6th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT 2024)
02-04 May 2024, Military Institute of Science Technology (MIST), Dhaka-1216, Bangladesh

BrailleSense: Deep Learning for Braille Character Classification

Abstract— **Individuals with visual impairments face challenges accessing written information. Braille is one of many solutions that only requires the reader to sense the depth of the paper with their hand to comprehend written information without the need to look at the text. However, learning Braille, especially for those losing sight later in life, presents difficulties. This research introduces the BrailleSense system, a technological solution designed to assist visually impaired individuals in learning and utilizing the Braille system effectively. The system features a virtual prototype of hand gloves equipped with a camera, aiming to alleviate challenges associated with Braille pattern memorization. Key contributions include the development of a custom Convolutional Neural Network (CNN) model for Braille pattern classification coined as the BrailleNet. This model is then deployed on a Raspberry Pi to investigate the feasibility of working with resource-limited portable devices, BrailleNet achieves an impressive accuracy of 97.44% under real-world constraints. The research outlines the conceptual design through a 3D model of the gloves, addressing spatial allocation. Acknowledging challenges in user comfort and alignment, BrailleSense presents a pioneering step towards empowering visually impaired individuals, enhancing literacy, and fostering independence. The dataset and code for the BrailleSense system are available on GitHub - `https://github.com/AnasIshfaque/Braille-CNN`**

Convolutional Neural Networks; Braille Character Classification; Deep Learning;

I. INTRODUCTION

People with visual impairment lack the ability to read by looking at the text materials. This has led to the invention of the Braille system which allows such people to read without the necessity to look at any text. Rather than depending on the eyes to perceive the reading content, this system utilizes the sensation of the depth of the paper surface with our fingers to mentally visualize certain patterns. These patterns are the encoded version of the different letters, punctuation, and numbers. Thus, the Braille system represents our traditional communication symbols in the form of patterns that consist of six dots in a tactile format. [1] Each dot can either be raised or lowered to form unique patterns. However, since most people who are visually impaired lose their sight after reaching a certain level of maturity, learning the Braille system can feel similar to learning a new language. This process of learning can be challenging especially if the visually impaired person is over 60 years old as our natural learning instinct degrades with age. Many people are discouraged by the sheer complexity of the various grades of the Braille system and the fact that reading only few sentence can take significant time [2]. Consequently, they may not intend to try to learn a new set of patterns just to be able to read again. Therefore, having a tool that can aid them in learning or using this system can uplift their motivation to start reading once again. Our system is an effort to show a virtual prototype of hand gloves that visually impaired people can wear while reading Braille texts. The extra effort of memorizing the patterns and their corresponding mapping with the traditional symbols can be overlooked as our system can classify the Braille patterns and let the user know which symbol is being focused by the index finger's camera. [3] While this prototype may provide an initial design, it is by no means a market-ready product. Factors that affect the convenience of the user for long-time wearing like the weight of the gloves, portability, and the ease of adapting to correctly focusing the index finger camera to the pattern aligning with the horizontal line which the series of Braille pattern follows are beyond the scope of this preliminary work. Nonetheless, it can act as an initial design upon which a more robust and user-friendly gadget might be developed. Our main contributions are:

Trained a custom CNN model for Braille pattern classification: The developed CNN architecture, known as BrailleNet, represents a crucial component of our system. It has been meticulously designed and trained to proficiently classify Braille patterns. By leveraging techniques like dropout and early stopping, we aimed to address prevalent challenges in training machine learning model, particularly overfitting.

Deployed the classifier in Raspberry Pi: Recognizing the need for a wearable and portable solution, our system's custom CNN model, BrailleNet, was successfully deployed on a Raspberry Pi. This step showcases the practicality and feasibility of developing a hardware-agnostic system that can seamlessly integrate into gloves. It's worth mentioning that future iterations may use RaspiCam for real-time Braille pattern inference and a small speaker for audio output.

Created a 3D model of the gloves: As a proof of concept, we designed a simplified 3D model illustrating the spatial arrangement of key components within the gloves. This includes the placement of the processing unit, camera, and speaker. While this serves as an initial prototype, factors like weight, portability, and user comfort for prolonged wear are acknowledged as essential considerations for further development. The 3D model lays the foundation for a user-friendly gadget in subsequent stages of our research.

In summary, our system represents a crucial step towards empowering visually impaired individuals to re-engage with reading by making Braille more accessible. The combination of advanced technology, machine learning, and hardware adaptability lays the groundwork for a potential solution that goes beyond a mere prototype, aiming to address the diverse needs and challenges faced by the visually impaired community.

II. RELATED WORKS

In recent years, the classification of Braille patterns is predominantly performed using various deep learning models and few works used SVM(Support Vector Machine) which have also shown accuracy over 90%. A lightweight CNN model with IRB(Inverted Residual Block) was used to identify Braille patterns from two datasets; one consisting of English Braille and the other one was of Chinese Braille in [4]. Different types of image preprocessing techniques were also used for aligning and enhancing the training images. In [5], the possibility of the braille image being slightly tilted(-1 to 1 degrees) was also considered when training the deep neural network and using the ”find contour” method. It was found that a tilt of 1.5 degrees would make the system unable to recognize the pattern. Similar to the previously mentioned paper [4], they have also used various preprocessing techniques like cropping, thresholding, dilation, erosion and greyscale image preprocessing techniques. In [6], a deep neural network was proposed to identify the Braille characters. Restoration of old Braille literature by converting them to digital form was done by scanning the books and then classifying the braille characters using the AlexNet deep neural network in [7]. They also used two dot-detecting algorithms; Hough transformation and cascade routine with LBP characteristics to classify the normal and reverse dots present in the Braille patterns.

A Support Vector Machine (SVM) was used in [8] to recognize Braille characters. For translating Braille characters to alphabet text that is readable by normal people, in [9] SVM and BOF(Bag of Feature) were used. Firstly, the BOF was used to identify the Braille patterns which have two subprocesses; SURF(Speeded-up Robust Features) and K-means clustering. Then, the SVM was used to classify the images into different characters of the Braille system. In [10], a feature extractor was trained for initializing the weights to be used in the deep learning model that can recognize the Braille patterns. They have used an unsupervised greedy layer-wise algorithm to train the feature extractor. Mainly, it was a proposed solution to mitigate the problem associated with dimension reduction and auto-feature extraction that arises when constructing such a deep learning model. A character-based Braille translator was proposed in [11] using three versions of the ResNet deep learning model which includes the ResNet-18, ResNet-34, and the ResNet-50. Additionally, a novel word-based detector was proposed which was coined as the ABCNet(Adaptive Bezier-Curve Network) that uses a STR(Scene Text Recognition) method. To evaluate the performance of this proposed network, a comparative analysis was conducted between these deep learning models. In [12], a general assistance system in the form of an eyeglass for visually impaired people was presented. Unlike the previous papers, this work had a different approach; they focused on building a set of features that can aid a visually impaired person in conducting day-to-day tasks rather than classifying Braille patterns. There are four distinct modules; object detection, video stream analysis, face recognition, and conversion between text and speech. Similar to our work, they have also used a Raspberry Pi which is embedded within the spectacles.

III. ARCHITECTURAL OVERVIEW OF THE SYSTEM

A. System Overview and Virtual Prototype:

The BrailleSense system introduces a virtual prototype of gloves equipped with a camera, specifically designed to assist individuals with visual impairments in reading Braille texts. The primary objective is to alleviate the challenges associated with memorizing Braille patterns by employing a classification approach. The camera positioned on the index finger identifies the focused Braille pattern, thereby facilitating users in learning or utilizing the system without the need for extensive memorization.

B. High-Level Workflow:

The high-level workflow of the system is illustrated in Figure 2. The process begins with capturing images of Braille text using a camera. Subsequently, the captured images undergo preprocessing to enhance the Braille characters for more effective processing. These preprocessed images are then fed into the BrailleNet CNN model, the core of the system responsible for recognizing and classifying Braille characters. The model outputs predicted character classes for each identified Braille cell in the image. These predicted characters are concatenated to form the final recognized Braille text string, which is then converted into voice.

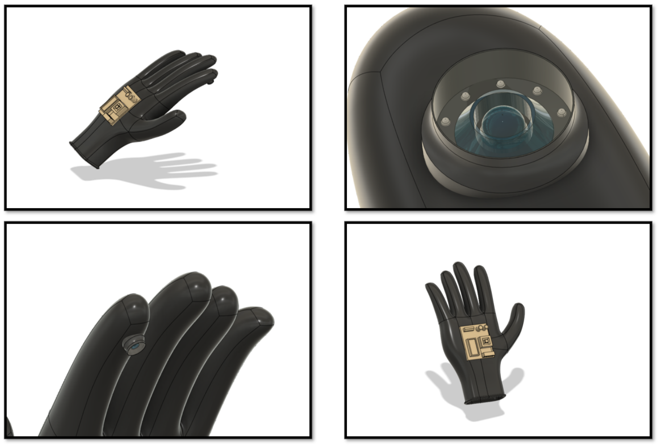

Fig. 1. 3D Model of the Gloves

C. Proof of Concept - 3D Model:

As a proof of concept, in Fig 1 a simplified 3D version of the gloves has been modeled. Note that, for now, the development of these gloves is limited to this 3D prototype. This 3D model provides a visual representation of the envisioned design, outlining the space allocation for the processing unit and camera within the gloves. It serves as an initial design concept upon which a more robust and user-friendly gadget might be developed in the future.

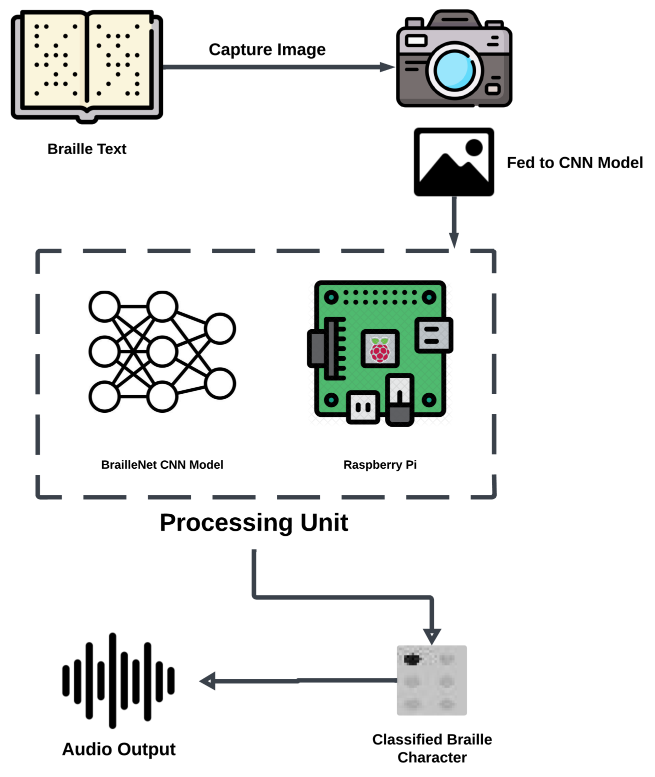

Fig. 2. High-level Overview of the system

IV. DATASET DESCRIPTION

We used the Braille Character Dataset [13] to train our model. This dataset was curated for training a Convolutional Neural Network (CNN) intended for Braille Character Recognition. The dataset includes 26 characters of the English alphabet, each subjected to 3 augmentations, resulting in a total of 20 different images with distinct augmentation values (e.g., different shift, rotational, and brightness values).

A. Image Description

Each image is a 28x28 grayscale image. The image filename comprises the character alphabet, the image number, and the type of data augmentation it underwent (e.g., `whs` for width height shift, `rot` for rotation, `dim` for brightness).

B. Dataset Composition

The dataset aims to provide diversity in terms of augmentation to enhance the robustness of the CNN model. Each character has been augmented through width-height shift, rotation, and brightness adjustment. The total number of images in the dataset is $26 \times 3 \times 20 = 1560$, ensuring a comprehensive training set for effective Braille character recognition.

C. Summary Table of the Dataset

The summary of the dataset is shown in Table I.

Character	Augmentation Type	Number of Images
A	whs	20
A	rot	20
A	dim	20
B	whs	20
B	rot	20
B	dim	20
C	whs	20
C	rot	20
C	dim	20
D	whs	20
D	rot	20
D	dim	20
E	whs	20
E	rot	20
E	dim	20
...	...	...
W	whs	20
W	rot	20
W	dim	20
X	whs	20
X	rot	20
X	dim	20
Y	whs	20
Y	rot	20
Y	dim	20
Z	whs	20
Z	rot	20
Z	dim	20

TABLE I
SUMMARY OF BRAILLE CHARACTER DATASET

V. HARDWARE DEVICES AND CONFIGURATION

As our IoT device, we have selected the Raspberry Pi 4 Model B. Table II lists the specifications of our particular version of the Raspberry Pi [1]. Since the implementation of the CNN model requires various machine learning work-related packages, we have opted for a custom Raspberry Pi OS [2] that has libraries like TensorFlow, TensorFlow-Lite, Pytorch, TorchVision, Paddle and Paddle-Lite preinstalled giving us a build-ready OS out-of-the-box. In order to burn this custom OS image file in a 32 GB SD card, we have used the official Raspberry Pi imager. Additionally, the SSH protocol was used to configure and execute the Python program in the Raspberry Pi from a Windows Laptop using the PuTTY [3] software. Moreover, the Raspberry Pi OS customization not only includes the necessary machine learning libraries but also streamlines the development environment, minimizing setup complexities. The 32 GB SD card offers ample storage capacity for the dataset, and the 8 GB RAM for model weights, and additional resources required during execution. The choice of the Raspberry Pi 4 Model B aligns with our goal of creating a compact, energy-efficient, and cost-effective IoT device capable of executing real-time Braille pattern classification. For the deployment process, the official Raspberry Pi imager simplifies the OS installation, making it accessible even for users with limited technical

[1]`https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/`
[2]`https://github.com/Qengineering/RPi-Bullseye-DNN-image`
[3]`https://www.putty.org/`

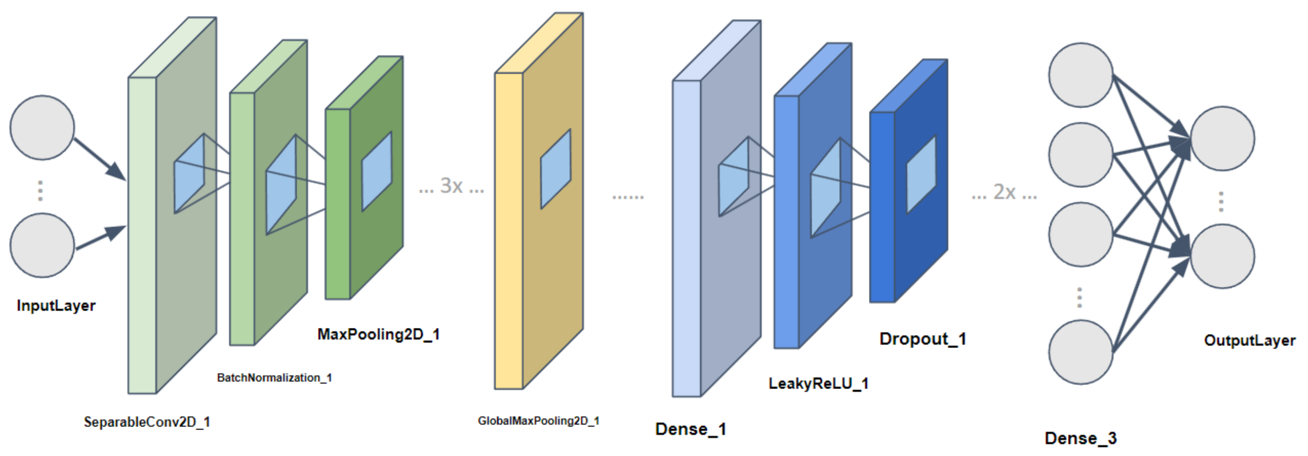

Fig. 3. BrailleNet Model Architecture

expertise. This user-friendly approach aligns with our objective of creating an inclusive and user-centric solution. Additionally, leveraging the SSH protocol for configuration and program execution enhances flexibility, enabling remote interaction with the Raspberry Pi. The PuTTY software facilitates a seamless connection, making it convenient for users to control the device and monitor its performance. In summary, our selection of the Raspberry Pi as the IoT device, coupled with an ML-tailored OS and streamlined deployment process, underscores our commitment to creating an accessible, efficient, and user-friendly platform for Braille pattern classification. This integrated setup lays the foundation for future enhancements and scalability, ensuring the adaptability of our solution to evolving requirements and advancements in the field of assistive technology.

TABLE II
RASPBERRY PI 4 MODEL B SPECIFICATIONS

Entity	Specification
Processor	Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz
RAM	8 GB
Power consumption	2.5 A
Operating temperature	0 - 50 degrees C ambient

VI. MODEL ARCHITECTURE

In this section, we provide an in-depth dissection of a custom Convolutional Neural Network (CNN) model, coined as the BrailleNet, designed for Braille Image Classification. The summary of the model architecture is shown in Fig 3.

A. Input Layer

The model begins with an InputLayer, accommodating variable batch sizes and a specific format: 28x28 pixel images with 3 channels, likely representing RGB colors.

B. Convolutional Blocks: Extracting and Refining Features

The output of a convolutional layer can be calculated using the convolution operation. Assuming X is the input tensor, W is the weight tensor, and b is the bias, the convolution operation can be expressed as:

$$Y[i,j] = \sum_{m}\sum_{n} X[i+m, j+n] \cdot W[m,n] + b$$

Here, Y is the output tensor, and, i, j iterate over the spatial dimensions of Y.

The core of the network lies in three sequential convolutional blocks, each meticulously crafted to extract and refine image features.

1) SeparableConv2D layer: This layer extracts features using separable convolutions, a computationally efficient approach compared to traditional convolutions. It utilizes 64 filters to extract initial features, resulting in a slightly smaller output due to border effects and padding.

2) BatchNormalization layer: This layer ensures stable training by normalizing internal activations. It employs 128 filters and 2x2 max pooling, significantly reducing the spatial dimensions while further refining features.

3) MaxPooling2D layer: This layer downsamples the feature maps, reducing spatial resolution and boosting robustness to input variations. It uses 256 filters for even deeper feature extraction, again leading to a smaller output through max pooling.

C. Global Max Pooling: Image Mining

Max pooling is a downsampling operation. Assuming X is the input tensor, the max pooling operation with a pool size of k×k can be expressed as:

$$Y[i,j] = \max_{m,n} X[i \times k + m, j \times k + n]$$

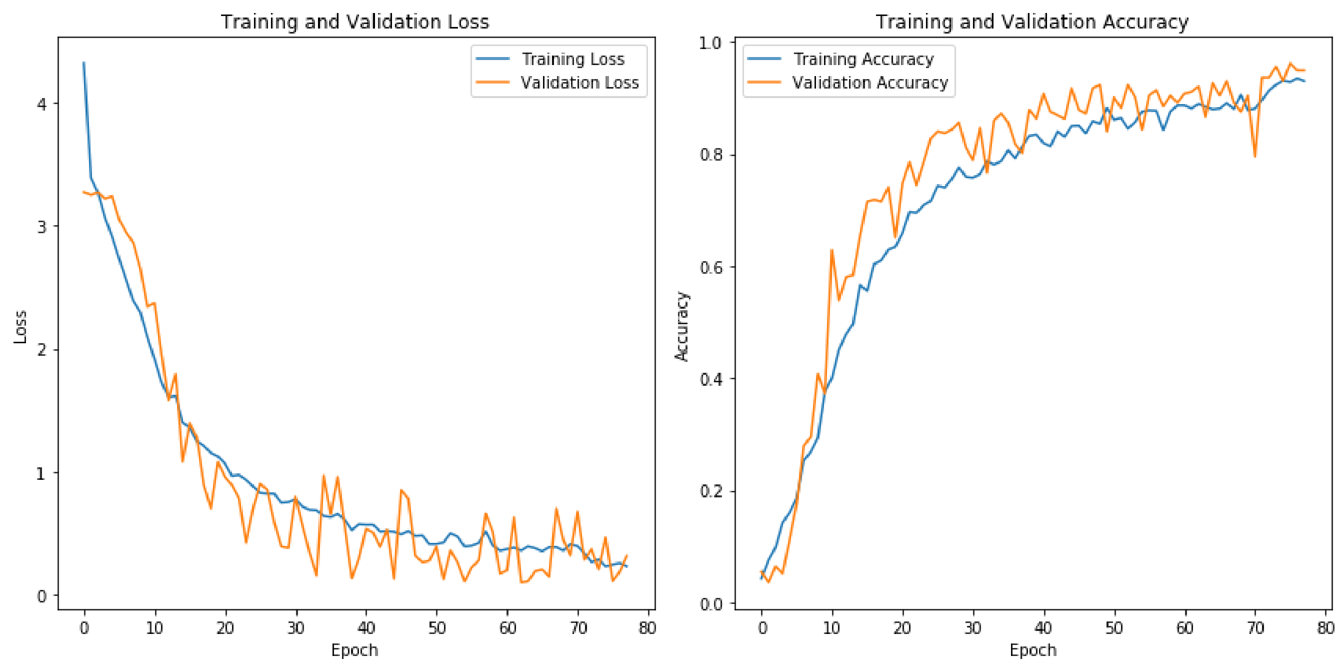

Fig. 4. Loss and Accuracy Curves. (Left) Training and validation loss during training. (Right) Training and validation accuracy. The curves plateau around 97.44% accuracy.

Here, Y is the output tensor, and, i,j iterate over the spatial dimensions of Y.

1) global_max_pooling2d: This layer captures the most dominant features across the entire image by summarizing each 4x4 feature map into a single element, creating a 256-dimensional feature vector.

2) Dense layers and Leaky ReLU activations: These layers project the features into new spaces and introduce non-linearity, enhancing the model's expressiveness. ReLU Function:

$$f(x) = \max(0, x)$$

3) Dropout layers: These layers randomly deactivate some neurons, promoting generalization and reducing the possibility of overfitting.

D. Final Output Layer

For a fully connected layer, assuming X is the input vector, W is the weight matrix, and b is the bias vector, the output can be calculated as:

$$Y = \sigma(W \cdot X + b)$$

Here, Sigma is the activation function (e.g., softmax for classification).

The final dense layer maps the processed features to 26 output neurons, indicating the model's potential to classify images into 26 distinct categories.

VII. Result Analysis

This section evaluates the performance of the proposed model through both loss analysis and accuracy evaluation. Analyzing the model's behaviour on both training and validation sets provides crucial insights into its generalizability and potential for real-world deployment.

A. Loss Analysis and Accuracy Evaluation

Figure 4 (Left) depicts the training and validation loss curves during the training process. Both curves exhibit a downward trend, indicating decreasing discrepancies between the model's predictions and actual values. The validation loss closely follows the training loss, suggesting minimal overfitting and effective generalization to unseen data.

Figure 4 (Right) presents the training and validation accuracy curves. Both curves steadily increase, culminating in a plateau around 97.44%. This remarkably high accuracy demonstrates the model's ability to correctly classify approximately 97.44% of the data points in both training and validation sets. The proximity of the training and validation accuracy curves further reinforces the model's generalizability and resistance to overfitting.

B. Model Evaluation on Different Matrices

The performance of the BrailleNet model is evaluated using key metrics: accuracy, recall, f1-score, and precision. These metrics provide insights into the model's ability to correctly classify Braille patterns.

- **Accuracy**: It is the ratio of correctly predicted instances to the total instances. The accuracy of our model is

97.44%.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

- **Recall (Sensitivity or True Positive Rate)**: It is the ratio of correctly predicted positive observations to all observations in the actual class. The recall of our model is 96.78%.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1-Score**: It is the weighted average of precision and recall. It considers both false positives and false negatives. The F1-Score of our model is 96.81%.

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Precision (Positive Predictive Value)**: It is the ratio of correctly predicted positive observations to the total predicted positives. The precision of our model is 95.67%.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Metric	**Value (%)**
Accuracy	97.44
Recall	96.78
F1-Score	96.81
Precision	95.67

TABLE III

SUMMARY OF MODEL EVALUATION METRICS

VIII. CONCLUSION AND FUTURE WORK

In conclusion, the BrailleSense system represents a significant leap forward in assisting visually impaired individuals in learning and utilizing the Braille system. Through the deployment of a custom Convolutional Neural Network (CNN) model on a Raspberry Pi, our system achieves an impressive accuracy of 97.44%, showcasing its potential for real-world applications. The loss analysis and accuracy evaluation demonstrate the model's robustness, generalizability, and resistance to overfitting. The virtual prototype of hand gloves equipped with a camera provides a tangible representation of our technological solution. The BrailleSense system addresses key challenges in Braille literacy, particularly for those losing sight later in life, offering a promising tool for enhanced literacy and independence. The next focus is on optimizing the BrailleSense system for real-time applications. We plan to develop a Recurrent Neural Network (RNN) model [14], specifically designed for sequential data processing. This upgrade aims to enhance the system's ability to classify Braille patterns in real-time.

The summary of the model evaluation is shown in III. These findings highlight the success of the proposed model in achieving high accuracy with minimal overfitting.

REFERENCES

[1] S. Isayed and R. Tahboub, "A review of optical braille recognition," in *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*. IEEE, 2015, pp. 1–6.

[2] D. Dimitrova-Radojichikj *et al.*, "Students with visual impairments: Braille reading rate," *International Journal of Cognitive Research in Science, Engineering and Education (IJCRSEE)*, vol. 3, no. 1, pp. 1–5, 2015.

[3] H. Kawabe, Y. Shimomura, H. Nambo, and S. Seto, "Application of deep learning to classification of braille dot for restoration of old braille books," in *Proceedings of the Twelfth International Conference on Management Science and Engineering Management*. Springer, 2019, pp. 913–926.

[4] T. Kausar, S. Manzoor, A. Kausar, Y. Lu, M. Wasif, and M. A. Ashraf, "Deep learning strategy for braille character recognition," *IEEE Access*, vol. 9, pp. 169 357–169 371, 2021.

[5] J. Subur, T. A. Sardjono, and R. Mardiyanto, "Braille character recognition using find contour and artificial neural network," *JAVA Journal of Electrical and Electronics Engineering*, vol. 14, no. 1, 2016.

[6] K. Smelyakov, A. Chupryna, D. Yeremenko, A. Sakhon, and V. Polezhai, "Braille character recognition based on neural networks," in *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*. IEEE, 2018, pp. 509–513.

[7] H. Kawabe, Y. Shimomura, H. Nambo, and S. Seto, "Application of deep learning to classification of braille dot for restoration of old braille books," in *Proceedings of the Twelfth International Conference on Management Science and Engineering Management*. Springer, 2019, pp. 913–926.

[8] J. Li and X. Yan, "Optical braille character recognition with support-vector machine classifier," in *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 12. IEEE, 2010, pp. V12–219.

[9] S. Ibrahim, N. A. Tarmizi, N. Sabri, N. F. M. Johari, and A. F. A. Fadzil, "Braille image recognition for beginners," in *2018 9th IEEE Control and System Graduate Research Colloquium (ICSGRC)*. IEEE, 2018, pp. 81–84.

[10] T. Li, X. Zeng, and S. Xu, "A deep learning method for braille recognition," in *2014 International Conference on Computational Intelligence and Communication Networks*. IEEE, 2014, pp. 1092–1095.

[11] C. Li and W. Yan, "Braille recognition using deep learning," in *Proceedings of the 4th International Conference on Control and Computer Vision*, 2021, pp. 30–35.

[12] A. Bhuiyan, M. A. Islam, M. H. Shahriar, T. H. Supto, M. A. Kasem, and M. E. Daud, "An assistance system for visually challenged people based on computer vision and iot," in *2020 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2020, pp. 1359–1362.

[13] Shanks0465. (2020) Braille character dataset. Accessed on: Date accessed. [Online]. Available: https://www.kaggle.com/datasets/shanks0465/braille-character-dataset

[14] S. Zaman, M. A. Abrar, M. M. Hassan, and A. N. Islam, "A recurrent neural network approach to image captioning in braille for blind-deaf people," in *2019 IEEE International Conference on Signal Processing, Information, Communication & Systems (SPICSCON)*. IEEE, 2019, pp. 49–53.