EE 541 – Computational Introduction to Deep Learning

American Sign Language Gesture Recognition

Haoyu Xie, Anusha Poornesh

May 8, 2023

Table of Contents

[1 Abstract 1](#_Toc134422486)

[2 Introduction 2](#_Toc134422487)

[2.1 Literature Review 3](#_Toc134422488)

[3 Data 3](#_Toc134422489)

[3.1 Dataset Description 3](#_Toc134422490)

[3.2 Data Pre-Processing 4](#_Toc134422491)

[4 Models Description 4](#_Toc134422492)

[4.1 ResNet-18 4](#_Toc134422493)

[4.1.1 Results 5](#_Toc134422494)

[4.2 SqueezeNet 6](#_Toc134422495)

[4.2.1 Results 7](#_Toc134422496)

[4.3 Our Model 8](#_Toc134422497)

[4.3.1 Results 9](#_Toc134422498)

[5 Extension 10](#_Toc134422499)

[6 Discussion 10](#_Toc134422500)

[7 Future Expansion 11](#_Toc134422501)

[8 Conclusion 11](#_Toc134422502)

[References 11](#_Toc134422503)

# 1 Abstract

In this project, our primary objective is to leverage deep learning techniques for the recognition of American Sign Language (ASL) to enhance communication opportunities for individuals with disabilities. We trained an ASL dataset through the PyTorch framework for three distinct models, ResNet, SqueezeNet, and a model built by us. By comparing the models and fine-tuning their parameters, we identified the optimal architecture for ASL recognition. We also developed a website for American Sign Language Detection, which captures users' signs through camera input and utilizes the Text-to-Speech API to convert the recognized ASL gestures into audible speech.

The most effective model demonstrated robust ASL recognition capabilities, enabling seamless communication for users with disabilities through the website. The Text-to-Speech API ensured accurate conversion of recognized signs into audible speech, significantly enhancing accessibility and inclusivity.

# 2 Introduction

Sign languages are an extremely important communication tool for many deaf and hard-of-hearing people. The significance of American Sign Language recognition technology lies in its potential to improve communication and accessibility for the deaf community. The deaf community face challenges in communicating across various sectors where ASL is not predominant. A reliable and accurate ASL recognition system can help build a more inclusive environment between deaf and hard-of-hearing individuals and those who do not understand sign language.



*Figure 1: American Sign Language Chart*

A solution to this problem has benefits across various industries. Firstly, it can empower the deaf community by enabling more effective communication with others and improving their access to essential services and opportunities. Healthcare facilities can benefit from ASL recognition technology to provide clear and accurate communication between the deaf community and healthcare professionals, while schools and universities can utilize the technology to provide better educational support. Businesses can leverage ASL recognition technology to provide accessible customer support and reach a broader audience. Academic research in the fields of computer vision and natural language processing can apply ASL recognition advancements to develop for sign language understanding and translation.

We aim to implement a robust and adaptable model that can accurately recognize ASL gestures in real-time. In this project, we will explore various CNN architectures, such as ResNet, SqueezeNet, and a custom model inspired by SqueezeNet, to determine the optimal model for ASL recognition. We will implement these models using the PyTorch framework and train them on a comprehensive ASL dataset.

Our goal is to develop a system that effectively bridges the communication gap between the deaf and hard-of-hearing community and the broader society. The successful implementation of this project has the potential to significantly improve accessibility and inclusivity in various sectors, empowering the deaf community by facilitating more effective communication and opening new opportunities for them.

## 2.1 Literature Review

He et al. introduced the concept of deep residual learning in their paper "*Deep Residual Learning for Image Recognition*" [1]. They proposed ResNet, a deep learning architecture which allowed the network to learn more efficiently by enabling the direct flow of information from one layer to another. The structure of ResNet can accelerate the training of neural networks very quickly, and the accuracy of the model has also been greatly improved. It is widely used in computer vision-related fields.

SqueezeNet, a compact CNN architecture that achieved AlexNet-level accuracy with significantly fewer parameters and smaller model size was introduced by Iandola et al in their paper ‘*SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <50MB Model Size*’ [2]. It is a miniaturized network model structure, which uses 50 times fewer parameters than AlexNet while ensuring that the detection accuracy is not reduced. It has been applied in various computer vision tasks, including ASL recognition, due to its computational efficiency and effectiveness in processing image data.

In the study, ‘DeepASLR: A CNN based human computer interface for American Sign Language recognition for hearing-impaired individuals’ [3], KASAPBASI et al. proposed DeepASLR. The study showed that the DeepASLR model can provide accurate and efficient ASL recognition, for easy communication for hearing-impaired individuals.

In "*A Real-time American Sign Language Recognition System Using Convolutional Neural Network for Real Datasets*," Kadhim and Khamees demonstrated the feasibility of using CNNs to recognize ASL gestures in real-time [4]. Their study presented a real-time ASL recognition system that achieved promising results using CNNs, highlighting the effectiveness of such architectures for ASL recognition tasks.

# 3 Data

## 3.1 Dataset Description

The dataset is a collection of images of alphabets from the American Sign Language, separated in 29 folders that represent the various classes. The training dataset contains 87,000 images which are 200 x 200 RGB pixels. There are 29 classes, of which 26 are for the letters A-Z and 3 classes for space, delete and nothing. The images in each category cover a range of different scenarios such as gestures in dimly lit environment, gestures in bright light and gestures with shadow. The test dataset contains a mere 29 images to encourage the use of real-world test images. As the test set is very small, results from our own testing are also included.

## 3.2 Data Pre-Processing

In the data processing stage, we first apply grayscale processing to the dataset. This involves converting the original colour images into grayscale images, thereby simplifying the data and reducing computational complexity. Next, we normalize the pixel values of the grayscale images to a range of 0 to 255. These preprocessing steps ensure that the hand gestures in the images exhibits a solid contrast against the background, making it easier for the model to recognize ASL signs.

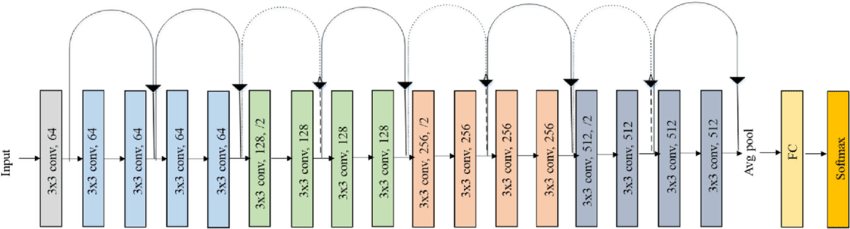
Following the preprocessing, we partition the dataset into three distinct subsets. These subsets comprise a training set (80% of the data), a validation set (10% of the data), and a test set (10% of the data). This division enables us to perform cross-validation during the model training process, which helps to minimize the risk of overfitting. By regularly evaluating the model's performance on the validation set, we can fine-tune its parameters and improve its ability to generalize to new, unseen data. This rigorous validation process contributes to the development of a more robust and accurate ASL recognition model.

## 4 Models Description

In this project, we have implemented and tested ResNet-18, SqueezeNet and a custom model based on SqueezeNet. The following subsections provide implementation details and results for the models mentioned.

## 4.1 ResNet-18

ResNet-18 is a convolutional neural network (CNN) architecture specifically designed for image classification tasks. It has been trained on an extensive dataset consisting of over a million images from the ImageNet database. The architecture comprises 18 layers, providing both computational complexity and good classification performance.



*Figure 2: Original ResNet-18 Architecture* [5]

ResNet-18 introduced the concept of skip connections or residual connections which allow for the direct flow of information from one layer to another, bypassing certain layers in the network. The architecture of ResNet can be observed in Figure 2. This mechanism facilitates the training of deeper networks without encountering the vanishing gradient problem, which is commonly associated with deep architectures. As a result, ResNets can effectively learn complex patterns and representations without increasing the training error percentage.

ResNet-18 can be efficient in ASL recognition for several reasons. Firstly, its ability to handle large numbers of layers ensures that the network can be trained efficiently without compromising accuracy. Secondly, the architecture addresses the vanishing gradient problem through identity mapping, which further enhances the stability and effectiveness of the model during training. ResNet-18 has demonstrated higher performance in various image classification tasks, making it appropriate for ASL recognition applications.

# 4.1.1 Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Learning Rate** | **Train Data Accuracy** | **Validation Data Accuracy** | **Test Data Accuracy** |
| 0.02 | 1.00 | 1.00 | 0.8571 |
| 0.002 | 0.9946 | 0.9949 | 0.4286 |
| 0.0002 | 0.4426 | 0.4445 | 0. 1786 |

*Table 1: Comparison of accuracy for different values of learning rate*

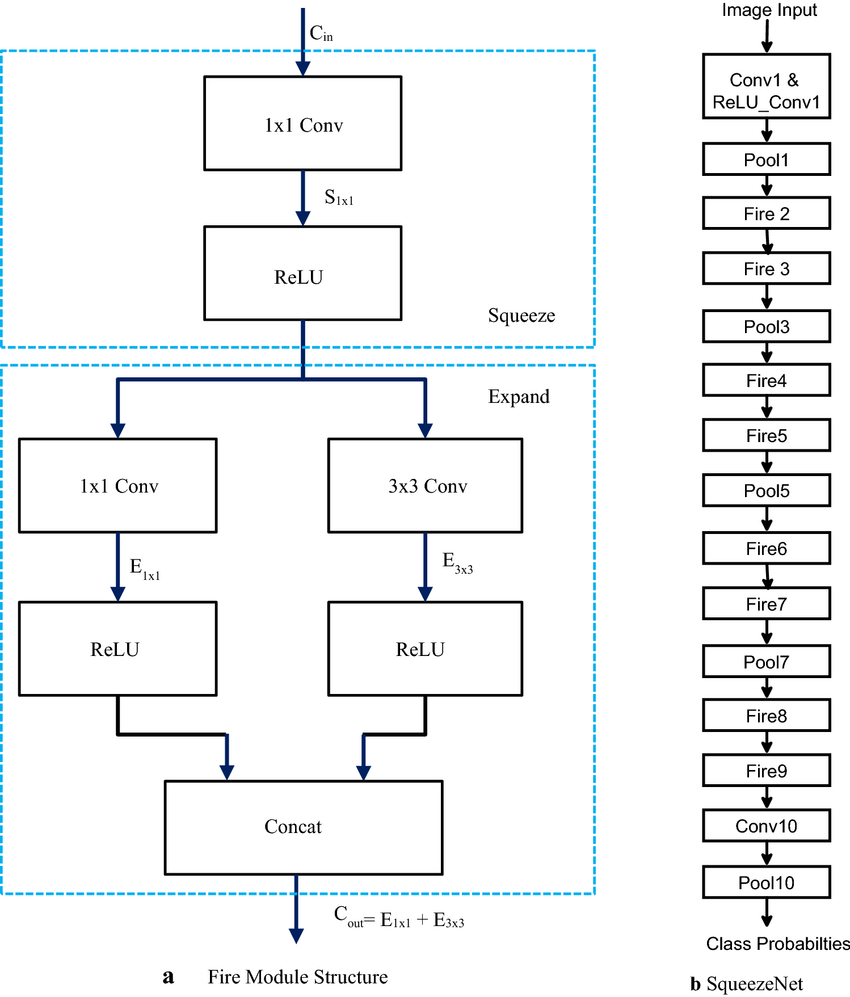
|  |  |  |
| --- | --- | --- |
| **Learning Rate** | **Loss Curve** | **Accuracy Curve** |
| 0.02 |  |  |
| 0.002 |  |  |
| 0.0002 |  |  |

*Table 2: Plotting of loss and accuracy curves for different values of learning rate*

## 4.2 SqueezeNet

SqueezeNet is a convolutional neural network (CNN) architecture that aims to achieve high computational performance by reducing the number of learnable parameters and the computational load of the entire network. By optimizing these aspects, the model's training and testing speeds can be significantly improved. SqueezeNet achieves comparable results to AlexNet on the ImageNet dataset and has 50 times fewer parameters.

The architecture of SqueezeNet comprises a single convolutional layer (conv1) at the beginning, followed by eight Fire modules, and concluding with a final convolutional layer (conv10). The number of filters per Fire module gradually increases from the start to the end of the network. Max pooling with a stride of 2 is performed after the conv1, fire4, fire8, and conv10 layers, further reducing the spatial dimensions of the feature maps and decreasing computational requirements. The architecture of SqueezeNet can be observed in Figure 3.



*Figure 3: Fire module and SqueezeNet architecture* [6]

SqueezeNet can be a valuable choice for ASL recognition due to its efficient and compact architecture. With fewer parameters and reduced computational load, the model can be trained and tested more quickly, which is beneficial for real-time ASL recognition tasks. Moreover, the smaller model size makes it suitable for deployment on devices with limited storage and computational capabilities. Although SqueezeNet was originally designed for general image classification tasks, its efficiency makes it a suitable for ASL recognition applications.

# 4.2.1 Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Learning Rate** | **Train Data Accuracy** | **Validation Data Accuracy** | **Test Data Accuracy** |
| 0.02 | 0.9592 | 0.9637 | 0.8929 |
| 0.002 | 0.9697 | 0. 9861 | 0.9286 |
| 0.0002 | 0.0835 | 0.0825 | 0.0714 |

*Table 1: Comparison of accuracy for different values of learning rate*

|  |  |  |
| --- | --- | --- |
| **Learning Rate** | **Loss Curve** | **Accuracy Curve** |
| 0.02 |  |  |
| 0.002 |  |  |
| 0.0002 |  |  |

*Table 2: Plotting of loss and accuracy curves for different values of learning rate*

## 4.3 Our Model

We designed a custom model for ASL recognition, based on the SqueezeNet architecture. We made modifications to the base architecture to try to enhance the model's performance and generalizability. The modified model structure is described below.

The custom model for ASL recognition comprises two main components: feature extraction and classification. The feature extraction stage starts with a convolutional layer that focuses on local features within the input image, followed by a ReLU activation function and a max-pooling layer to reduce spatial dimensions. The classifier component begins with a dropout layer to prevent overfitting, a convolutional layer that acts as a fully connected layer, another ReLU activation function, and finally an adaptive average pooling layer to condense the feature maps into a compact representation.

# 4.3.1 Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Learning Rate** | **Train Data Accuracy** | **Validation Data Accuracy** | **Test Data Accuracy** |
| 0.02 | 0.6473 | 0.7090 | 0.7857 |
| 0.002 | 0.9511 | 0.9778 | 0.8929 |
| 0.0002 | 0.0395 | 0.0401 | 0.0420 |

*Table 1: Comparison of accuracy for different values of learning rate*

|  |  |  |
| --- | --- | --- |
| **Learning Rate** | **Loss Curve** | **Accuracy Curve** |
| 0.02 |  |  |
| 0.002 |  |  |
| 0.0002 |  |  |

*Table 2: Plotting of loss and accuracy curves for different values of learning rate*

# 5 Extension

For this project, we have extended our work by developing a full-stack website that integrates our ASL recognition model with real-time camera input, providing a seamless and user-friendly interface for ASL gesture recognition. We have utilized Flask, a lightweight web framework, to facilitate communication between the frontend and backend components.

Our system is designed to access the local camera and transmit real-time images to the frontend webpage as a data stream. Simultaneously, the backend invokes the trained CNN model to recognize the ASL gestures captured by the camera. The recognition results are then conveyed to the frontend in both visual and auditory formats.

To achieve this, we have incorporated a Text-To-Speech (TTS) API called pyttsx3. This cross-platform TTS API converts the letters corresponding to the recognized gestures into spoken words. However, during the initial implementation, we encountered issues with the stability of the model's recognition, which resulted in the TTS API occasionally pronouncing incorrect results.

To address this issue and enhance the user experience, we refined the code by triggering the TTS API only when the recognized gesture is detected as stable. Furthermore, we introduced a time delay between broadcasts, ensuring that each result can be entirely read aloud before the next recognition takes place. These adjustments aim to provide a more accurate user experience while using our ASL recognition system.

# 6 Discussion

In this project, we explored three different models to train on the ASL dataset with varying results. The first model, ResNet-18, demonstrated high accuracy on the training set and validation. While it performed well overall, we sought to improve training efficiency and experimented with alternative models.

The second model we tested was SqueezeNet, which provided faster training and reduced computational time. However, SqueezeNet showed lesser accuracy when compared to ResNet. Nevertheless, we aimed to develop a custom model to further enhance our ASL recognition system.

Our custom model, based on SqueezeNet architecture, achieved similar training speeds as SqueezeNet while the accuracy remained comparable on both the training and validation sets. This demonstrates the possible effectiveness of our tailored approach to ASL recognition.

We discovered that different hyperparameters, such as learning rate, have a significant impact on model performance. Our experiments revealed that a learning rate of 0.002 was the optimal choice for our custom model. This choice led to higher accuracy and lower loss rates in comparison to other learning rates. The importance of fine-tuning the learning rate stems from the fact that an excessively high learning rate may lead to large fluctuations in model performance or even the risk of gradient explosion, whereas an overly low learning rate may result in slow model learning.

# 7 Future Expansion

Future expansion of this ASL recognition project could involve developing a real-time continuous sign language recognition and translation system for use with Augmented Reality (AR) devices, such as smart glasses. This would enable seamless conversations between ASL users and non-signers by overlaying subtitles or translating text to spoken language.

Applications of this technology could include facilitating communication in educational and healthcare settings, as well as in customer service and business contexts. Such advancements have the potential to significantly improve the lives of ASL users and contribute to a more inclusive society.

# 8 Conclusion

In conclusion, our project has demonstrated the efficacy of employing deep learning models for ASL recognition, specifically by testing three different models with varying architectures and parameters.

The development of a full-stack website with real-time gesture recognition, visual display of recognized labels, and Text-to-Speech API integration showcases the potential of deep learning models in creating accessible and user-friendly applications for the deaf and hard-of-hearing community. By leveraging advanced techniques in deep learning and computer vision, we can contribute to building more inclusive communication solutions and enhance the overall quality of life for individuals with hearing disabilities

# References

[1] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE. pp. 770–778.

[2] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv. https://doi.org/10.48550/arXiv.1602.07360

[3] KASAPBASI, Ahmed Elbushra, Ahmed AL-HARDANEE, Omar YILMAZ, Arif. (2022). DeepASLR: A CNN based Human Computer Interface for American Sign Language Recognition for Hearing-Impaired Individuals. Computer Methods and Programs in Biomedicine Update. 2. 100048. 10.1016/j.cmpbup.2021.100048.

[4] ‘A Real-Time American Sign Language Recognition System using Convolutional Neural Network for Real Datasets’, Rasha Amer Kadhim and Muntadher Khamees TEM Journal. Volume 9, Issue 3, Pages 937-943, ISSN 2217-8309, DOI: 10.18421/TEM93-14, August 2020.

[5] Ramzan, Farheen & Khan, Muhammad Usman & Rehmat, Asim & Iqbal, Sajid & Saba, Tanzila & Rehman, Amjad & Mehmood, Zahid. (2019). A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer’s Disease Stages Using Resting-State fMRI and Residual Neural Networks. Journal of Medical Systems. 44. 10.1007/s10916-019-1475-2.

[6] Chandra, Mahesh & Lall, Brejesh. (2021). A Novel Method for CNN Training Using Existing Color Datasets for Classifying Hand Postures in Bayer Images. SN Computer Science. 2. 10.1007/s42979-021-00450-w.