# **Real Time Hand-Written Digit Recognition to Speech**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Carlos Alonso Gradillas  [c.alonso68@usp.ceu.es](mailto:c.alonso68@usp.ceu.es)  22320065  CEU san Pablo  Computer Science | Park John  [Parkjohn1027@handong.ac.kr](mailto:Parkjohn1027@handong.ac.kr)  21900307  Handong Global University  ICT Convergence | Alfin  [22100826@handong.ac.kr](mailto:22100826@handong.ac.kr)  22100826  Handong Global University  ICT Convergence |  |  |

**Abstract**

*Real-time digit classification is a fascinating challenge with significant implications for applications designed to assist visually impaired individuals. This study proposes the development of a dedicated application that integrates advanced machine learning models for real-time digit recognition. Specifically, we leverage a convolutional neural network (CNN) and the AlexNet model, utilizing the MNIST dataset loaded via the Keras library. Our objective is to seamlessly embed these models into a user-friendly Pygame-based application, thereby enabling real-time digit recognition. To enhance the user experience, our application includes a voice output feature that announces the recognized digit, providing immediate auditory feedback to the user. Conducted by Caros Alonso Gradillas, Alfin, and Park John at Handong Global University (HGU), this research emphasizes the design and implementation of the CNN model within the application framework, highlighting its potential to aid visually impaired individuals. We achieved a 97% accuracy rate with the AlexNet model and 99% with the CNN model. Our project demonstrates the practical utility of deep learning in creating accessible tools for digit recognition. This project exemplifies how cutting-edge technology can be harnessed to improve daily life for visually impaired users, showcasing the significant benefits of integrating deep learning models into real-time applications.*

**GitHub Repository: [Link to GitHub Repository]**

# **Introduction**

**1.1 Problem Definition**

Visually impaired individuals encounter numerous obstacles when trying to interpret numerical data from various sources, such as electronic displays, written documents, and educational materials. Traditional assistive technologies, while helpful, often fall short in providing real-time, intuitive, and accurate digit recognition. There is a pressing need for an accessible tool that can recognize digits in real-time and offer immediate, clear auditory feedback, thereby enhancing the independence and efficiency of visually impaired users in educational, professional, and everyday contexts.

Consequently, Real-time digit classification offers a valuable opportunity to enhance assistive technologies for visually impaired individuals. This project aims to develop a sophisticated application that utilizes cutting-edge deep learning models for real-time digit recognition. By integrating a convolutional neural network (CNN) and the AlexNet model, we strive to create a reliable tool that can provide instant auditory feedback through voice output, significantly aiding users with visual impairments.

**1.2. Literature Survey**

In our research on real-time digit classification, we examined five significant studies that provide a comprehensive understanding of various approaches and technologies used in this field. These articles offer valuable insights into the methodologies, challenges, and advancements in digit recognition, particularly focusing on applications that can aid visually impaired individuals.

**1.2.1 Article I:**

In the study titled Real-Time Handwritten Digits Recognition Using Convolutional Neural Network, authors Kaveti Upender and Venkata Siva Kumar Pasupuleti address the challenge of predicting real-time handwritten digits using the MNIST dataset for model training. The authors utilized the OpenCV Python library to detect patterns in handwritten digits, achieving human-level accuracy with a Convolutional Neural Network (CNN) model. This research highlights the effectiveness of CNNs in real-time handwritten digit recognition, providing a foundation for our application’s digit recognition component (Upender & Pasupuleti, n.d.).

**1.2.2** **Article II:**

In their article titled *Handwritten Digit Recognition Using Machine and Deep Learning Algorithms*, authors Ritik Dixit, Rishika Kushwah, and Samay Pashine provide a comprehensive comparison of various machine learning and deep learning algorithms for handwritten digit recognition. The study evaluates algorithms such as Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Network (CNN) based on metrics including accuracy, errors, and training-testing time. Visualization using matplotlib supports their analysis. The findings suggest that CNNs outperform other models in digit recognition tasks, reaffirming our decision to incorporate CNNs into our application (Dixit, Kushwah, & Pashine, n.d.).

**1.2.3 Article III:**

In their research titled *Image Recognition and Voice Translation for Visually Impaired*, authors Sandeep Pasupuleti, Lahari Dadi, Manikumar Gadi, and R. Krishnaveni focus on merging image recognition with voice translation to assist visually impaired individuals. The study employs VGG16 and Long Short-Term Memory (LSTM) networks to enhance recognition and translation capabilities. Utilizing both Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN), the research underscores the significance of integrating robust neural network models to achieve high accuracy in recognition tasks, which is aligned with our project’s objectives to provide auditory feedback (Pasupuleti et al., n.d.).

**1.2.4 Article IV:**

In her thesis titled *An IoT System for Converting Handwritten Text to Editable Format via Gesture Recognition*, author Nidhi Patel proposes an Internet of Things (IoT) framework utilizing hand gesture recognition (HGR) with Raspberry Pi and a camera to convert handwritten text into an editable format. The system integrates edge detection algorithms to mitigate computational complexity and employs OpenCV and machine learning algorithms to transform handwritten elements into editable text. This study underscores the potential of merging IoT with machine learning for practical applications, which resonates with our project's objective of improving accessibility through technology (Patel, n.d.).

**1.2.5 Article V:**

In their paper titled *NeuroWrite: Predictive Handwritten Digit Classification using Deep Neural Networks*, authors Kottakota Asish, P. Sarath Teja, R. Kishan Chander, and D. Deva Hema discuss the challenges and solutions involved in handwritten digit recognition using deep neural networks. The study utilizes the MNIST dataset and implements various methodologies such as CNN-based deep handwritten text recognition (DHTR), few-shot learning, and dictionary learning. Emphasizing the significance of neural networks, the research underscores their role in distinguishing visually similar digits and handling handwriting variations, which are critical for our project's success in achieving high accuracy and reliability (Asish et al., n.d.).

## **2. Dataset**

**2. 1 Keras/Mnist Library**

At the heart of our project lies the MNIST dataset, a cornerstone in the realm of handwritten digit recognition. Comprising 70,000 grayscale images depicting digits 0 through 9, each meticulously labeled, MNIST serves as a standardized benchmark for training and testing image processing systems. Its division into 60,000 training images and 10,000 test images facilitates supervised learning, enabling us to evaluate model performance effectively. While MNIST forms the backbone of our dataset, we augment our resources with open-source repositories from platforms like GitHub, leveraging existing implementations and models to expedite our development process and refine our approach.

텍스트, 디자인, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**Figure 1. Overall Process**

In our endeavor to create a model capable of transforming handwritten letters into audio, we rely on Python's open-source neural networks library. This versatile tool, designed for simplicity and modularity, empowers us to construct and train deep learning models seamlessly. With support for various neural network architectures and compatibility with TensorFlow and Theano, the library ensures efficient computation and flexible deployment. Additionally, its compatibility with CPU and GPU acceleration, coupled with provisions for distributed training, equips us to optimize performance and scalability. As we embark on this journey, we are bolstered by a vibrant community and extensive documentation, confident in our ability to harness deep learning to address the challenges of handwritten digit recognition and audio transformation.

타이포그래피, 폰트, 흑백, 화이트이(가) 표시된 사진

자동 생성된 설명

**Figure 2. Overall Process**

## **3. Technical approach**

For technical approach, we used a pygame app to get the user input and immediately recognize the digit for the expected outcome. The upcoming approach will be comparing other models and incorporating voice recognition to the predicted number.

텍스트, 스크린샷, 폰트, 로고이(가) 표시된 사진

자동 생성된 설명

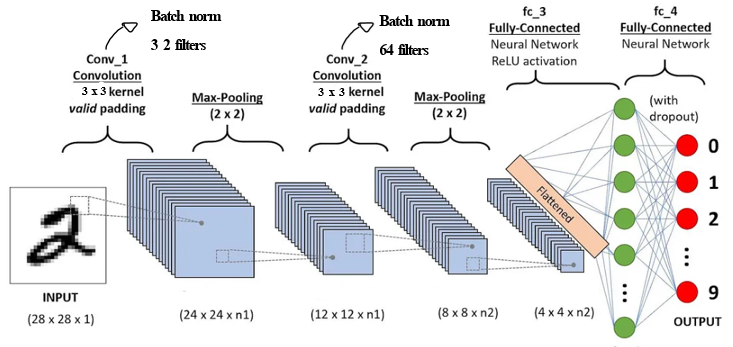
**Figure 3. Overall Process**

To commence our project, we capitalize on the Keras MNIST library to extract the requisite data, furnishing us with a substantial dataset pivotal for training our neural network model. Empowered by the extensive functionalities of the Keras library, we adeptly assemble and train our Convolutional Neural Network (CNN) model. Meticulously crafted, this model architecture features alternating convolutional and pooling layers, bolstered by batch normalization and dropout layers, thus augmenting regularization and bolstering generalization performance.

Upon the successful training of the CNN model, we seamlessly integrate a preprocessing pipeline, drawing upon existing source code to refine user input and ensure impeccably accurate predictions. This preprocessing stage plays a pivotal role in the selection and accurate classification of handwritten numbers, laying the bedrock for our subsequent real-time digit recognition application.

In the ultimate phase of our project, we execute the implementation of a Pygame application to facilitate real-time user input of newly created numbers. This interactive interface not only enriches user engagement but also enables the seamless integration of our digit recognition model into practical applications. With each phase meticulously orchestrated, we are primed to deliver a resilient and user-friendly solution for handwritten digit recognition, further underscoring the prowess of deep learning, including the utilization of AlexNet in our approach.

**2.2 Architecture of CNN model**



**Figure 4. Architecture of CNN model**

The model consists of an input layer where the input shape is specified by input\_shape. This is where the model expects input images of a particular shape. Moreover, convolutional layers, there are two sets of convolutional layers with 32 and 64 filters respectively. Each convolutional layer uses a 3x3 kernel size and ReLU activation function to extract features from the input images.

Also, Batch normalization layers are added after each convolutional layer. These layers help in stabilizing and speeding up the training process by normalizing the input to the next layer. Max pooling layers follow each set of convolutional layers. They down sample the feature maps obtained from the convolutional layers by taking the maximum value from each non-overlapping 2x2 patch. Dropout layers are added after each max pooling layer to prevent overfitting. They randomly set a fraction of input units to zero during training, which helps in reducing the likelihood of overfitting by preventing the network from becoming overly reliant on any particular set of features.

Then the flatten layer is used to convert the 2D feature maps into a 1D vector, which can be fed into the densely connected layers. This layer has 128 neurons and ReLU activation function. This layer learns high-level features from the flattened representations of the input. Batch normalization and dropout layers are again added after the dense layer to further regularize the model. The final layer is a dense layer with a SoftMax activation function, which produces the probability distribution over the output classes. The number of neurons in this layer is determined by number of classes, and it is activated by SoftMax to output class probabilities. Finally, the model is compiled with categorical cross-entropy as the loss function, Adam optimizer, and accuracy as the evaluation metric.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**Figure 5. Preprocessing of CNN model**

**2.3 AlexNet model**

AlexNet represents a pioneering convolutional neural network (CNN) architecture that reshaped the landscape of deep learning, particularly in the domain of image recognition. Comprising five convolutional layers seamlessly integrated with three fully connected layers, AlexNet's inception marked a paradigm shift, ushering in unprecedented advancements in accuracy and efficiency. Beginning with an input layer tailored for 227x227x3 images, the network embarks on a transformative journey of feature extraction and refinement.

도표, 스크린샷, 텍스트, 라인이(가) 표시된 사진

자동 생성된 설명

**Figure 6. Architecture of AlexNet Model**

The initial layers, notably Conv1 and Conv2, lay the groundwork by applying large convolutional filters and leveraging Rectified Linear Unit (ReLU) activation, augmented by max-pooling and local response normalization (LRN). As the network delves deeper, Conv3, Conv4, and Conv5 continue this saga of feature refinement, harnessing smaller filters to capture intricate details within the image. Transitioning to fully connected layers, FC1 and FC2 serve as conduits for aggregating higher-level features, each boasting 4096 neurons and ReLU activation.

Finally, FC3 emerges as the apex, housing 1000 neurons corresponding to ImageNet classes, orchestrating the synthesis of a probability distribution. Noteworthy is the strategic use of dropout between fully connected layers, a potent weapon against overfitting. The success of AlexNet in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012, boasting a top-5 error rate of 15.3%, cemented its status as a transformative force in deep learning.

**2.4 Soken Digit**

Pyttsx3 version 2.90 is a versatile text-to-speech conversion library for Python, offering offline functionality and compatibility with both Python 2 and 3. With Pyttsx3, users can effortlessly convert text into audible speech, enhancing accessibility and usability across various applications. One of the key features of Pyttsx3 is its ability to control volume and voice characteristics. Users can dynamically adjust volume levels from 0 to 1, tailoring the speech output to their preferences. Additionally, Pyttsx3 provides access to multiple voices, allowing users to choose between different genders and accents to suit their needs. Pyttsx3 offers fine-grained control over speaking rates, enabling users to adjust the speed of speech output according to their preferences. By customizing the speaking rate, users can ensure optimal comprehension and engagement with the synthesized speech.

Overall, Pyttsx3 empowers developers to integrate robust text-to-speech functionality into their Python applications, facilitating seamless communication and interaction with users.

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

**Figure 7. Soken Digit**

In addition to its user-friendly interface and versatile functionality, Pyttsx3 version 2.90 provides developers with comprehensive control over various aspects of speech synthesis. With Pyttsx3, developers can access detailed properties of the speech output, including volume, voice characteristics, and speaking rates. This granular control allows for precise customization of the synthesized speech, ensuring that it aligns with the specific requirements and preferences of users.

Moreover, Pyttsx3 offers seamless integration with Python applications, facilitating effortless implementation of text-to-speech functionality across diverse projects. Whether developers are creating accessibility features for visually impaired users, developing interactive chatbots, or enhancing user experience in multimedia applications, Pyttsx3 provides the necessary tools and capabilities to realize their vision. By leveraging Pyttsx3's robust features and intuitive API, developers can streamline the development process and deliver immersive, engaging, and accessible user experiences.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**Figure 8. Voice Digit Setting**

In the context of our project, once we receive user input in the form of handwritten digits, our system's predictive capabilities come into play. Leveraging the output of our trained model, which accurately predicts the digit represented by the input, we seamlessly assign a corresponding spoken digit to enhance user experience. By mapping the predicted digit to its auditory counterpart, our system ensures accessibility and ease of understanding for users, particularly those with visual impairments. This integration of spoken digits adds an extra layer of utility and inclusivity to our application, further enhancing its practicality and usability in real-world scenarios.

**3. Experiment Design**

**3.1. Training the Models**

Several key improvements have been implemented to enhance the performance and robustness of the neural network model. Batch normalization has been incorporated after each convolutional and dense layer, optimizing training speed and stability by normalizing the activations. Additionally, the dropout rate has been increased to 0.5, effectively mitigating overfitting by randomly dropping out nodes during training. Furthermore, L2 regularization has been applied to both convolutional and dense layers, serving as a preventative measure against overfitting by penalizing large weights.

In terms of optimization algorithms, the switch to SGD with momentum has been made, leveraging momentum to accelerate gradient descent in directions with consistent gradients and dampen oscillations in others, potentially leading to better convergence for deep networks. Moreover, a learning rate scheduler has been employed to dynamically adjust the learning rate during training, allowing for finer control over the optimization process and potentially improving model performance and convergence. Collectively, these enhancements contribute to a more stable, efficient, and effective training process, ultimately resulting in improved performance and generalization of the neural network model.

**텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명**

**Figure 9. Training CNN model**

**4. Results**

**4.1 Result of CNN Model**

스케치, 디자인, 흑백, 타이포그래피이(가) 표시된 사진

자동 생성된 설명

**Figure 10. Result of Digit Recognition**

텍스트, 스크린샷, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

**Figure 11. Result of CNN model**

The training and validation performance of the neural network model reveals several key insights into its effectiveness and areas for potential improvement. During training, the model exhibits rapid initial learning, reflected in a steep increase in training accuracy and a corresponding decrease in training loss, indicating a strong fit to the training data. However, as training progresses, validation accuracy and loss plateau, with a slight discrepancy between training and validation metrics suggesting some degree of overfitting. Despite this overfitting, the model demonstrates good generalization to unseen data, as evidenced by the relatively high validation accuracy.

Key observations highlight the presence of overfitting, indicated by the persistent gap between training and validation metrics. To address this, potential improvements include increasing dropout rates or incorporating L2 regularization to mitigate overfitting tendencies. Additionally, leveraging data augmentation techniques to introduce variability into the training data and implementing early stopping mechanisms to halt training when validation loss ceases to improve could further enhance model performance and generalization capabilities. By systematically addressing these areas for improvement, the neural network model can be refined to achieve greater robustness and effectiveness in real-world applications.

텍스트, 스크린샷, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 라인, 도표이(가) 표시된 사진

자동 생성된 설명

**Figure 12. Improved Result of CNN model**

The analysis of key observations unveils significant strides in mitigating overfitting within the model architecture. Through careful adjustments and a reduction in the number of training epochs, the gap between training and validation metrics has notably narrowed, indicating a more balanced and robust learning process. This reduction in overfitting not only enhances the model's ability to generalize to unseen data but also instills greater confidence in its predictive capabilities. Moreover, the swift convergence observed, characterized by high accuracy and low loss achieved within a few epochs, underscores the efficiency of the training process. This quick convergence suggests a well-chosen learning rate and an adeptly crafted model architecture, further bolstering the model's effectiveness and reliability.

Furthermore, the remarkable performance exhibited by the model across both training and validation datasets reflects its overall prowess and generalization capabilities. With consistently high accuracy and low loss metrics observed in both datasets, the model demonstrates its capacity to effectively capture and learn from underlying patterns within the data. This stellar performance not only validates the efficacy of the improvements made to the model architecture but also underscores its potential utility in real-world applications. As such, the culmination of reduced overfitting, quick convergence, and exemplary performance solidifies the model's position as a powerful tool for data analysis and prediction, poised to make significant contributions across various domains.

**4.2 Result of AlexNet Model**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명스크린샷, 멀티미디어 소프트웨어, 소프트웨어, 그래픽 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**Figure 13-14. Result of AlexNET Model**

The training process of an AlexNet model for digit recognition over 50 epochs demonstrates effective learning and performance. Starting with a training accuracy of 78.9%, the model improves to around 97.1% by the 50th epoch. The validation accuracy remains consistently high, beginning at 97.1% and slightly decreasing to 96.8% by the end. Both training loss and validation loss show a decreasing trend, indicating successful learning. Overall, the model performs well on both training and validation sets, suggesting it has effectively learned to recognize digits.

When we tried with our model the results were very accurate with every number that we tried in real time writing was accurately predicted. However, when tried using our improved model it did as well as the previous model. We can ensure that the model was accurate even after we lower the epochs and upon the model architecture.

## **5. Limitations**

The integration of digit recognition with speech output encountered several limitations that warrant attention for future improvements. Particularly challenging were digits with intricate shapes or similarities, such as zero, eight, seven, and nine, which posed difficulties in accurate recognition and subsequent conversion into speech. This limitation may have hindered the overall effectiveness of the system, as inaccuracies in speech output could lead to confusion or misinterpretation of the recognized digits, thereby impacting user experience and usability.

Furthermore, the system faced challenges when presented with multiple digits drawn simultaneously. Recognizing and interpreting multiple digits concurrently introduced complexities in the digit recognition process, potentially leading to errors or inaccuracies in the speech output. These limitations underscore the importance of refining the digit recognition algorithm and optimizing the speech synthesis functionality to ensure robust performance across various scenarios, ultimately enhancing the system's usability and effectiveness for users with visual impairments.

## **6. Future goals**

Our vision transcends the realm of real-time digit recognition, extending towards the creation of a comprehensive solution seamlessly integrated into ubiquitous devices like smartphones and smart glasses. By breaking free from conventional limitations, our aim is to empower individuals with visual impairments through auditory cues that illuminate their surroundings. This holistic approach serves as a beacon of awareness, offering invaluable insights into the environment and providing individuals with the ability to stay informed about unfolding events in real time. Through the innovative fusion of cutting-edge technology, we are committed to illuminating the path towards inclusivity and accessibility, nurturing independence, and confidence among those navigating the world with visual limitations.

In essence, our project aspires to harness the transformative potential of sound to transcend barriers and enrich the lives of individuals with visual impairments. By expanding our focus to encompass not only digits but also alphabets, we seek to provide a comprehensive tool that enhances daily experiences and fosters a deeper connection with the surrounding world. As we embark on this journey towards greater inclusivity and empowerment, our unwavering dedication to innovation and empathy drives us to create solutions that not only address existing challenges but also pave the way for a future where everyone can thrive, regardless of their visual abilities.

## **7. GitHub repository**

<https://github.com/carlosalongra/Ai_vision_Final_project_repo>

**References**

1. Asish, K., Teja, P. S., Chander, R. K., & Hema, D. D. (n.d.). NeuroWrite: Predictive Handwritten Digit Classification using Deep Neural Networks.
2. Dixit, R., Kushwah, R., & Pashine, S. (n.d.). Handwritten Digit Recognition Using Machine and Deep Learning Algorithms.
3. Hakur, H. (2021). *Handwriting recognition using machine learning* (Bachelor of Technology thesis). Jaypee University of Information Technology.
4. Hakur, H. (2023). *Speech recognition on spoken digit dataset*. GitHub. Retrieved from <https://github.com/aniruddhapal211316/spoken_digit_recognition>
5. Pasupuleti, S., Dadi, L., Gadi, M., & Krishnaveni, R. (n.d.). Image Recognition and Voice Translation for Visually Impaired.
6. Patel, N. (n.d.). An IoT System for Converting Handwritten Text to Editable Format via Gesture Recognition.
7. Upender, K., & Pasupuleti, V. S. K. (n.d.). Real Time Handwritten Digits Recognition Using Convolutional Neural Network.
8. Vakalopoulou, M., Christodoulidis, S., Burgos, N., Colliot, O., & Lepetit, V. (2023). Deep learning: basics and convolutional neural networks (CNN). In O. Colliot (Ed.), *Machine Learning for Brain Disorders*. Springer. DOI: 10.1007/978-1-0716-3195-9\_3
9. Spoken digit library. (n.d.). Retrieved from <https://pypi.org/project/pyttsx3/>
10. Wu, C.-E., Lee, J.-H., Wan, T. S. T., Chan, Y.-M., & Chen, C.-S. (2020). Merging well-trained deep CNN models for efficient inference. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)* (pp. 1594-1600). Auckland, New Zealand.