International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Generating Diverse Synthetic Morse Code Datasets for Neural-Network Based Classification

[1] Dr. A.M Chandrashekar, [1] Deeksha G, [1] Deeksha Devangi, [1] Dhanush R

Department of Computer Science and Engineering, Sri Jayachamarajendra College of Engineering, JSSSTU

Mysore, Karnataka, India

Abstract - This work presents a system for translating between Morse code and human-readable text, with applications in secure and efficient communication. A novel algorithm is proposed to generate one-dimensional synthetic datasets for classifying Morse code using supervised learning, especially neural networks. Despite minimal features, these datasets carry high information density and include added noise to test model robustness. An automated decoder converts Morse signals into audio, offering potential use in secure transmissions by intelligence agencies. The open-source algorithm and datasets encourage further research in classification and communication systems.

Keywords—Morse code, neural networks, machine learning, secure communication, signal decoding, dataset generation.

1. INTRODUCTION

This work introduces a practical communication system using Morse code, designed for both secure transmission and accessible communication. Morse code, once a cornerstone of early telecommunication, is leveraged here as a lightweight, low-complexity communication tool. The system is built around a single-button input mechanism, where the duration of button presses determines whether a dot or dash is registered. Pauses between presses indicate separations between letters or words. This model replicates the mechanics of traditional telegraphy while enabling minimalist and accessible communication—particularly valuable for users with physical impairments or in covert communication scenarios. To support machine learning-based Morse code decoding, the project generates synthetic datasets that simulate Morse sequences. These datasets are labeled for supervised learning tasks, enabling models to recognize and classify signals effectively. Each sequence corresponds to a specific alphanumeric character or symbol.

The artificial generation of data allows the simulation of a wide variety of real-world conditions, significantly reducing the need for manual data collection while maintaining consistency and diversity in input patterns. Morse signals are also simulated as sequences of frames, with each frame containing an intensity value representing signal strength, brightness, or amplitude, depending on the transmission medium (e.g., audio or video). This representation makes the data suitable for time-series or vision-based deep learning models, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Furthermore, the dataset complexity is systematically varied—through changes in noise levels, signal distortions, and dataset sizes—to study how different conditions affect model training, generalization, and robustness. A full end-to-end pipeline is implemented, converting Morse input into text and subsequently into speech using text-to- speech (TTS) synthesis. This functionality supports users who depend on assistive technologies by transforming simple physical input into audible communication. The conversion from Morse code to speech facilitates practical, real-time communication and enables the system to serve both educational and accessibility-focused applications.

Previous approaches to Morse code conversion have largely been limited to text-based decoding. These systems relied on manual input or static datasets and were designed to display output on screens without audio feedback. As a result, users with visual or communication impairments received limited benefit, and manual decoding introduced delays and potential errors. The proposed system addresses these limitations by automating both decoding and audio output, offering a more inclusive and efficient solution.

2. PROPOSED METHOD

This work proposes an improved Morse code decoding system that not only converts Morse signals into text but also turns that text into clear, natural-sounding speech. By adding audio output, the system moves beyond traditional text-based decoding and makes communication easier and more accessible—especially for users who may rely on hearing messages, such as people with visual impairments or those needing discreet communication.

To train the system, synthetic Morse code datasets were created. These datasets include labeled examples that vary in complexity and size, helping the model and learns to recognize and decode Morse code accurately under different conditions. The input comes from a single-button device,

where the timing of presses is translated into intensity-based frames that represent dots, dashes, and pauses. After decoding the Morse input into text, the system automatically converts the text into speech using tools like Google Text-to-Speech (GTTS). The audio output focuses on clear pronunciation, natural tone, and proper pacing making sure the message is easy to understand. This makes the system practical not only for covert agents sending secret messages but also for users who need assistive communication technology.

For larger inputs, the system improves performance by optimizing how it processes the data during clustering. One challenge that was addressed is occasional decoding errors with very short inputs (like single words), for which a fallback default word was introduced to improve reliability. Performance tests showed that decoding Morse code to text takes longer than converting text to speech, especially for longer messages. However, the system consistently maintained high accuracy throughout. Overall, the method aims to be simple to use, reliable, and inclusive, providing a complete solution for translating Morse code.

Advantages of the Proposed Method:

- Makes Morse code communication more accessible by producing audio output, which benefits visually impaired users and those with communication difficulties.

- Minimizes human error by automating the decoding process, removing the need for manual interpretation.

- Speeds up the overall conversion from Morse code to words, supporting real-time communication, such as in secret operations.

- Uses carefully designed synthetic datasets with different difficulty levels, which helps improve decoding accuracy across varied Morse signals.

3. METHADOLOGY

The methodology adopted in this project focuses on building a robust Morse code translation system that can process single-button input, decode Morse sequences, and convert them into audio output using both K-means clustering and fuzzy logic techniques. This section outlines the structured pipeline of data acquisition, processing, clustering, and signal-to-text-to-audio conversion. To begin, the input is captured via a single-button interface, simulating the process of Morse code communication.

The system records two key features: the time duration of each button press and the interval between successive presses. These values are stored as two columns—denoted as X (gaps between presses) and Y (duration of presses)—in a CSV file for further processing. The keyboard and time libraries were used to monitor and record the button events, while pandas and csv modules organized the data systematically. The Google Text-to-Speech (GTTS) engine was integrated to synthesize the decoded output into audio, enabling auditory message delivery. Once the data is captured, the next phase involves decoding the signal from the CSV format into human-readable text and subsequently into speech. To classify the nature of each input, unsupervised machine learning—specifically, K-means clustering—was employed. The time gap column (X) is clustered into three categories: intra-character pause, inter- character pause, and inter-word pause. Similarly, the duration column (Y) is grouped into two clusters, representing dots and dashes. This step is crucial for accurately interpreting Morse sequences without prior labeling.

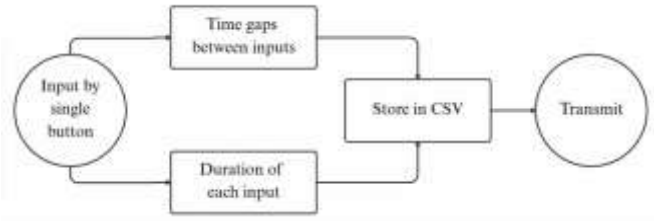

Fig.1- Architectural Diagram for Input Mechanism

In addition to K-means, fuzzy clustering was integrated to handle ambiguous or noisy inputs. Unlike traditional crisp clustering, fuzzy models assign degrees of membership to each cluster, making them suitable for cases with overlapping or uncertain data points. Fuzzy logic was applied using inference rules such as: “If length is SHORT and interval is SHORT, then the symbol is a dot” and “If length is LONG and interval is SHORT, then the symbol is a dash.” This approach allows for nuanced decoding, especially under conditions of signal distortion or inconsistency. To simulate and train models under real-world conditions, synthetic Morse code datasets were generated. Each Morse word was encoded into a one-dimensional frame of 64 values, with varying lengths for dots, dashes, and pauses.

X	Y		
0.439437	0.364921	0.280457	0.439203
1.214664	0.349808	1.605461	0.122652
0.160359	0.341395	0.077557	0.099557
0.333286	0.396302	0.243219	0.369079
0.930155	0.4568	1.26617	0.086739
0.116044	0.106879	1.775321	0.086739
0.154803	0.426991	0.278583	0.094365
1.308877	1.015337	0.132268	0.371013
0.100278	0.069934	0.231029	0.119728
0.17978	0.078022	1.14731	0.400273
1.12641	0.089816	0.945635	0.139784
2.598788	0.060222	0.065607	0.336089
0.298788	0.381522	1.484257	0.107623
0.254822	0.085594	0.265472	0.141302
0.177349	0.402255	0.244237	0.417716
0.225862	0.086671	1.356375	0.444395
1.333611	0.323673	0.30393	0.452026
0.229004	0.418517	0.092992	0.09292
0.284354	0.521365		
1.328338	0.089871		

Fig.2- Dataset Generated

Intensity values were assigned using normal distributions, and noise was introduced to simulate variations in signal transmission. Frames were normalized to [0, 1] range with three-decimal precision. The datasets were structured to vary in difficulty, allowing performance evaluation under multiple conditions such as signal distortion, class overlap, and sequence ambiguity.

The synthetic data generation pipeline involved the creation of labeled data for 64 classes, corresponding to different Morse symbols. To ensure robustness, variations were introduced by manipulating input lengths, signal intensity, and timing. These artificial datasets enabled rigorous training of neural network models, enhancing generalization across scenarios. Further, augmentation techniques like scaling and shifting were considered to improve model robustness.

Evaluation metrics such as classification accuracy and F1-score were identified to measure performance, with potential benchmarking against real-world Morse samples. This comprehensive methodology enables the development of a reliable and accessible Morse code decoding system. The integration of fuzzy logic and clustering, combined with dataset simulation and audio conversion, ensures that the model is not only technically sound but also inclusive for users with visual or communication impairments.

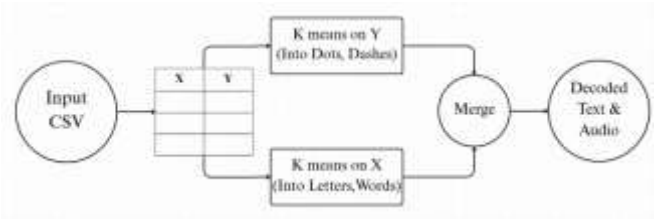

Fig 3: Architectural Diagram for conversion into text

4. RESULTS

A) Results after K-Means Clustering of Input Data

The Morse code decoding process begins with the application of K-means clustering on the two key input features: X (time gaps between inputs) and Y (duration of button presses). The X-column, which holds inter-press intervals, is categorized into three distinct clusters representing input boundaries, character boundaries, and word boundaries. This clustering enables the system to segment the continuous input into structured Morse code sequences.

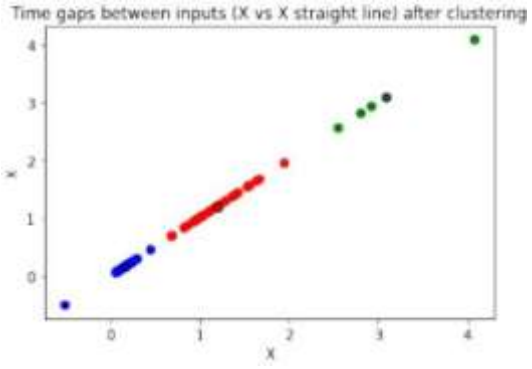

Fig.4 - X vs X graph after clustering

The output of K-means clustering is visualized using a Y vs. Y scatter plot, where color-coded points (e.g., Blue, Red, Green) represent the three identified clusters. This visualization offers intuitive insight into how the system differentiates between various boundary types using temporal features.

Similarly, clustering is applied to the Y-column to distinguish between dots and dashes. These are the fundamental components of Morse code, with dots having shorter durations and dashes longer ones. K-means efficiently categorizes the inputs into two clusters based on their duration, supporting accurate decoding of Morse elements. This classification is visualized using a simplified 1D plot (Y vs. Y), which offers a minimal yet effective representation of cluster boundaries using only a single feature.

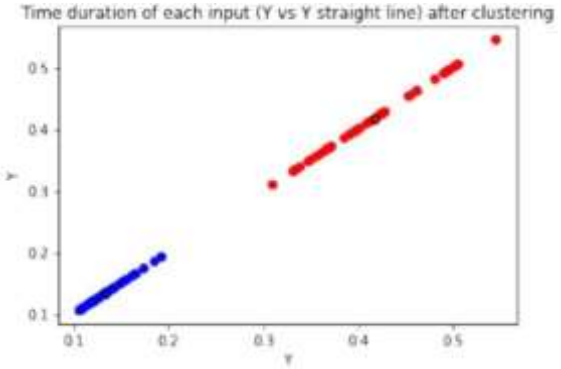

Fig 5: Y vs Y graph after clustering

This approach not only improves interpretability but also aids in identifying patterns and irregularities within the data. It simplifies debugging, optimization, and real-time adjustments when deployed in communication-sensitive scenarios such as covert operations or emergency services.

B) Results After Fuzzy Clustering of Input Data

The decoding process is further refined using fuzzy clustering, which provides a probabilistic interpretation of data rather than a hard assignment. Fuzzy logic enables each input to belong to multiple clusters with varying degrees of membership, providing a more flexible and realistic modeling of uncertain or noisy data—especially in cases with overlapping signal characteristics.

For the Y-column, fuzzy clustering classifies inputs as either dots or dashes, while also capturing intermediate variations that may arise due to transmission errors or inconsistent press durations. Inputs with short durations receive higher membership to the "dot" cluster, while longer durations are associated with the "dash" cluster.

Likewise, the X-column is fuzzy-clustered into three segments corresponding to input, character, and word boundaries. This flexible classification system captures the nuanced transitions between signal boundaries, enhancing decoding accuracy in cases of inconsistent input intervals

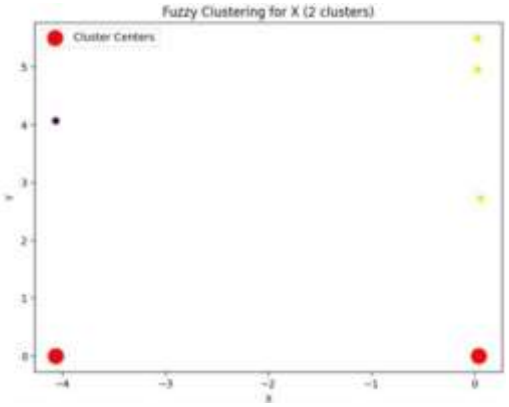

Fig 6: Y vs Y graph after clustering

Figures illustrating fuzzy clustering for both two and three clusters (for Y and X respectively) demonstrate how the fuzzy model accounts for ambiguity, providing a deeper understanding of signal patterns. This adaptability makes fuzzy clustering particularly advantageous in real-world scenarios where exact boundary definitions may not always be clear.

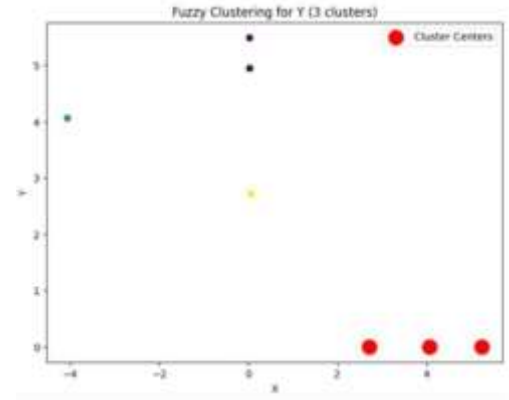

Fig 7: Clustering on X and Y for 3 clusters

Figure illustrating fuzzy clustering for both two and three clusters (for Y and X respectively) demonstrate how the fuzzy model accounts for ambiguity, providing a deeper understanding of signal patterns. This adaptability makes fuzzy clustering particularly advantageous in real-world scenarios where exact boundary definitions may not always be clear.

C) Final Output and Interpretation

Upon processing and decoding the clustered data, the final output from the system—derived from a sample CSV file—was accurately interpreted as the phrase:

"JSS Science and Technology University"

This text is then converted to audio using the Google Text- to-Speech (gTTS) engine.

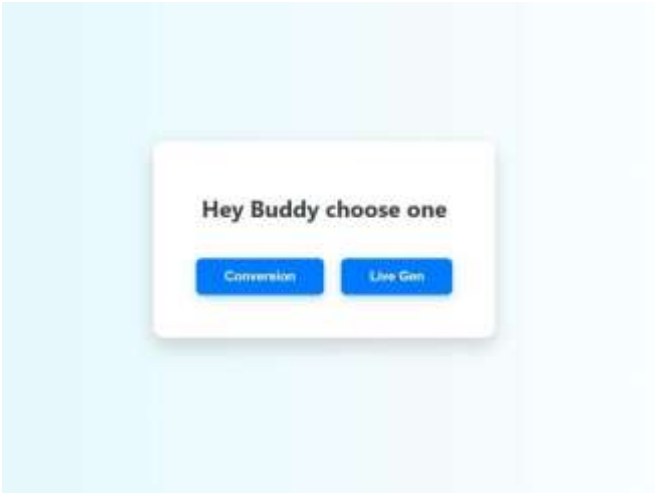

Fig 8: Entry Page

The system ensures clear pronunciation, appropriate intonation, and correct pacing for natural-sounding speech output. Such audio feedback is particularly valuable for visually impaired users or in scenarios requiring hands-free communication.

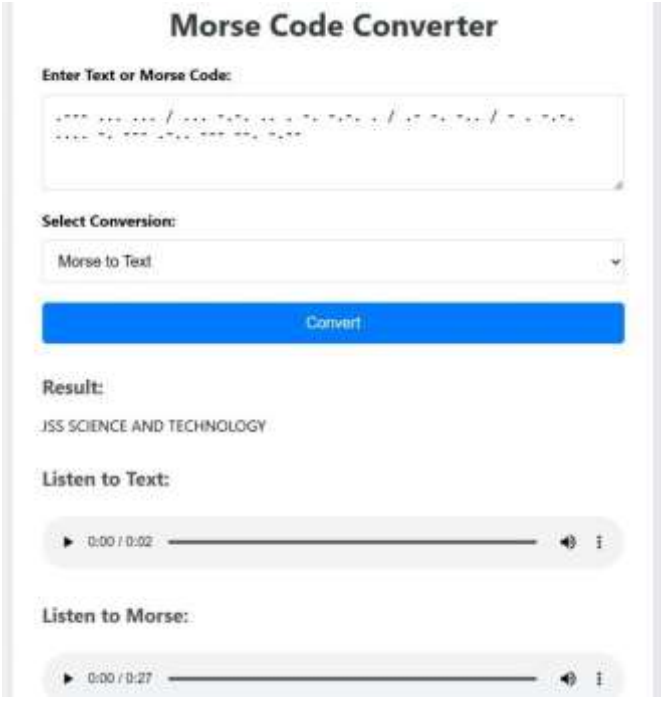

Fig 9: Morse to text

Fig 10: Single Button Input

D) Time Complexity Analysis

An analysis of the system's time complexity reveals that converting Morse code (CSV format) to text typically takes more time than converting text to audio. This is due to the iterative nature of the K-means clustering algorithm, especially when processing large datasets with many inputs. The clustering step involves multiple loop iterations to achieve convergence, making it the most time-intensive component of the pipeline.

Time taken for converting Morse to Text: 0.89333800000002 sec

Time taken for converting Text to Audio: 0.23341999999974 sec

Time taken of Execution (Morse to Audio): 1.126758 sec

Conversely, the gTTS engine processes text-to-audio conversion efficiently, following a standardized procedure. However, in cases where the input consists of only a single word, the clustering model may underperform due to a lack of sufficient data points. A proposed workaround is to embed a default fallback word—such as the agent's name—into each device, ensuring the model always processes at least two words, thereby stabilizing clustering accuracy.

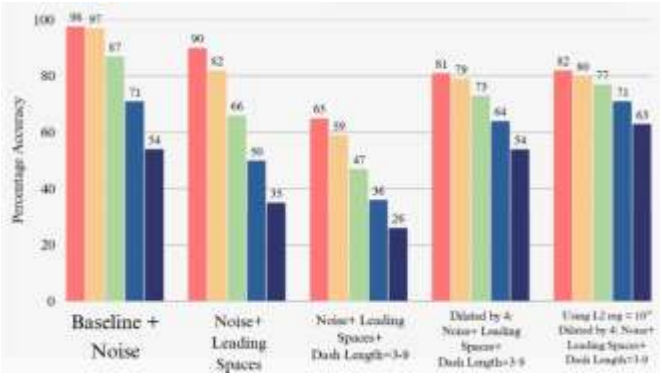

Fig 10: Percentage accuracies

The system demonstrated near-perfect decoding accuracy across a variety of input lengths and signal types. Furthermore, by automating the decoding and speech synthesis pipeline, the model minimizes human error, significantly enhancing efficiency over traditional manual Morse code interpretation methods.

5. CONCLUSION

This project successfully demonstrates the design and implementation of a Morse code decoding system that combines traditional communication methods with modern machine learning techniques, particularly K-means and fuzzy clustering. By converting single-button Morse code inputs into structured data, and then decoding them into both text and audio outputs, the system provides an accessible and efficient communication tool. The use of K-means clustering allowed the model to classify signal durations and gaps effectively into dots, dashes, letters, and words, while fuzzy clustering added flexibility by handling overlapping or ambiguous signal patterns. Together, these approaches enhanced the decoding accuracy, even in the presence of noise or signal variation.

In addition to decoding, synthetic Morse datasets were generated to simulate a wide range of signal intensities, frame lengths, and difficulty levels. These datasets enabled robust training and testing of the system under various real world scenarios. The conversion from Morse to audio— using tools such as Google Text-to-Speech (gTTS) ensured that the decoded messages could be delivered in a clear and human-like voice, improving usability for visually impaired users and those in covert operations. Time complexity analysis revealed that while Morse-to-text decoding took longer due to iterative clustering, the system remained highly accurate, with only minor limitations in single-word inputs.

Looking ahead, the project opens several promising avenues for development. Future work can focus on integrating more advanced machine learning models such as deep learning or ensemble methods, which could offer even higher accuracy by learning complex patterns within Morse sequences. Practical applications may include real-time decoding systems embedded in emergency response tools, wearable communication devices, or assistive technologies for individuals with speech or vision impairments. Further improvements may involve optimizing the model for faster processing, expanding the dataset with real-world noise profiles, or developing a user-friendly graphical interface. Hardware deployment on platforms like Raspberry Pi or custom embedded systems is also a feasible extension, making the solution portable and reliable in field scenarios.

Ultimately, this work bridges the gap between an age-old signaling method and the capabilities of modern AI. By transforming Morse code into a fully automated, audio-supported communication system, it not only preserves its historical significance but also reimagines its role in today's digital and inclusive communication landscape.

6. REFERENCES

[1] Q. Shanhu, L. Hongbo, and Z. Xu, "Morse Recognition Algorithm Based on K-means," 2019 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference (CSQRWC), Taiyuan, China, 2019, pp. 1–2, doi: 10.1109/CSQRWC.2019.8799149.

[2] B. J. Kranthi, G. Suhas, K. B. Varma, and G. P. Reddy, "A Two-Way Communication System with Morse Code Medium for People with Multiple Disabilities," 2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Prayagraj, India, 2020, pp. 1–6.

[3] S. M., N. Kolkar, S. G. S., and K. Kulkarni, "Morse Code Detector and Decoder Using Eye Blinks," 2021Third International Conference on Inventive Research in Computing

Applications (ICIRCA), Coimbatore, India, 2021, pp. 1–6, doi: 10.1109/ICIRCA51532.2021.9545039.
[4] S. Dey, K. M. Chugg, and P. A. Beerel, "Morse Code Datasets for Machine Learning," 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 2018, pp. 1–7, doi: 10.1109/ICCCNT.2018.8494011.
[5] C.-H. Yang, L.-C. Chin, and S.-C. Hsieh, "Morse Code Recognition Using Support Vector Machines," IEEE EMBS Asian-Pacific Conference on Biomedical Engineering, Kyoto, Japan, 2003, pp. 220–222.
[6] S. Dey, K. Chugg, and P. Beerel, "Morse Code Symbol Classification," IEEE Dataport, Jul. 14, 2020, doi: 10.21227/mam3-kp03.
[7] R. Li, M. Nguyen, and W. Q. Yan, "Morse Codes Enter Using Finger Gesture Recognition," in Proc. Int. Conf. Digital Image Computing: Techniques and Applications (DICTA), Nov. 2017.
[8] N. S. Bakde and A. P. Thakare, "Morse Code Decoder Using a PIC Microcontroller," International Journal of Science, Engineering and Technology Research (IJSETR), vol. 1, no. 5, 2012.
[9] N. Patki, R. Wedge, and K. Veeramachaneni, "The Synthetic Data Vault," in Proc. IEEE Int. Conf. Data Science and Advanced Analytics (DSAA), 2016, pp. 399–410. Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 2018, pp. 1–7, doi: 10.1109/ICCCNT.2018.8494011.
[10] C.-H. Yang, L.-C. Chin, and S.-C. Hsieh, "Morse Code Recognition Using Support Vector Machines," IEEE EMBS Asian-Pacific Conference on Biomedical Engineering, Kyoto, Japan, 2003, pp. 220–222.
[11] S. Dey, K. Chugg, and P. Beerel, "Morse Code Symbol Classification," IEEE Dataport, Jul. 14, 2020, doi: 10.21227/mam3-kp03.
[12] R. Li, M. Nguyen, and W. Q. Yan, "Morse Codes Enter Using Finger Gesture Recognition," in Proc. Int. Conf. Digital Image Computing: Techniques and Applications (DICTA), Nov. 2017.
[13] N. S. Bakde and A. P. Thakare, "Morse Code Decoder Using a PIC Microcontroller," International Journal of Science, Engineering and Technology Research (IJSETR), vol. 1, no. 5, 2012.
[14] N. Patki, R. Wedge, and K. Veeramachaneni, "The Synthetic Data Vault," in Proc. IEEE Int. Conf. Data Science and Advanced Analytics (DSAA), 2016, pp. 399–410