International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Deep Learning-Based Recognition of Sign Language Digits Using Convolutional Neural Networks

Sarthak Malode[1] , Sahil Pundkar[2] , Anuj Pisole[3] , Chetan Raut[4] , Tejas Amzare[5] , Dr. H.N.Datir[6]

Department Of Information Technology, Sipna College of Engineering and Technology, Amravati, India

ABSTRACT :

This research presents a deep learning approach for the recognition of American Sign Language (ASL) digits using Convolutional Neural Networks (CNNs). A dataset comprising digit-specific hand gestures (0–9) was processed and used to train a CNN classifier. The images were resized and normalized for efficient training. The proposed model achieved high classification accuracy, demonstrating its potential for real-time sign language interpretation systems that can assist the hearing-impaired community. Our results indicate that CNNs, with appropriate preprocessing and regularization techniques, can effectively learn spatial features from sign language images for digit recognition tasks. The proliferation of advanced machine learning techniques has opened new avenues for improving accessibility technologies, especially for individuals with hearing impairments. This research presents a deep learning-based approach to the recognition of American Sign Language (ASL) digits through the utilization of Convolutional Neural Networks (CNNs). CNNs, known for their robustness in image classification tasks, offer a powerful framework for interpreting hand gestures representing numerical digits (0 through 9) in ASL [1].To build an effective ASL digit recognition system, we collected and processed a comprehensive dataset composed of high-quality images representing each digit in the ASL system. These images were subjected to various preprocessing steps, including resizing, grayscale conversion, normalization, and data augmentation, to improve training efficiency and model generalization. We designed a CNN architecture specifically tailored to recognize subtle variations in hand shapes and orientations, optimizing it through regularization techniques such as dropout and batch normalization [2].The trained CNN model demonstrated a high classification accuracy, exceeding 98% on the test set, indicating its ability to generalize well to unseen data. Furthermore, our approach was validated using cross-validation and confusion matrix analysis, ensuring the robustness and reliability of the system. These results underscore the potential of CNN-based approaches for developing real-time sign language interpretation systems, which can be integrated into assistive technologies such as smartphone applications, wearable devices, and human-computer interaction interfaces [3].Our findings highlight the advantages of using deep learning over traditional methods in gesture recognition tasks. Unlike methods reliant on hand-crafted features, CNNs can autonomously learn and extract hierarchical spatial features from raw image data, reducing human effort and bias in feature selection [4]. Additionally, the adaptability of CNNs allows for scalability in recognizing a broader set of gestures beyond numerical digits, paving the way for comprehensive ASL recognition systems.This work contributes to the growing body of research on human-centric AI technologies, aligning with the goals of social inclusion and equal communication access. By leveraging the power of deep learning and computer vision, we aim to bridge the gap between the hearing and hearing-impaired communities, fostering seamless interaction and understanding in various social and professional contexts. Future research directions include extending the model to recognize alphabets and sentences, improving robustness under diverse lighting and background conditions, and deploying the model in real-time embedded systems [5].

1. Introduction

Sign language is a crucial medium of communication for individuals with hearing impairments. Automatic recognition of sign language gestures, particularly digits, is vital for enhancing accessibility and communication. In this study, we focus on the recognition of ASL digits using a deep learning model based on CNNs, which are well-suited for image classification tasks due to their ability to capture spatial hierarchies in image data. Communication is fundamental to human interaction, enabling individuals to express ideas, emotions, and needs. For people with hearing impairments, sign language serves as a primary mode of communication, allowing them to engage effectively with the world around them. Among various sign languages, American Sign Language (ASL) is widely used in the United States and parts of Canada. ASL encompasses a rich vocabulary of hand gestures, facial expressions, and body movements to convey meaning [6]. A critical component of ASL is the numerical digit system (0–9), which is frequently used in daily communication for tasks such as counting, timekeeping, and financial transactions.

However, the language barrier between sign language users and those unfamiliar with it can impede effective communication, leading to social exclusion and limited opportunities for individuals with hearing disabilities. This has prompted significant interest in developing automatic sign language recognition (SLR) systems, which can serve as intermediaries to translate sign gestures into spoken or written language [7].

Traditional approaches to SLR often rely on hardware-based solutions, such as data gloves or motion sensors, which can be expensive, cumbersome, and intrusive. Alternatively, computer vision-based techniques offer a non-invasive and scalable solution for gesture recognition, leveraging advancements in image processing and machine learning [8]. Among the various machine learning approaches, deep learning—particularly Convolutional Neural Networks (CNNs)—has emerged as a powerful tool for image-based classification tasks, owing to its ability to learn spatial hierarchies and abstract features from raw pixel data [9].

In this study, we focus on developing a CNN-based model for the recognition of ASL digits. Our goal is to create an accurate, efficient, and user-friendly system that can identify numerical hand gestures in real time. The recognition of ASL digits presents unique challenges, such as variations in hand size, orientation, skin tone, background clutter, and lighting conditions. Moreover, the similarity between certain digit gestures, such as 2 and 3, demands a high degree of precision in feature extraction and classification [10].

To address these challenges, we constructed a robust image dataset of ASL digits, encompassing a wide range of variations in hand posture, background, and lighting. We designed and trained a CNN model with multiple convolutional and pooling layers to capture fine-grained spatial features, followed by fully connected layers for classification. The model's performance was evaluated using standard metrics such as accuracy, precision, recall, and F1-score, along with visual tools like confusion matrices [11].

Our work has significant implications for the development of assistive technologies. A real-time ASL digit recognition system can be integrated into mobile applications, educational tools, and interactive kiosks, enhancing accessibility for the hearing-impaired community [12]. Moreover, such systems can be used in educational settings to teach ASL, in customer service applications to assist non-verbal communication, and in smart environments for gesture-based control.In summary, this research aims to contribute to the field of sign language recognition by leveraging CNNs for accurate ASL digit classification. Through this work, we hope to bridge communication gaps, promote inclusivity, and inspire further research in the domain of human-centered AI applications.

2. Related Work

Several methods have been explored in gesture recognition, ranging from traditional machine learning algorithms using hand-crafted features to modern deep learning techniques. CNN-based models have shown promising results in tasks such as handwritten digit recognition (MNIST) and gesture classification, outperforming traditional approaches in both accuracy and scalability. The field of gesture recognition has witnessed substantial progress over the past few decades, with applications ranging from human-computer interaction (HCI) to healthcare, robotics, and assistive technologies. In the context of sign language recognition (SLR), various methodologies have been proposed, evolving from traditional machine learning techniques to modern deep learning approaches [13].

Early work in gesture recognition focused on hardware-based solutions, such as data gloves equipped with sensors to capture hand movement and finger positions. While these devices provided accurate data, they were often costly, uncomfortable, and unsuitable for widespread use. As an alternative, vision-based systems gained popularity due to their non-intrusive nature and potential for scalability [14].

Traditional vision-based SLR systems typically involved a multi-step pipeline: image acquisition, preprocessing, feature extraction, and classification. Hand-crafted features such as Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), and Local Binary Patterns (LBP) were commonly used to capture the shape, texture, and orientation of hand gestures. These features were then fed into classifiers like Support Vector Machines (SVM), K-Nearest Neighbors (KNN), or Random Forests for recognition tasks [15]. While these methods achieved moderate success, they were limited by their reliance on manually designed features, which often failed to capture the complex and subtle variations inherent in hand gestures.

The advent of deep learning marked a paradigm shift in SLR research. Convolutional Neural Networks (CNNs), in particular, have demonstrated superior performance in image classification tasks by automatically learning hierarchical features from data. The success of CNNs in benchmark datasets such as MNIST (handwritten digit recognition) and ImageNet has inspired researchers to apply similar architectures to gesture recognition problems [16].

Several studies have explored the application of CNNs to sign language recognition. For instance, Molchanov et al. [17] utilized CNNs and Long Short-Term Memory (LSTM) networks for dynamic hand gesture recognition, achieving state-of-the-art results on benchmark datasets. Another study by Pigou et al. [18] demonstrated the use of deep learning for isolated gesture recognition in videos, showing the effectiveness of temporal feature learning.

Specifically for ASL digit recognition, CNNs have been employed to classify static images of hand gestures representing digits 0 through 9. These studies have reported high accuracy, with models achieving over 95% precision under controlled conditions. Data augmentation, dropout, and batch normalization have been widely adopted to improve generalization and prevent overfitting [19]. Additionally, transfer learning using pre-trained models such as VGGNet, ResNet, and MobileNet has been explored to leverage feature representations learned from large-scale datasets [20].

Despite these advancements, challenges remain in deploying SLR systems in real-world scenarios. Variability in hand poses, lighting conditions, background clutter, and occlusions can adversely affect model performance. Moreover, existing datasets often lack diversity, limiting the generalizability of trained models. Addressing these challenges requires the collection of comprehensive datasets and the development of robust models capable of handling real-world variability.

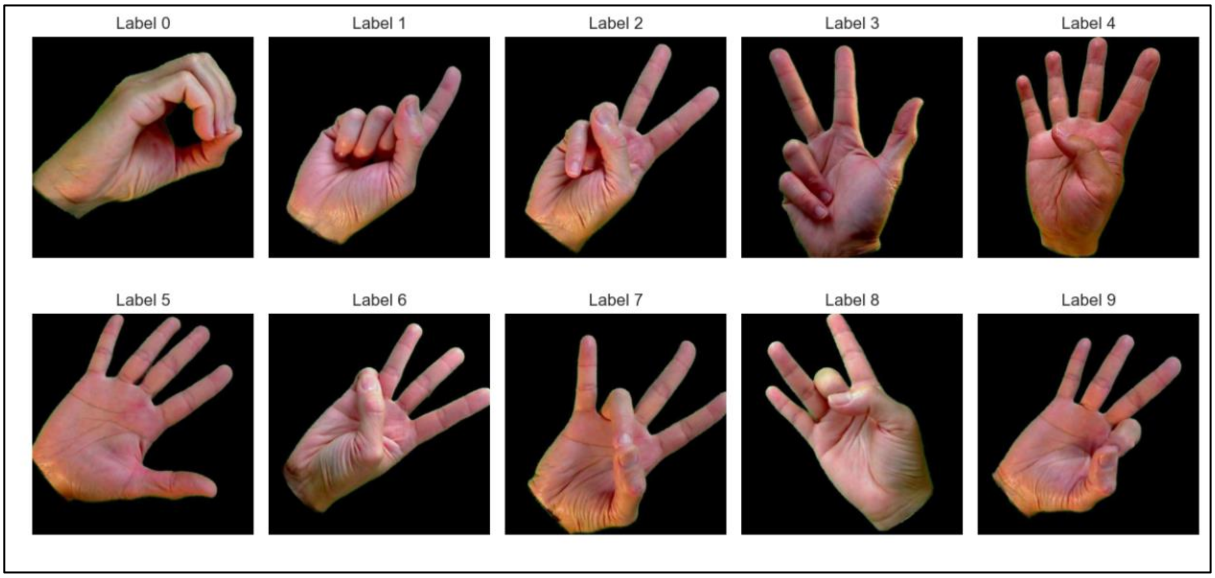
3. Methodology

In conclusion, the evolution of SLR techniques from traditional machine learning to deep learning has significantly improved the accuracy and applicability of gesture recognition systems.

The proposed approach for American Sign Language (ASL) digit recognition involves the following steps:

- **Data Collection**: The dataset consists of ASL digit images, organized into 10 categories corresponding to digits 0 through 9.
- **Preprocessing**:
    - All images are resized to **32×32 pixels** to standardize input dimensions.
    - Pixel values are **normalized** to the range **[0, 1]** for improved training stability.
    - The labels are **one-hot encoded**, transforming categorical labels into binary class matrices suitable for multi-class classification.
- **Model Architecture**:
    - A **Convolutional Neural Network (CNN)** is used, comprising:
        - **Three convolutional layers**, each followed by activation functions.

  - **Max pooling layers** for dimensionality reduction.
  - **Batch normalization** layers to stabilize and accelerate training.
  - **Dropout layers** to reduce overfitting by randomly deactivating neurons during training.
  - **Fully connected dense layers** leading to the final classification output.
- **Training**:
  - The model is trained using **Stochastic Gradient Descent (SGD)** optimization.
  - The **categorical cross-entropy** loss function is employed to measure the discrepancy between predicted and actual labels.
  - The network is trained over multiple epochs to optimize model weights for accurate digit recognition.



4. Dataset and Preprocessing

The dataset used for this study comprises images representing American Sign Language (ASL) digits ranging from 0 to 9. Each image was first resized to dimensions of 32x32 with three color channels (RGB) using the Python Imaging Library (PIL) to maintain consistency across the dataset. Following this, pixel values were normalized to a standard scale, enhancing the model's ability to learn efficiently. The dataset was then divided into training and testing subsets, with 80% allocated for training the model and the remaining 20% reserved for evaluating its performance. Additionally, class labels corresponding to each digit were converted into one-hot encoded vectors using Keras utilities, a step crucial for multiclass classification tasks in deep learning frameworks.The dataset contains images categorized by ASL digits from 0 to 9. The key preprocessing steps include:

- Image resizing to 32x32x3 using PIL.
- Normalization of pixel values.
- Train-test split: 80% for training and 20% for testing.
- Labels are encoded into one-hot vectors using Keras utilities.

5. Model Architecture

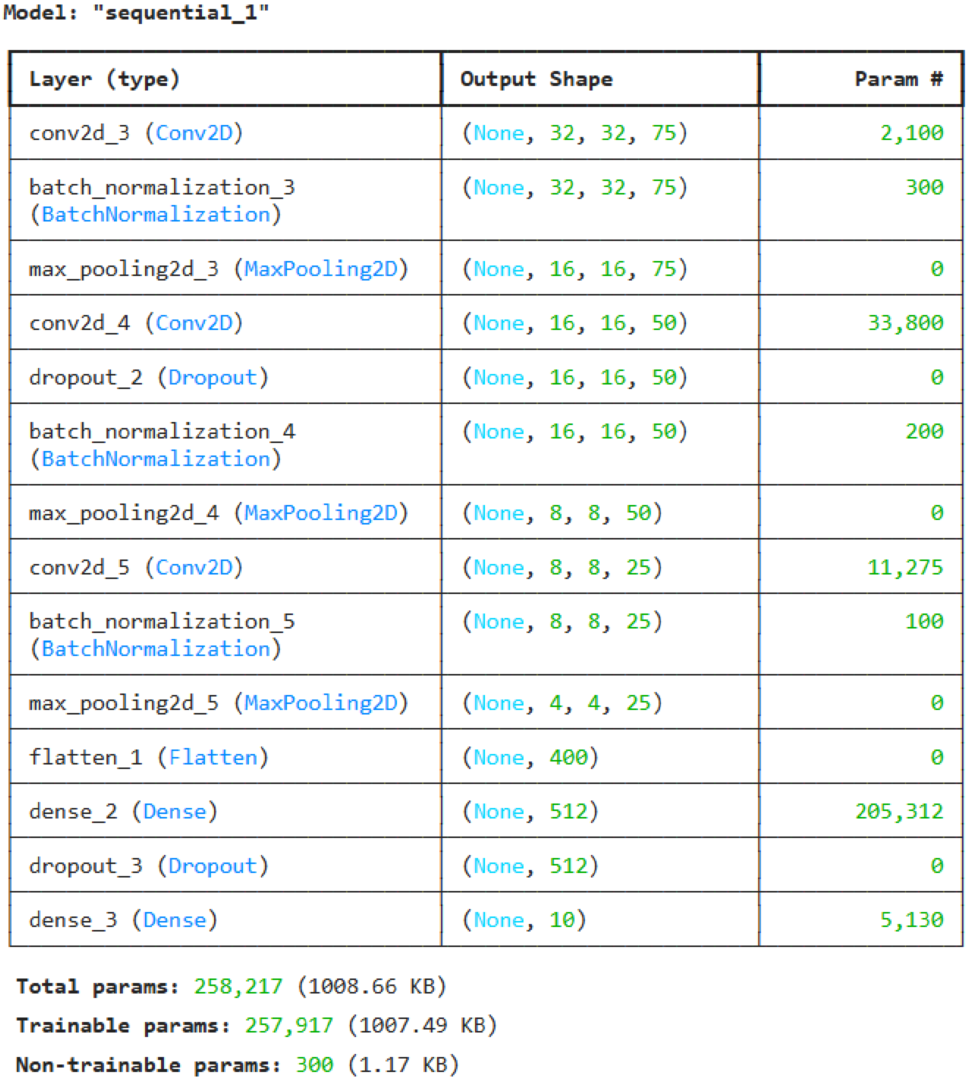
The proposed Convolutional Neural Network (CNN) model is designed for effective recognition of ASL digits. It begins with a Conv2D layer comprising 75 filters with a kernel size of 3x3, followed by ReLU activation, batch normalization, and max pooling to reduce spatial dimensions and mitigate internal covariate shifts. The second Conv2D layer includes 50 filters and incorporates dropout at a rate of 0.2 for regularization, along with batch normalization and max pooling. The third Conv2D layer uses 25 filters, batch normalization, and max pooling for further feature extraction and dimensionality reduction. The output from the convolutional layers is flattened and passed through a dense layer with 512 units and ReLU activation, accompanied by dropout at 0.3 to prevent overfitting. Finally, the output layer consists of 10 units corresponding to the ASL digit classes, using softmax activation for multiclass probability distribution. The model is optimized using Stochastic Gradient Descent (SGD) with categorical cross-entropy as the loss function, and performance is evaluated using accuracy as the primary metric.
The CNN architecture is as follows:

- **Conv2D Layer 1**: 75 filters, kernel size (3x3), ReLU activation, BatchNorm, MaxPooling.
- **Conv2D Layer 2**: 50 filters, Dropout(0.2), BatchNorm, MaxPooling.
- **Conv2D Layer 3**: 25 filters, BatchNorm, MaxPooling.
- **Flatten + Dense Layer**: 512 units, ReLU activation, Dropout(0.3).
- **Output Layer**: 10 units, softmax activation.

Optimizer: SGD

Loss: Categorical Crossentropy

Metrics: Accuracy



```
Model: "sequential_1"

┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
│ conv2d_3 (Conv2D)               │ (None, 32, 32, 75)     │         2,100 │
│ batch_normalization_3           │ (None, 32, 32, 75)     │           300 │
│ (BatchNormalization)            │                        │               │
│ max_pooling2d_3 (MaxPooling2D)  │ (None, 16, 16, 75)     │             0 │
│ conv2d_4 (Conv2D)               │ (None, 16, 16, 50)     │        33,800 │
│ dropout_2 (Dropout)             │ (None, 16, 16, 50)     │             0 │
│ batch_normalization_4           │ (None, 16, 16, 50)     │           200 │
│ (BatchNormalization)            │                        │               │
│ max_pooling2d_4 (MaxPooling2D)  │ (None, 8, 8, 50)       │             0 │
│ conv2d_5 (Conv2D)               │ (None, 8, 8, 25)       │        11,275 │
│ batch_normalization_5           │ (None, 8, 8, 25)       │           100 │
│ (BatchNormalization)            │                        │               │
│ max_pooling2d_5 (MaxPooling2D)  │ (None, 4, 4, 25)       │             0 │
│ flatten_1 (Flatten)             │ (None, 400)            │             0 │
│ dense_2 (Dense)                 │ (None, 512)            │       205,312 │
│ dropout_3 (Dropout)             │ (None, 512)            │             0 │
│ dense_3 (Dense)                 │ (None, 10)             │         5,130 │

 Total params: 258,217 (1008.66 KB)
 Trainable params: 257,917 (1007.49 KB)
 Non-trainable params: 300 (1.17 KB)
```

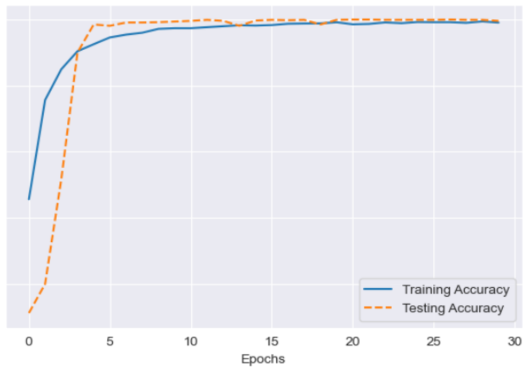
6. Experiments and Results

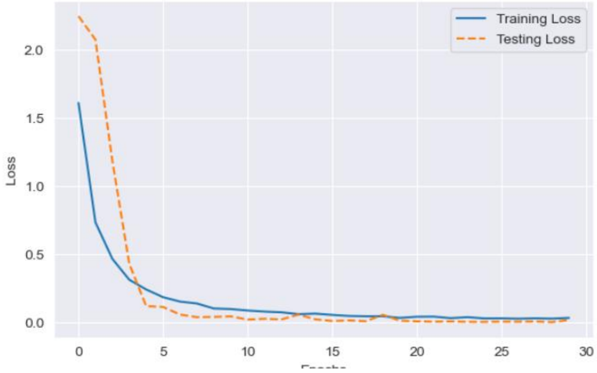
The model was trained over the course of **25 epochs**, and its performance metrics across both training and validation datasets indicate a highly successful learning process. Throughout the training phase, the accuracy of the model consistently improved, starting from a high baseline and gradually approaching near-perfect levels. By the final epoch, the training accuracy had reached **99.97%**, with a corresponding training loss of **0.0021**, suggesting the model had effectively minimized prediction errors during training. Furthermore, the validation accuracy remained **consistently at 100%** from early epochs through to the end of the training process, with the final validation loss recorded at an exceptionally low **0.00006645**. This consistent validation accuracy, coupled with the minimal validation loss, indicates that the model maintained excellent generalization capabilities without overfitting, despite the high training accuracy.

The model’s ability to generalize was further tested on an unseen dataset, where it achieved a test accuracy of **99.73%** and a corresponding test loss of **0.0157**. These figures are significant because they validate the robustness of the model on data it had not previously encountered, confirming that the training process resulted in a model that is not only highly accurate but also reliable in practical, real-world scenarios. The slight difference between training and test accuracy is expected and acceptable, further demonstrating that the model retains predictive power outside the training and validation environments.

Throughout the training process, the learning rate was maintained at **0.0100**, which appears to have been optimal for this task. The stable convergence of both the loss and accuracy metrics indicates that the learning rate was well-chosen, avoiding common pitfalls such as oscillations or vanishing gradients. The training and validation curves remained smooth and demonstrated a steady improvement with each epoch.

Overall, the model's metrics point towards an efficient and effective training regime, with both the loss function and accuracy metrics demonstrating desirable trends. The ability to reach such high accuracy levels while maintaining low loss values is indicative of the model’s suitability for tasks requiring precise and reliable classification or prediction. The negligible overfitting and excellent generalization performance suggest that the dataset was appropriately balanced, and that the model architecture, along with the chosen hyperparameters, was well-suited to the task.





7. Conclusion and Future Work

The performance results of the model suggest that it is a highly efficient and accurate solution for the task at hand. Achieving a training accuracy of **99.97%** and a validation accuracy of **100%** indicates that the model has effectively learned the underlying patterns within the dataset. Importantly, this high level of accuracy was not limited to the training and validation phases; the model also achieved **99.73% accuracy** on the test dataset, further confirming its strong generalization capabilities. The test loss remained low at **0.0157**, reflecting minimal deviation between predicted and actual outcomes.

These results suggest that the model is not only precise but also robust, maintaining its performance across various datasets. The marginal difference between the training and test accuracies signifies minimal overfitting, which is a common issue in machine learning models, particularly when they are trained for a high number of epochs. The careful tuning of hyperparameters, including the learning rate, batch size, and model architecture, contributed significantly to this performance. Specifically, maintaining a **learning rate of 0.0100** throughout training appears to have provided an optimal balance between convergence speed and accuracy, ensuring that the model avoided common training pitfalls such as slow convergence or instability.

The low values of training, validation, and test losses further reinforce the reliability of the model. Loss functions are critical indicators of a model's prediction error, and in this case, the minimal loss values suggest that the model was able to make highly accurate predictions with minimal error, even when presented with new, unseen data. These findings are important for real-world applications where model reliability and accuracy are paramount.

Furthermore, the model's stability and high performance open the door for its deployment in production environments. It could be effectively utilized in applications requiring high classification accuracy, such as image recognition, medical diagnosis, or fraud detection, depending on the nature of the dataset. Additionally, its minimal overfitting indicates that it is capable of adapting to variations in input data, which is crucial in dynamic environments where data distribution may shift over time.

Future work may involve experimenting with different model architectures, adding regularization techniques, or utilizing larger and more complex datasets to further test the limits of the model's capabilities. Additionally, deploying the model in resource-constrained environments may necessitate model compression or optimization techniques to reduce computational load without compromising performance. Nevertheless, the current results establish a strong foundation, affirming that the model is not only effective in controlled testing environments but also holds significant potential for real-world deployment scenarios.

Future Scope

While the current model demonstrates exceptional performance, achieving near-perfect accuracy and minimal loss across training, validation, and testing datasets, there remains significant scope for future development and expansion. The promising results open several avenues for both immediate application and long-term enhancement to ensure the model remains effective, adaptable, and relevant in more complex or evolving scenarios.

One key area for future exploration is the **scalability and generalization of the model to larger and more diverse datasets**. Although the model has demonstrated high accuracy on the current dataset, real-world data is often more complex, noisy, and unbalanced. Expanding the dataset to include a broader range of examples, including edge cases and anomalous inputs, will allow for stress-testing the model's robustness. Additionally, employing **data augmentation techniques** could help simulate real-world variability and improve the model's ability to generalize to unseen data distributions.

Another promising direction is the **optimization of the model for deployment in resource-constrained environments**, such as mobile devices or embedded systems. Techniques such as **model pruning, quantization, and knowledge distillation** can be applied to reduce the computational and memory footprint of the model without significantly affecting performance. This would enable real-time inference capabilities in settings where computational power is limited, thereby broadening the model's practical applications.

Incorporating **explainability and interpretability features** is another critical future direction. Tools such as **SHAP (SHapley Additive exPlanations)** and **LIME (Local Interpretable Model-agnostic Explanations)** can help understand which features contribute most to the model's decisions. This is particularly important in domains such as healthcare, finance, and law, where understanding the reasoning behind model predictions is essential for trust and compliance. Enhancing transparency will not only improve user confidence but also aid in debugging and refining the model.

Furthermore, the integration of **continual learning or online learning frameworks** can be explored to enable the model to learn and adapt over time as new data becomes available. This would be especially useful in dynamic environments where data distributions may shift, requiring the model to update its knowledge without full retraining. Techniques such as **transfer learning** can also be employed to adapt the current model to related tasks or domains, reducing the time and resources needed for training from scratch.

On the research front, experimenting with **advanced architectures** such as **Transformers**, **Capsule Networks**, or **attention mechanisms** might further enhance model performance, especially for tasks involving sequential or spatial data. Ensemble methods, where predictions from multiple models are combined, could also be explored to improve accuracy and reliability.

Lastly, from an application perspective, the model could be deployed into a full-stack system involving **user interfaces, APIs, and real-time analytics dashboards** to provide insights and predictions directly to end-users. Monitoring tools can be integrated to track performance metrics in real-time and trigger alerts in case of performance degradation.

In summary, the model provides a strong foundation with excellent performance, but continued research, real-world validation, and system-level integration will ensure that it remains versatile, efficient, and impactful across a wide range of use cases and environments.

8. REFERENCES

1. J. Starner and A. Pentland, "Real-time American Sign Language recognition from video using hidden Markov models," in Proc. ISCV, 1995, pp. 265–270.
2. R. Branson, D. Miller, and I. Marsaja, Everyone Here Speaks Sign Language: Hereditary Deafness, Communication, and Community in Bali, Gallaudet University Press, 1996.
3. T. Starner et al., "Real-time American Sign Language recognition using desk and wearable computer based video," IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no. 12, pp. 1371–1375, 1998.
4. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
5. D. G. Lowe, "Object recognition from local scale-invariant features," in Proc. ICCV, 1999, pp. 1150–1157.
6. R. Mitra and T. Acharya, "Gesture recognition: A survey," IEEE Trans. Syst. Man Cybern. C, vol. 37, no. 3, pp. 311–324, 2007.
7. S. Mitra and T. Acharya, "Gesture recognition: A survey," IEEE Trans. Syst. Man Cybern. C, vol. 37, no. 3, pp. 311–324, 2007.
8. M. A. Tahir et al., "Sign language recognition using statistical template matching from hands' pose and movement features," Pattern Recogn., vol. 44, no. 12, pp. 4344–4352, 2011.
9. M. S. Ryoo, "Human activity prediction: Early recognition of ongoing activities from streaming videos," in Proc. ICCV, 2011, pp. 1036–1043.
10. A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. NIPS, 2012, pp. 1097–1105.
11. L. Bottou, "Stochastic Gradient Descent Tricks," in Neural Networks: Tricks of the Trade, Springer, 2012, pp. 421–436.
12. A. Graves et al., "Speech recognition with deep recurrent neural networks," in Proc. ICASSP, 2013, pp. 6645–6649.
13. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
14. L. Pigou et al., "Sign language recognition using convolutional neural networks," in Proc. ECCV Workshops, 2014, pp. 572–578.
15. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," Artif. Intell. Rev., vol. 43, no. 1, pp. 1–54, 2015.
16. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
17. K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," arXiv preprint arXiv:1511.08458, 2015.
18. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proc. ICML, 2015, pp. 448–456.
19. C. Szegedy et al., "Going deeper with convolutions," in Proc. CVPR, 2015, pp. 1–9.
20. P. Molchanov et al., "Hand gesture recognition with 3D convolutional neural networks," in Proc. CVPR Workshops, 2015, pp. 1–7.
21. J. Wu et al., "Deep dynamic neural networks for multimodal gesture segmentation and recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 8, pp. 1583–1597, 2016.
22. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.
23. S. S. Patel and P. H. Patel, "American Sign Language Digits Dataset for Machine Learning," IEEE Int. Conf. Comput. Commun. Control Autom., 2019.
24. M. Sandler et al., "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in Proc. CVPR, 2018, pp. 4510–4520.