Dean&Francis

ISSN 2959-6157

# A Comparative Study of Deep CNN Architectures for Static American Sign Language Recognition

**Chi Zhang**

Department of Electrical and Electronic Engineering, the University of Melbourne, Melbourne, Australia
zhcz4@student.unimelb.edu.au

**Abstract:**

The goal of Sign Language Recognition (SLR), a crucial computer vision job, is to automatically understand sign motions in order to lower communication barriers between the hearing and deaf communities. Despite advances in deep learning, achieving high accuracy and deployment efficiency in real-world SLR systems remains challenging. In this work, a comparative analysis of four Convolutional Neural Network (CNN) architectures—Custom CNN, ResNet-50, EfficientNet-B0, and Inception-V3—for static American Sign Language (ASL) fingerspelling classification was performed. Using pre-trained models, this study applied data augmentation, transfer learning, and fine-tuning to the ASL Alphabet dataset, which comprises more than 87,000 images in 29 classifications. All models are trained with consistent protocols using PyTorch, including early stopping and learning rate scheduling. The results show that EfficientNet-B0 achieved the highest accuracy of 99.8% with minimal misclassifications, outperforming ResNet-50 (99.6%) and the Custom CNN (99.2%). Inception-V3 performed substantially worse, with 84.3% accuracy and a noisier confusion matrix, indicating more errors in distinguishing similar gestures. Confusion matrices confirmed that EfficientNet-B0 and ResNet-50 produced highly reliable, nearly diagonal predictions. The Custom CNN, while slightly less accurate, offered a lightweight baseline. These findings demonstrate the benefits of transfer learning and contemporary model scaling strategies in attaining high ASL identification accuracy, while also emphasizing the necessity of striking a balance between accuracy and computing efficiency for real-time deployment in real-world applications.

**Keywords:** Sign language recognition; deep learning; convolutional neural networks.

## 1. Introduction

Sign language is a vital communication tool for the deaf and hard-of-hearing cultures. It functions as a gesture and visual medium that allows people to communicate intricate ideas, feelings, and intentions. However, due to the limited number of hearing individuals proficient in sign language, communication barriers persist between deaf and hearing populations. Bridging this gap remains a pressing social need, and recent advances in artificial intelligence have opened up promising possibilities for automatic Sign Language Recognition (SLR) systems.

Deep learning has significantly changed the area of computer vision in the last ten years. Particularly impressive results have been obtained by Convolutional Neural Networks (CNNs) in tasks including gesture recognition, object identification, and picture categorization. Given their capacity to extract hierarchical spatial characteristics from pictures, CNN-based models have shown themselves to be particularly effective in the setting of SLR. Early studies demonstrated the feasibility of using shallow CNNs to recognize static signs representing the American Sign Language (ASL) alphabet. Pigou et al. introduced one of the first CNN-based systems for classifying isolated sign language images, establishing a baseline for static ASL recognition [1]. Molchanov et al. extended these approaches by employing 3D CNNs to capture spatio-temporal features in hand gestures, demonstrating the potential of deep learning for dynamic gesture recognition [2]. Kopuklu et al. further advanced the field by designing real-time hand gesture classification models optimized for deployment efficiency [3]. Subsequent research built on these foundations by adopting deeper and more sophisticated network architectures. Koller applied sequence modeling and deep feature extraction to improve the recognition of continuous sign language data, highlighting the advantages of combining spatial and temporal information [4]. Zhao et al. proposed SL-ResNet, which leverages residual learning techniques to achieve higher accuracy in static sign recognition, demonstrating the effectiveness of modern deep CNN architectures for SLR tasks [5].

Previous research has explored individual deep learning models for SLR, but comprehensive comparative studies across modern architectures remain limited. Furthermore, in addition to classification accuracy, few studies include system performance factors like computation efficiency and inference latency, which are crucial for real-time applications. For sign language recognition systems to be accurate and practicable, it is essential to comprehend the trade-offs between model complexity and deployment feasibility.

Four deep learning models for the static classification of the ASL alphabet are being compared in this work. The chosen architecture includes three cutting-edge pre-trained models: ResNet-50, EfficientNet-B0, and Inception-V3, in addition to a specially created CNN. A consistent training and assessment process is used to assess each model's classification accuracy, generalization performance, and computing efficiency. Future real-time sign language recognition systems are to be developed using the approach and results of this work.

## 2. Method

This section outlines the thorough process used to assess how well four CNN models performed on the ASL gesture detection challenge. The methodology is divided into three main parts: the dataset and preprocessing pipeline, the architecture of each model, and the experimental setup including training details and evaluation metrics.

### 2.1 Dataset and Preprocessing

#### 2.1.1 Dataset description

The dataset used in this study is the "ASL Alphabet" dataset, which is publicly available on Kaggle [6]. About 87,000 RGB pictures with a 200x200 pixel resolution make up this collection. The pictures are divided into 29 classes that include the 26 English alphabetic letters as well as three extra signs: „space," „delete," and „nothing." Each class is stored in a separate folder, and each folder contains around 3,000 images. The images were captured under consistent lighting conditions with uncluttered backgrounds, making this dataset well-suited for supervised image classification tasks.

In order to preserve the class balance across training and validation sets, the dataset was split using a stratified sampling approach. The training set contained 80% of the photos, with the remaining 20% left aside for validation. This split was applied consistently across all models in order to ensure a fair comparison.

#### 2.1.2 Preprocessing pipeline

To make the images compatible with different model architectures and to enhance model generalization, a standardized preprocessing pipeline was implemented. Every picture was adjusted to fit the input dimensions that each model needed. Images were specifically reduced to 299×299 pixels for Inception-V3 and 224×224 pixels for the Custom CNN, ResNet-50, and EfficientNet-B0 models.

The training pictures were subjected to a number of data augmentation approaches in order to avoid overfitting and enhance the models' capacity to generalize new data.

ISSN 2959-6157

These featured horizontal flipping, random rotations of up to ±15 degrees, and color jittering to create small changes in brightness, contrast, and saturation.

Following augmentation, all pixel values were normalized using the standard mean and standard deviation values from the ImageNet dataset. This normalization process helped to stabilize training and ensured compatibility with pre-trained model weights. The preprocessing pipeline was implemented using PyTorch's torchvision.transforms module and was applied dynamically during training and validation using data loaders with GPU-accelerated batch handling.

## 2.2 Model Architectures

This study evaluated four convolutional neural network architectures: a custom-designed CNN, ResNet-50, EfficientNet-B0, and Inception-V3. These models were selected to represent a range of design principles, from lightweight and interpretable to deep and high-performing.

### 2.2.1 Custom CNN

The custom CNN was designed from scratch to serve as a lightweight and interpretable baseline model. It consists of three convolutional blocks, each comprising a two-dimensional convolutional layer followed by a ReLU activation and a max-pooling layer. The resulting feature maps are flattened and passed through two fully connected layers. A dropout layer with a dropout rate of 0.5 is inserted between the dense layers to reduce overfitting. The output layer contains 29 neurons with a softmax activation to match the number of ASL gesture classes.

### 2.2.2 ResNet-50

The ResNet-50 architecture, introduced by He et al. [7], is a deep residual network that utilizes identity shortcut connections to allow gradients to flow more effectively through the network. This lessens the vanishing gradient issue that very deep networks frequently face. The backbone of this investigation was the ImageNet-pretrained ResNet-50 model, whose last fully connected classification layer was swapped out for a new linear layer with 29 output units. At first, the backbone was left frozen and just the recently inserted classification head was trained. To enhance task-specific feature learning, the whole model was refined on the ASL dataset after many epochs.

### 2.2.3 EfficientNet-B0

The basic form of the EfficientNet family, EfficientNet-B0, was put out by Tan and Le [8]. It uses compound scaling to grow network depth, breadth, and resolution all at once. This approach results in a highly efficient model that maintains high accuracy while using fewer computational resources. In this study, the ImageNet-pretrained EfficientNet-B0 was adopted, and its classifier head was replaced with a new fully connected layer suitable for the 29-class ASL recognition task. During training, an initial transfer learning phase was conducted, followed by full fine-tuning of the model parameters.

### 2.2.4 Inception-V3

Inception-V3, developed by Szegedy et al. [9], is a deep convolutional architecture that utilizes Inception modules to capture multi-scale spatial features. Each Inception module applies multiple convolutional operations of different kernel sizes in parallel, and then concatenates their outputs. This enables the model to concurrently learn local and global properties. In this work, a dense layer with 29 output units was used in place of the final classifier layer after a pre-trained Inception-V3 model was loaded. The auxiliary classifier and factorization layers were disabled to simplify the training process. As with other models, a two-phase training strategy was used, starting with the frozen backbone and progressing to full fine-tuning.

All three pre-trained models (ResNet-50, EfficientNet-B0, and Inception-V3) were initialized with ImageNet weights. Their original classification heads were removed and replaced with new layers tailored to the ASL task. This fine-tuning process allowed the models to retain their general visual feature extraction capabilities while adapting to the specific requirements of ASL gesture classification.

## 2.3 Experimental Setup and Evaluation

### 2.3.1 Training configuration

All experiments were conducted using the PyTorch 2.0 deep learning framework. Training was performed on a local workstation equipped with an NVIDIA RTX 3090 GPU with 24 GB of memory, an Intel Core i9 processor, and 64 GB of RAM. This configuration ensured fast training and consistent evaluation across all models.

The Adam optimizer was used to train each model, and 0.001 was chosen as the initial learning rate [10]. The learning rate was lowered by a factor of 0.1 every seven epochs through the use of a step-based learning rate scheduler. The cross-entropy loss function was used as the optimization objective, as it is standard for multi-class classification tasks.

The batch size was fixed at 32 for all models. A maximum of 50 training epochs was allowed, with early stopping enabled to prevent overfitting. Training was terminated if the validation accuracy did not improve for 10 consecutive epochs, and the best-performing model weights were saved for evaluation. Random seeds were fixed through-out the experiment to ensure reproducibility.

**2.3.2 Evaluation metrics**

Both deployment efficiency and classification quality were used to assess the model's performance. The categorization metrics were F1-score, recall, weighted precision, and overall accuracy. For each model, confusion matrices were also produced in order to offer comprehensive information on prediction performance by class.

To assess real-world usability, system performance metrics were also measured. These included inference latency, defined as the average time required to classify a single image (in milliseconds), and throughput, measured as the number of images processed per second. Inference tests were conducted with gradient computation disabled to simulate deployment conditions.

## 3. Results and Discussion

### 3.1 The Performance of Models

The ASL fingerspelling dataset used in this study consists of approximately 87,000 RGB images categorized into 29 classes, including 26 alphabet letters plus "space," "delete," and "nothing." The models were evaluated on a stratified train-validation split to ensure balanced class representation. Table 1 and Fig. 1 summarize the overall classification metrics: EfficientNet-B0 outperformed ResNet-50 (99.6%) and Custom CNN (99.2%), achieving the maximum accuracy at 99.8% with a weighted F1-score of 99.8%. Inception-V3 lagged behind with a substantially lower accuracy of 84.3% and a weighted F1-score of 83.3%. Confusion matrices shown in Fig. 2, Fig. 3, Fig. 4 and Fig. 5. further illustrate these results. EfficientNet-B0's confusion matrix was nearly perfectly diagonal, indicating extremely high class-specific accuracy with minimal misclassifications. ResNet-50 also produced a highly diagonal confusion matrix with very few off-diagonal errors, typically limited to classes with visually similar handshapes. The Custom CNN performed strongly overall but showed slightly more confusion between such challenging pairs. In contrast, Inception-V3 exhibited a noisier confusion matrix with many off-diagonal errors, confirming its reduced ability to distinguish certain hand gestures. These results demonstrate the effectiveness of transfer learning and modern architecture in improving classification performance for ASL fingerspelling recognition. The compound scaling approach used by EfficientNet-B0, which balances network depth, breadth, and resolution, is responsible for its better performance. The analysis also underscores the importance of considering both accuracy and computational efficiency when designing real-time sign language recognition systems.

**Table 1. Overall classification performance of all models on the ASL test set**

| Model | Accuracy | Precision (Weighted) | Recall (Weighted) | F1-Score (Weighted) |
|---|---|---|---|---|
| Custom CNN | 99.2% | 99.2% | 99.2% | 99.2% |
| ResNet50 | 99.6% | 99.6% | 99.6% | 99.6% |
| EfficientNetB0 | 99.8% | 99.8% | 99.8% | 99.8% |
| InceptionV3 | 84.3% | 90.0% | 84.3% | 83.3% |

ISSN 2959-6157

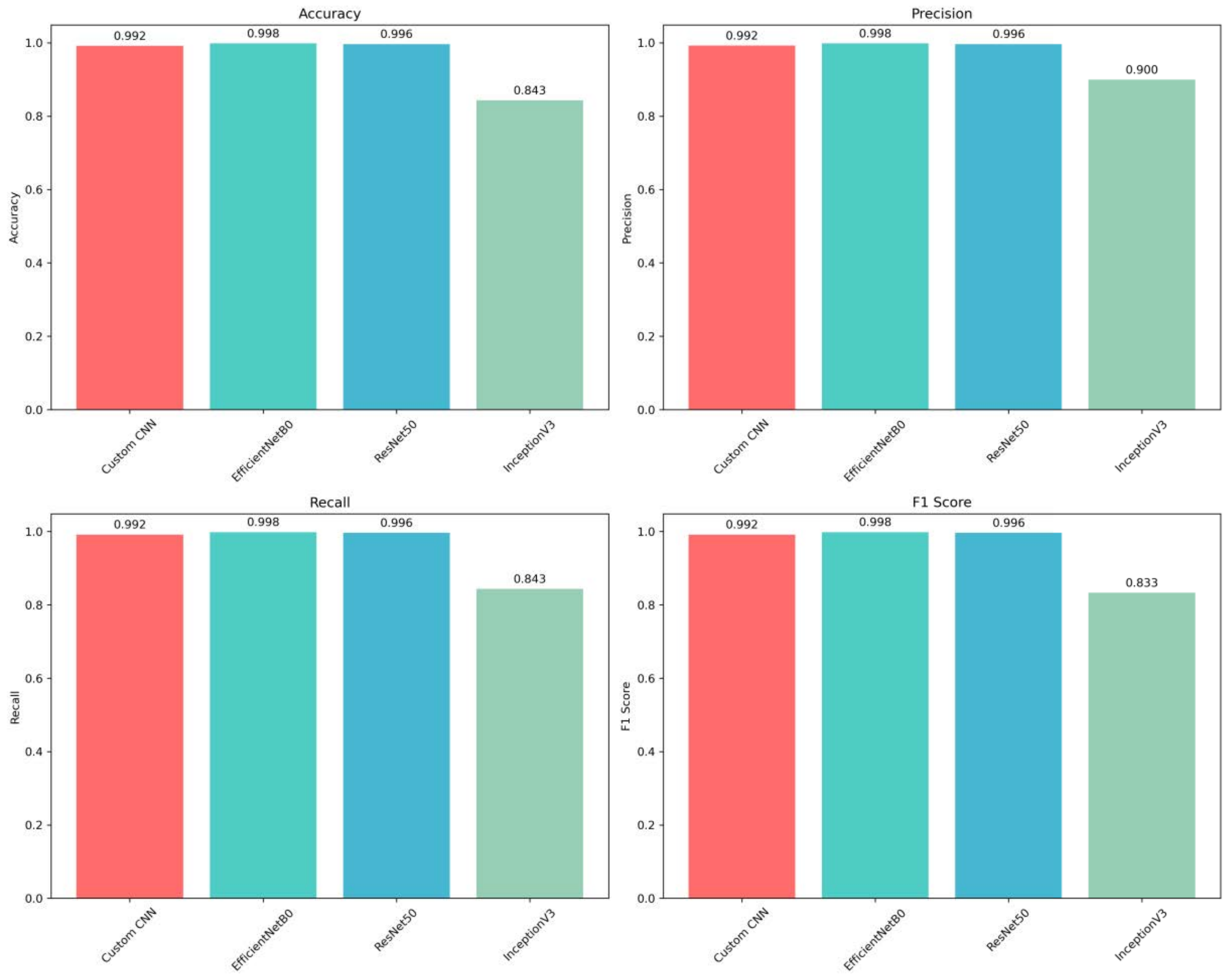

**Fig. 1 Comparison of each model's overall performance in terms of F1-score, recall, accuracy, and precision (Picture credit : Original)**

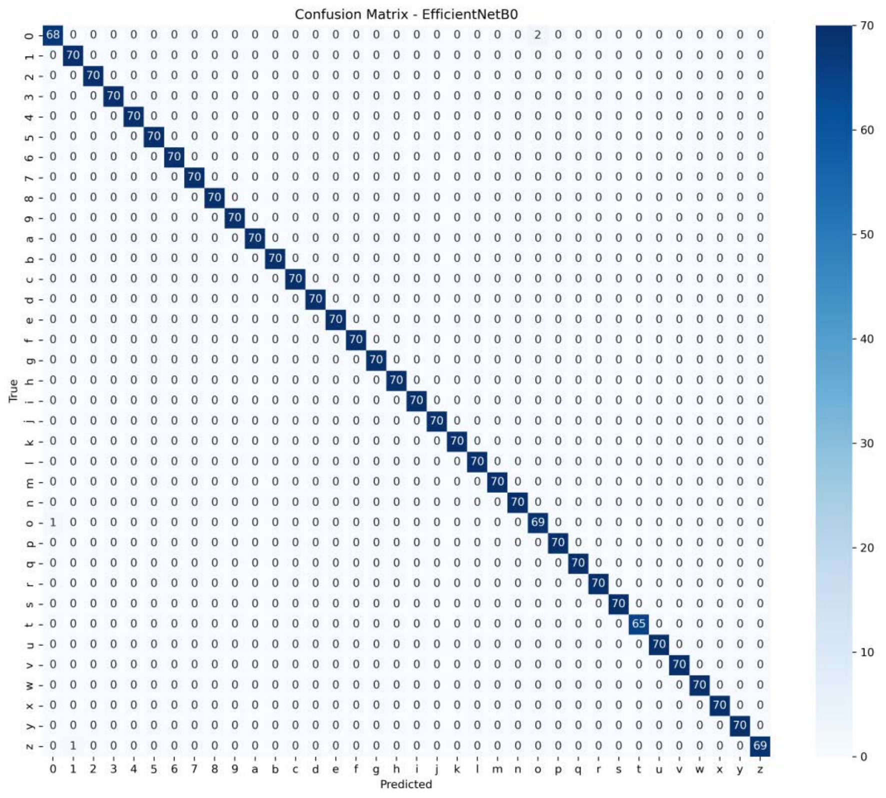

Fig. 2 Confusion matrix for EfficientNetB0 (Picture credit : Original)

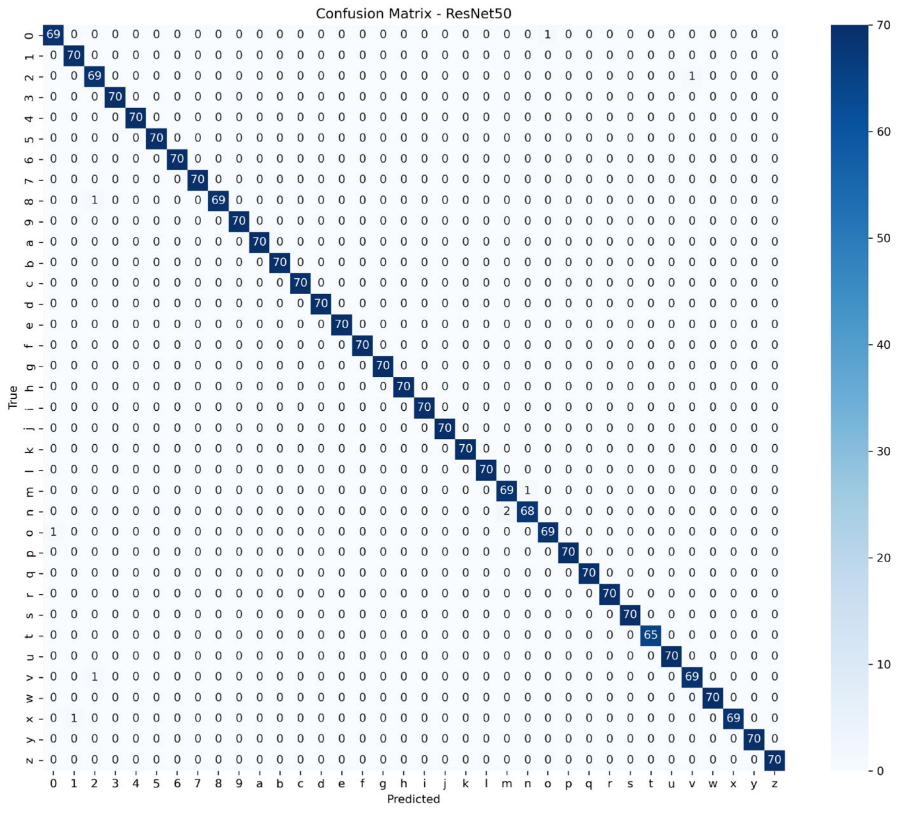

Fig. 3 Confusion matrix for ResNet50 (Picture credit : Original)

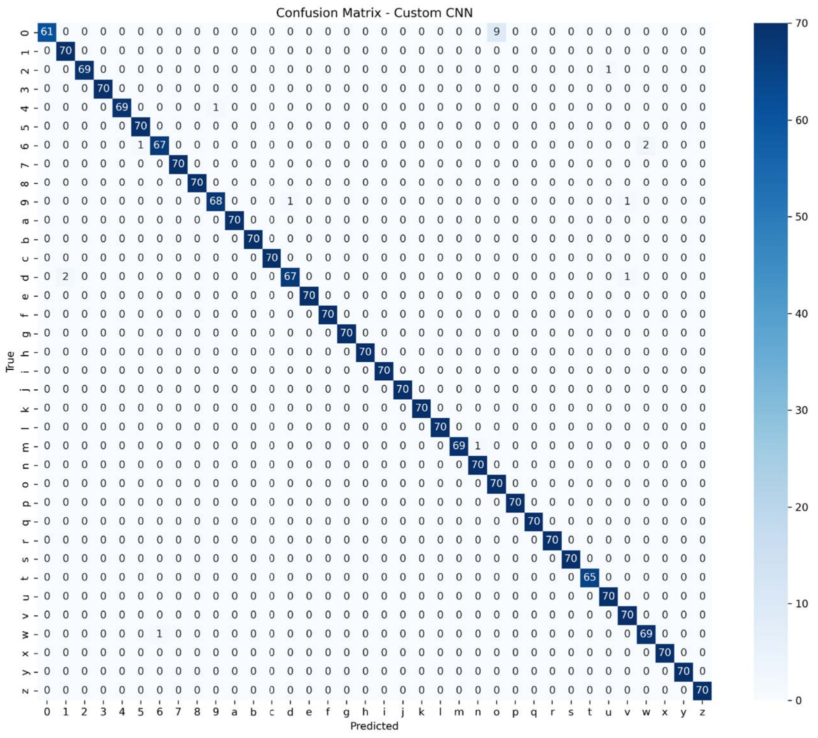

**Fig. 4 Confusion matrix for custom CNN (Picture credit : Original)**

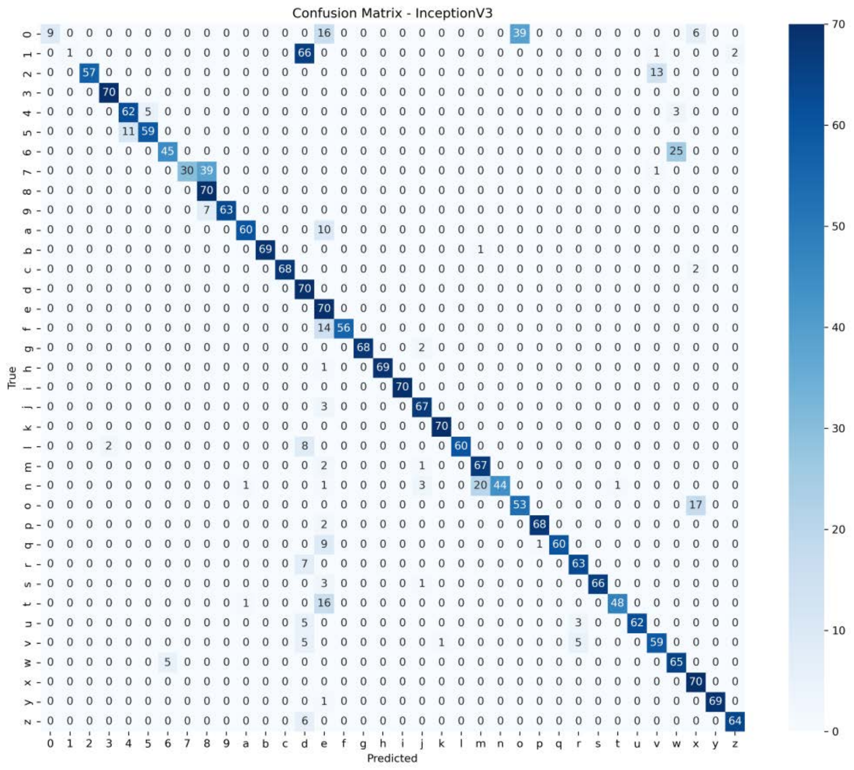

**Fig. 5 Confusion matrix for InceptionV3 (Picture credit : Original)**

## 3.2 Discussion

These results clearly highlight the advantages of transfer learning for ASL fingerspelling recognition. In terms of accuracy and F1-score, EfficientNetB0 and ResNet50, which use pre-trained weights from extensive datasets, both performed noticeably better than Custom CNN. The compound scaling approach of EfficientNetB0, which strikes a balance between network depth, width, and resolution for maximum accuracy and efficiency, is responsible for its exceptional performance.

The confusion matrices offer further insight into each model's strengths and weaknesses. EfficientNetB0 and ResNet50 made very few errors, typically between highly similar gestures. Custom CNN, although highly accurate, showed slightly more confusion between such challenging pairs.

InceptionV3's markedly higher error rate suggests that, for this dataset and training setup, it was less effective at learning the discriminative features required to separate similar classes. Possible contributing factors include differences in input resolution requirements, sensitivity to overfitting or underfitting, and the specific hyperparameter tuning applied during transfer learning.

Another important consideration is the practical trade-off between accuracy and deployment. While Custom CNN offers a simpler architecture with potentially faster inference on low-power devices, its slightly lower accuracy may limit its suitability for highly reliable applications. Conversely, EfficientNetB0 and ResNet50 deliver both exceptional accuracy and robust performance, making them strong candidates for real-world ASL recognition systems.

## 4. Conclusion

In this study, four deep learning models were evaluated for American Sign Language fingerspelling recognition using a balanced dataset of 29 classes. The results show that transfer learning models, particularly EfficientNetB0, achieve superior accuracy and robustness compared to a custom-designed CNN. EfficientNetB0 reached a test accuracy of 99.8% with minimal misclassifications. These findings highlight the effectiveness of modern pre-trained architecture for ASL recognition tasks. Future work may focus on real-time deployment, expansion to dynamic gestures, or further optimization of inference speed for deployment on edge devices.

## References

[1] Pigou L, Dieleman S, Kindermans P J, Schrauwen B. Sign Language Recognition Using Convolutional Neural Networks. European Conference on Computer Vision (ECCV) Workshops, 2015: 572–578.

[2] Molchanov P, Gupta S, Kim K, Kautz J. Hand Gesture Recognition With 3D Convolutional Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2015: 1–7.

[3] Kopuklu O, Gunduz A, Kose N, Rigoll G. Real-time Hand Gesture Detection and Classification Using Convolutional Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019.

[4] Koller O. Deep Sign Language Recognition: Modeling Sequential Dependencies for End-to-End Continuous Sign Language Recognition. International Journal of Computer Vision, 2020, 128(5): 1923–1940.

[5] Zhao J, Wang J, Cheng K, Jia K. SL-ResNet: Sign Language Recognition with Residual Neural Networks. IEEE Access, 2019, 7: 110514–110523.

[6] Ayuraj. ASL Dataset. Kaggle, 2020. https://www.kaggle.com/datasets/ayuraj/asl-dataset

[7] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016: 770–778.

[8] Tan M, Le Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. International Conference on Machine Learning (ICML), 2019: 6105–6114.

[9] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016: 2818–2826.

[10] Zhang Z. Improved adam optimizer for deep neural networks. In2018 IEEE/ACM 26th international symposium on quality of service (IWQoS) 2018 Jun 4 (pp. 1-2). IEEE.