# CS273P Machine Learning Final Project

Chuqi Wang, Zhihang Feng, Shengtong Sun

March 23, 2024

**Abstract**

Automatic recognition of sign language poses significant challenges and opportunities for enhancing communication accessibility for the deaf and hard-of-hearing community. This project aims to develop a deep learning model capable of accurately classifying sign language gestures from images. We primarily utilized a convolutional neural network (CNN) architecture, trained on a dataset of 27455 labeled images covering 24 sign language gestures. We trained LeNet, customized CNNs, ResNet and ran different experiments to explore the model performance. Our best model achieved a 99% accuracy on a separate test set after using data augmentation, outperforming previous benchmarks. These results demonstrate the potential of deep learning in bridging communication gaps and suggest avenues for future work in real-time sign language recognition systems.

## 1 Introduction

The automatic recognition of sign language presents a unique set of challenges and opportunities in the quest to improve accessibility for the deaf and hard-of-hearing community. This project endeavors to bridge these communication gaps through the development of a sophisticated deep learning model designed for the accurate classification of sign language gestures from images. By leveraging advanced convolutional neural network (CNN) architectures, including LeNet, custom models, and ResNet, and training on an extensive dataset of 27,455 labeled images representing 24 distinct gestures, we compared the test performance of each model and enhanced test accuracy by using data augmentation, culminating in an achievement of 99% accuracy using 2 layers CNN on a separate test set. The promising results of this project lay the groundwork for future research and development in real-time sign language recognition systems, promising a more connected and accessible world for individuals relying on sign language for communication.

## 2 Datasets

The data is collections of images in pixels of sign language [1], and they are divided into training and test datasets. Specifically, there are 784 pixels for one image, 27455 images for the training dataset, and 7172 images for the testing dataset. There are twenty-four letters that can be represented by hand sign, which letter 'j' and letter 'z' are excluded. In the training process, we distributed around 1000 images to each letter, which is shown in Figure 1. In the testing process, averaging 300 images are distributed to each letter, which is shown in Figure 2. In Figure 3, there are showcases for sign language in representing different letters.

## 3 Method

The first methodology used was sequential CNN, which is commonly used for image classification tasks. This particular model is structured to recognize sign language gestures, as implied by the final dense layer with 24 units, corresponding to the 24 classes of hand gestures (excluding gestures for letters that can be confused with motion, like "J" and "Z"). In CNN, LeNet is known for alternating between convolutional layers and subsampling (pooling) layers, and we will it in our model. The activation function is choosen to be ReLU for all models.

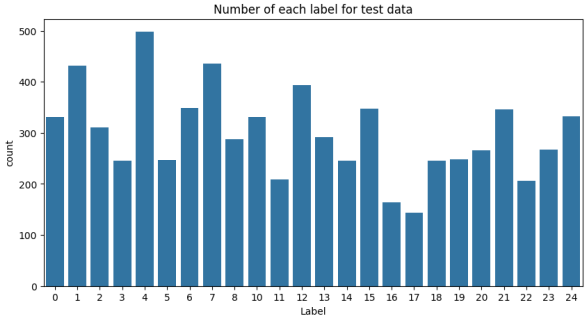


Figure 1: Testing Label Distribution

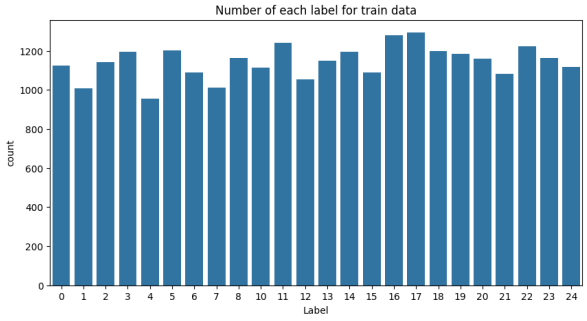


Figure 2: Training Label Distribution

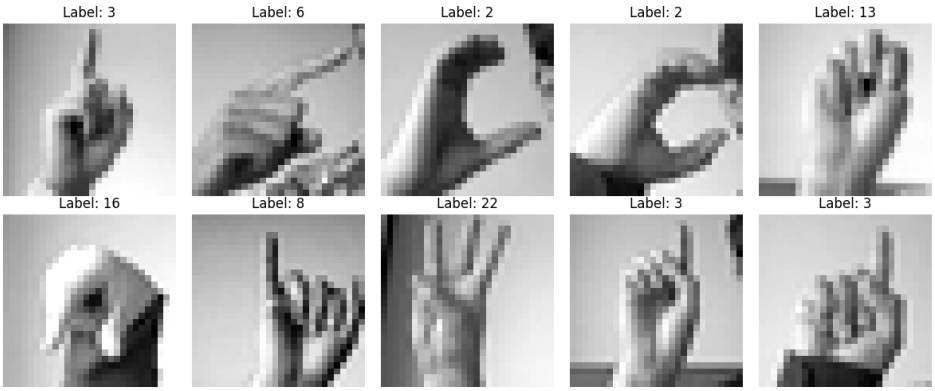


Figure 3: Training Data visualization

### 3.1 Customized 2-Layer CNN

This model is a customized two-layer CNN designed for image classification with a 28x28 grayscale input. It features two convolutional layers followed by max-pooling layers, aimed at extracting features at two levels of abstraction. The network flattens the output of the convolutions and follows it with a dense layer of 128 neurons with ReLU activation. A dropout layer is included before the final classification layer to mitigate overfitting by randomly setting 50% of the input units to 0 at each update during training time. The output layer uses a softmax activation function to classify the images into 24 classes.

### 3.2 LeNet- Inspried Model

The LeNet-inspired model [3] uses two convolutional layers with 6 and 16 filters, respectively, both followed by max-pooling layers. After flattening the pooled feature maps, the network employs two dense layers with 120 and 84 neurons each. The output layer is adapted to the specific number of classes (num_classes) with a softmax activation function. The use of padding='same' in the first layer ensures that the output feature map has the same dimensions as the input.

### 3.3 Customized 3-Layer CNN

This customized three-layer CNN extends the complexity of the two-layer CNN by incorporating an additional convolutional layer. Each convolutional layer is followed by a max-pooling layer to reduce dimensionality, and dropout layers are strategically placed after each pooling step and the last convolutional layer to prevent overfitting, with the dropout rate increasing to 50% before the final dense layer. The network concludes with a dense layer of 64 neurons, followed by the output layer with a softmax activation function, categorizing the input into 24 classes.

All models follow the general structure of a CNN, with convolutional layers followed by pooling layers,

and dense layers culminating in a softmax output for multi-class classification. Dropout layers are a common regularization technique used in all models to reduce overfitting, with varying rates of dropout used at different points in the network. The models differ in their depth and complexity, with the first being the simplest and the third the most complex. The choice between them should be informed by the specific requirements of the project, the complexity of the dataset, and the computational resources available.

### 3.4 ResNet Model

The ResNet model [2] created here is a customized, scaled-down version suitable for processing 28x28 grayscale images and distinguishing among 24 different classes. It incorporates the essence of the original ResNet architecture, which is characterized by its residual blocks. The network begins with an initial convolutional layer using 64 filters of size 7x7 and a stride of 2, preserving the spatial dimensions using 'same' padding. This layer is followed by batch normalization and a ReLU activation function. Then, a max-pooling layer with a 3x3 window and a stride of 2 further reduces the spatial dimensions.

Two residual blocks with 64 filters are applied, which maintain the dimensions of the feature maps. Subsequently, a residual block with 128 filters is used, where dimensionality reduction is applied via striding. This is followed by another 128-filter residual block without dimension reduction. In each residual block, the flow consists of two convolutional layers with batch normalization and ReLU activations. When reducing dimensions, a parallel convolution with a 1x1 filter and a stride of 2 is applied to the shortcut connection to match the dimensions before adding it to the main path. During training, the model is expected to learn to identify and classify 24 different sign language gestures. The effectiveness of this ResNet-inspired model, especially given the lower complexity suitable for the input size and class count, would be measured by its classification accuracy on validation and test data.

## 4 Results

In table 1, it is shown the results comparison with or without data augmentation. It showcases the impact of data augmentation on the accuracy of four CNN models: a 2-layer conv2d CNN, LeNet, a 3-layer conv2d CNN, and ResNet. Before data augmentation, the 3-layer conv2d CNN led with an accuracy of 96.42%, while ResNet trailed significantly at 4.85%. After implementing data augmentation, all models saw improved accuracy, with the 2-layer conv2d CNN reaching the highest accuracy of 99.29%. The most dramatic improvement was observed in ResNet, which saw accuracy rise to 87.05%, suggesting a substantial benefit from the augmentation techniques used. This demonstrates the effectiveness of data augmentation in enhancing the generalization ability of CNN models.

| | **2 conv2d CNN** | **LeNet** | **3 conv2d CNN** | **ResNet** |
|---|---|---|---|---|
| Before Data augmentation | 91.17% | 86.50% | 96.42% | 4.85% |
| After Data augmentation | **99.29%** | **98.83%** | **97.48%** | **87.05%** |

Table 1: Model performance before and after data augmentation

The 2-layer CNN exhibited (Figure 4, 5) a notable improvement in test accuracy after data augmentation, achieving close to peak performance. The loss for both train and test datasets showed a consistent decrease, indicating stable learning. The LeNet model (Figure 6, 7) showed significant fluctuations in test accuracy and loss, but with data augmentation, these variations reduced and both accuracy and loss improved markedly, showcasing the benefits of augmentation in stabilizing and enhancing model performance. The 3-Layer CNN model (Figure 8, 9) also benefited from data augmentation, with both accuracy and loss graphs indicating more stable and improved performance post-augmentation. The ResNet model (Figure 10, 11) initially had a very high train accuracy that was consistent even after data augmentation. However, its test accuracy and loss before augmentation were quite erratic, displaying high variance. Data augmentation helped in smoothing out these fluctuations considerably, indicating improved model robustness.

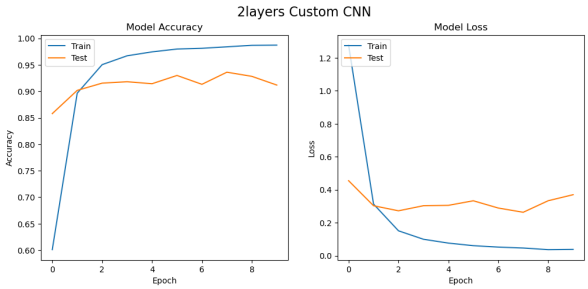


Figure 4: 2 Layer CNN Model Accuracy and Loss

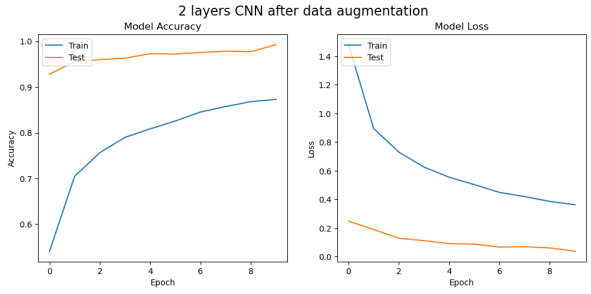


Figure 5: 2 Layer CNN Model Accuracy and Loss with Data Augmentation

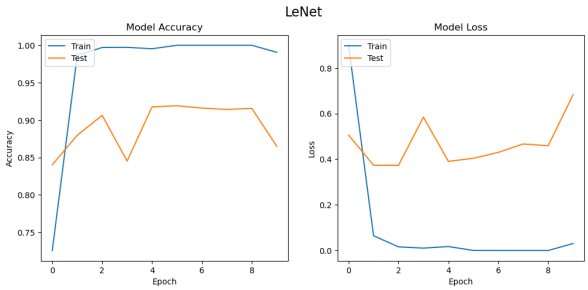


Figure 6: LeNet Model Accuracy and Loss

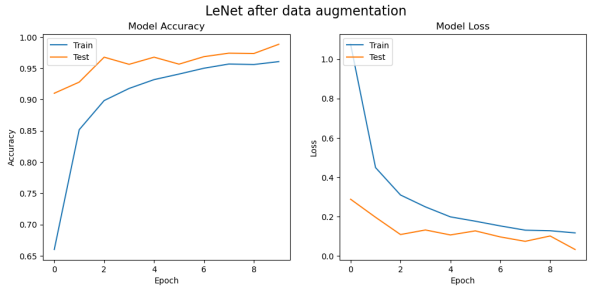


Figure 7: LeNet Model Accuracy and Loss with Data Augmentation

# 5 Analysis

## 5.1 Impact of Data Augmentation[4]

The Sign Language MNIST dataset encompasses a variety of gestures, each representing different letters. The application of rotation, translation, shearing, and scaling artificially increases the variation within the dataset's gesture images, which may include changes in angles, positions, sizes, and shape distortions. This aids the model in learning how to recognize the same gesture from various angles and sizes, thereby enhancing the model's accuracy in dealing with unseen gesture images. Given the limited size of the Sign Language MNIST dataset and the typically large number of parameters in deep learning models, there's a propensity for overfitting on small datasets. Data augmentation expands the dataset, making it less likely for the model to learn noise and irrelevant features within the data, and instead, to learn more generalized and truly useful features for gesture recognition. By increasing the diversity of training data, data augmentation helps improve the model's performance on unseen data.

## 5.2 Evaluation of ResNet Model

In the realm of complex image recognition challenges, such as those posed by the ImageNet dataset, architectures like ResNet are designed to excel by capturing intricate patterns through their deep and complex structures. However, this inherent complexity may not be essential, and can even be counterproductive, for simpler datasets like the Sign Language MNIST. The detailed and sophisticated layers of ResNet can lead to overfitting, a scenario where the model, instead of learning generalizable features, tends to memorize the noise inherent in the training data. Consequently, this diminishes the model's ability to generalize to unseen data, reflected in subpar performance on validation or test sets.

Simplified models, characterized by fewer parameters and a streamlined architecture, exhibit a reduced propensity for overfitting. These models, by virtue of their simplicity, are better suited for tasks with limited training data, often outperforming their more complex counterparts in such scenarios. The adaptation of ResNet or similar complex models to smaller or simpler datasets necessitates care-

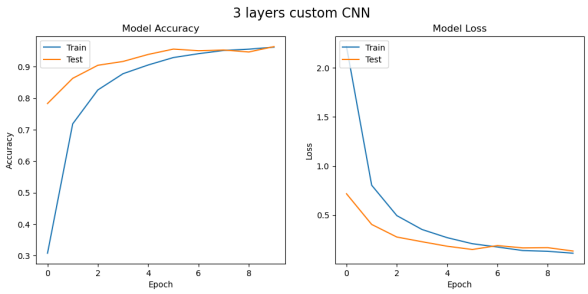


Figure 8: 3 Layer CNN Model Accuracy and Loss

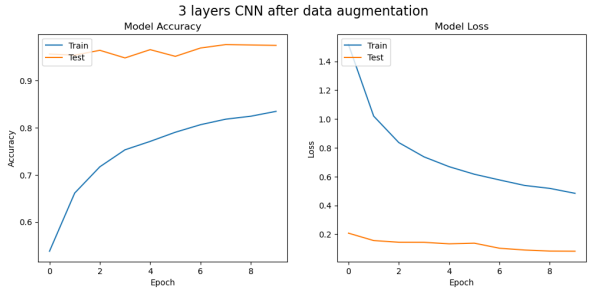


Figure 9: 3 Layer CNN Model Accuracy and Loss with Data Augmentation

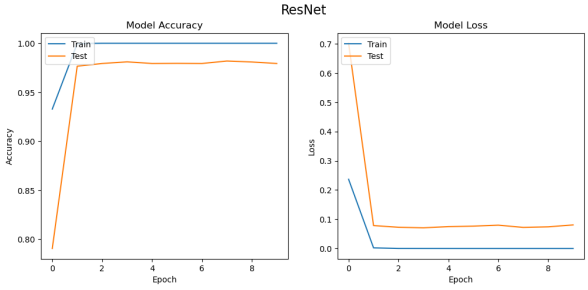


Figure 10: ResNet Model Accuracy and Loss

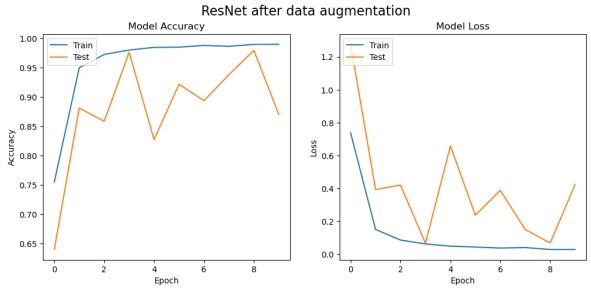


Figure 11: ResNet Model Accuracy and Loss with Data Augmentation

ful modifications to the network's structure. This might involve reducing the number of convolutional layers, adjusting the quantity of filters, or altering the connection patterns within the network to align more closely with the dataset's unique features and challenges. Neglecting to appropriately customize these models for the task at hand can lead to suboptimal outcomes, underscoring the importance of a tailored approach to model selection and design in achieving superior performance in image recognition tasks..

# 6 Discussion and Conclusion

Our project on automatic sign language recognition using convolutional neural networks (CNNs) demonstrates significant advancements in communication accessibility for the deaf and hard-of-hearing. By employing architectures like LeNet, customized CNNs, and ResNet on a dataset of 27,455 images, we achieved a 99% test accuracy with a 2-layer CNN model after implementing data augmentation. This breakthrough underscores the efficacy of deep learning in sign language recognition and highlights the critical role of data augmentation in enhancing model generalizability.

In summary, convolutional neural network (CNN) architectures model works very well in image classification problems, CNN can extract the most important features from images and it use less parameters than MLP.

# 7 State of Contributions

**Chuqi Wang:** Code implementation, contribution on writing report.
**Zhihang Feng:** Code implementation, contribution on writing report.
**Shengtong Sun:** Code implementation, contribution on writing report.

# References


[1] Diponkor Bala, Bappa Sarkar, Md Ibrahim Abdullah, and Mohammad Alamgir Hossain. American sign language alphabets recognition using convolutional neural network. *International Journal of Knowledge Based Computer Systems*, 9(1), 2021.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[4] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.