wang, L. W.-P. recognition, and undefined 2012, "Building global image features for scene recognition," Pattern recognition, vol. 45, no. 1, pp. 373–380, 2012.

[17] T. Ojala, M. Pietikäinen, D. H.-P. recognition, and undefined 1996, "A comparative study of texture measures with classification based on featured distributions," Pattern recognition, vol. 29, no. 1, pp. 51–59, 1996.

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[19] H. Bay, T. Tuytelaars, and L. van Gool, "SURF: Speeded up robust features," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2006, vol. 3951 LNCS, pp. 404–417.

[20] N. Dalal and T. Bill, "Histograms of oriented gradients for human detection," in IEEE computer society conference on computer vision and pattern recognition (CVPR'05), 2005, pp. 886–893.

[21] R. Margolin, L. Zelnik-Manor, and A. Tal, "OTC: A novel local descriptor for scene classification," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, vol. 8695 LNCS, no. PART 7, pp. 377–391.

[22] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints," in Workshop on statistical learning in computer vision, ECCV, 2004, pp. 1–2.

[23] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning Deep Features for Scene Recognition using Places Database," Advances in Neural Information Processing Systems (NIPS), vol. 27, 2014.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, Jun. 2017.

[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich., "Going Deeper with Convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2015, Accessed: Mar. 02, 2021. [Online]. Available: https://arxiv.org/abs/1409.1556.

[27] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Temporal residual networks for dynamic scene recognition," in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Nov. 2017, vol. 2017-January, pp. 7435–7444.

[28] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." Accessed: Mar. 23, 2021. [Online]. Available: https://arxiv.org/abs/1704.04861.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[30] C. Szegedy, V. Vanhoucke, S. Ioffe, and J. Shlens, "Rethinking the Inception Architecture for Computer Vision," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.