COMP9444 Project Summary

American Sign Language Image Classification

Group 0

Oguzhan Kaan Oguz - z5342510

Priyal Jain - z5512958

Aarushi Bhasin - z5522378

Chen Wang - z5447096

Daniel O'Connell - z5363291

I. Introduction

In the contemporary digital era, effective communication is crucial for social interaction and access to information. However, for the deaf and hard of hearing community, traditional forms of communication can pose significant challenges. By planning, implementing, evaluating and presenting two deep learning models, designed to classify images of the American Sign Language alphabet into their corresponding textual representations, the project aims to create a versatile and effective tool that can be integrated into various digital platforms, ultimately bridging the communication gap and promoting inclusivity in digital interactions for those who rely on sign language.

This project has a wide range of applications. Sign language learning and teaching can be made easier with the integration of this technology into educational tools. Additionally, with further development and improvement on top of the methods we have implemented, our models can be used for real-time translation in digital communication platforms, such as video conferencing tools and messaging apps, which enables seamless communication between users who are familiar with sign language and those who are not. Our contributions lie in the development and analysis of two distinct neural network models. The first is a Convolutional Neural Network (CNN). The second is a standard Neural Network.

By implementing and deploying the aforementioned two methods, the project aims to achieve higher accuracy and efficiency in sign language recognition, contributing to the development of more effective and inclusive communication tools for the deaf and hard of hearing community.

II. Literature Review

The field of digital Sign Language Recognition (SLR) includes various techniques, broadly categorised into vision-based and sensor-based modalities, as outlined in research by Adeyanju et al [1]. Vision-based methods utilise cameras to capture sign language, focusing on features like palms, finger positions, and joint angles. Notably, 3D Convolutional Neural Networks (3D CNNs) process temporal data, capturing both spatial and temporal aspects without requiring special user equipment. However, 3D CNNs demand high computational power and extensive, high-quality 3D data, posing accessibility and cost challenges.

Sensor-based approaches, contrastingly, use wearable devices like sensory gloves, as explored by Shi et al [2]. These devices, often incorporating Inertial Measurement Units (IMUs), converting hand movements into electrical signals for accurate hand posture recognition. Despite providing detailed hand movement data, IMUs face limitations like drift errors, susceptibility to environmental factors, and the need for

frequent calibration. Both IMUs and 3D CNNs are extremely costly and IMUs are challenging to integrate into digital environments due to their reliance on physical sensory inputs and high-quality equipment.
In this context, our 2D CNN model, while less adept at capturing temporal dynamics compared to 3D CNNs, offers a more feasible solution given its lower computational and data processing requirements. This approach represents a balance between effective SLR and the constraints of our resources, highlighting the ongoing evolution in the field of digital SLR with an emphasis on accessibility.

III. Methods

We selected two approaches for our methods, from the many concepts in the image classification field for the following two reasons. First, given our limited project timeframe and recent introduction to these advanced concepts as undergraduates, we wanted to balance the project's complexity with our learning capabilities and the quality of results achievable in the available time. Second, our choice was also influenced by our limited access to high-level computing power and specialised hardware that are essential for building more advanced image classification models and solutions which is again a constraint due to our positions as undergraduates.

Our first approach is a custom architecture CNN, NetConv, trained on images of sign language gestures. A diagram of this model's architecture can be seen in Figure 1. Our second approach is a standard neural network, KeyPointsNet, trained on key-point coordinate values extracted from the same images used for the CNN training. A diagram of this model's architecture can be seen in Figure 2.

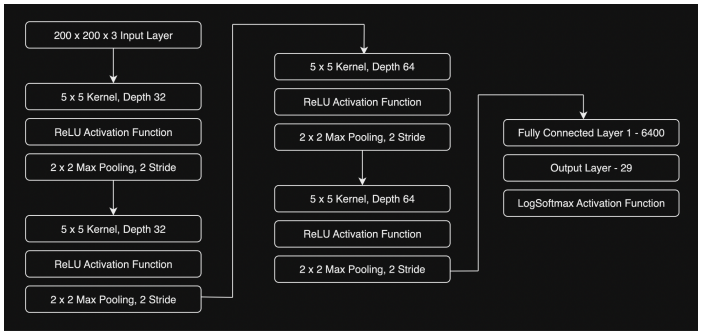

Figure 1. CNN Model Architecture Diagram

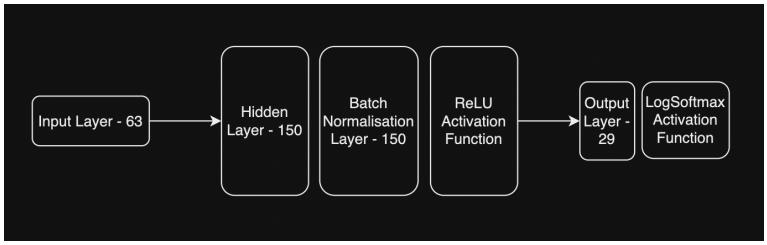

Figure 2. KeyPointsNet Model Architecture Diagram

IV. Experimental Setup

Our data exploration revealed key criteria for optimal model performance, necessitating a large dataset with image sizes up to 300 x 300 pixels for manageable training times and limited computing resources. The images also needed minimal pre-processing to maintain transferability and robustness, essential for real-world scenarios. While the Sign Language MNIST [3] dataset initially seemed promising, its heavy processing made it unsuitable for real-world application. We then discovered the ASL Alphabet [4] dataset, particularly the "asl_alphabet_train" folder, containing 29 folders with 3,000 minimally processed, 200 x 200 pixel colour images per folder, in various lighting conditions. Despite meeting our criteria, the dataset's challenge included overly dark images and limited background variation. Due to its large size of 87,000 images, we adopted a selective sampling strategy, retaining every third image, reducing the dataset to a more manageable 29,000 images. However, a key weakness was the oversimplified representation of the letters J and Z, which requires movement in ASL. The dataset's static images for these letters posed challenges in real-world gesture classification, as the models struggled with untrained movement parts.
For training the CNN, the raw set of 29,000 images was split using the 80% training and 20% testing rule to evaluate this model's learning and was converted into tensors, scaling pixel values of all channels to a

range between 0 and 1. These were then normalised per channel to a mean of 0.5 and a standard deviation of 0.5, ensuring the final input values ranged between -1 and 1. As for training the KeyPointsNet model, the MediaPipe [5] library was used to extract 63 keypoint co-ordinate values for all images in all the folders where the keypoint extractor could detect keypoints. This was to ensure we wouldn't need to extract these keypoints every time we changed our design, allowing us to reduce program run time for training sessions. We chose the MediaPipe library for extracting keypoints in hands due to its robust set of pre-built calculators tailored for image and media processing tasks. The values were then normalised using MinMax scaling with values -0.5 and 0.5 and again the total keypoints dataset was split using the 80% training and 20% testing rule to evaluate our model's learning.

To establish the training environment for both models, we adhered to lecture guidelines and assignment examples, employing Stochastic Gradient Descent as our optimiser with a learning rate of 0.01 and a momentum of 0.95, which yielded excellent results. It's important to note that for the CNN model's training, we utilised Kaggle's GPUs due to the high computational demands, as our local machine CPUs were inadequate. We initially set our training at 10 epochs, later adjusting to an optimal 13 epochs after several trials. Our approach used cross-entropy loss, a standard in classification problem-solving, and we monitored model performance by evaluating it on our test dataset and generating a confusion matrix for analysis accompanied by the percentage of correct classification and loss value for the test set classifications.

V. Results

Key Experimental Results:

1. Training Time: The keypoints model demonstrated efficiency, taking only 3 seconds to train on a local machine's CPU, while the CNN model took 13 minutes with Kaggle's GPUs.
2. Accuracy: The keypoints model achieved a final epoch test accuracy of 97%, slightly better than the CNN model.
3. Inference Strengths: The CNN model successfully classified gestures for letters A, B, and C and included a class for images with no sign language gestures.
4. The CNN model had a total inference time of 9.736 ms compared to the keypoints model which took 26.224 ms due to MediaPipe taking 26.164 ms to extract key points.
5. Inference Limitations: The CNN model struggled with distinguishing extremely similar gestures, making errors in classifying letter N as M and misinterpreting dynamic gestures like J.
6. Pre-processing: The keypoints model required a more complex pre-processing step, involving the extraction of keypoints and saving them as separate CSV files, making it more tedious compared to the CNN model.
7. Performance Discrepancies: Challenges arise in the KeyPointsNet model, accurately classifying images for certain letters, for example C and D, due to difficulties in key point extraction, and the model struggles to classify images without discernible keypoints.

The KeyPointsNet model demonstrates high accuracy when keypoints are successfully detected. Notably, it performs well in distinguishing between the ASL gestures for letters I and J, most likely due to its sensitivity for subtle differences in keypoint co-ordinate values even though it cannot process temporal data. However, challenges in the dataset and the model's inability to handle images without distinguishable keypoints highlights a significant limitation of the model and suggests areas for improvement, particularly in refining its sensitivity to diverse hand configurations.

The CNN model shows effectiveness in correctly classifying gestures for letters A, B, and C, highlighting its capability to capture and interpret distinct features. Additionally, it excels in classifying images with no sign language gestures, providing an advantage over the KeyPointsNet model in scenarios with no

keypoints for extraction. However, the CNN model exhibits limitations in distinguishing extremely similar gestures, leading to misclassifications, such as confusing the gestures for letters N and M.

The proposed models, with their unique strengths and limitations, contribute to the ongoing exploration of effective solutions for ASL image classification.

VI. Conclusions

The project makes significant contributions by implementing and evaluating two models, namely the keypoints and CNN model, for American Sign Language (ASL) gesture recognition. The keypoints model demonstrates notable efficiency and accessibility, achieving a 97% final epoch test set accuracy and proving highly effective with minimal computational resources. On the other hand, the CNN model, while computationally demanding, also exhibits a strong final epoch test set accuracy of 97%, also proving high effectiveness in recognizing various ASL gestures correctly.

The limitations of the current study are that the keypoints model faces difficulties in accurately classifying certain gestures (e.g., classes C and D) due to limitations in key point extraction. Our primary challenge stemmed from limitations within the dataset. The notable issue was the insufficient number of keypoints images for each class, primarily due to a scarcity of images featuring distinguishable fingers and palm areas, despite the dataset's initially substantial size.In addition, the CNN model struggles with nuanced differences between similar gestures. Its relatively simple architecture and inability to process temporal data is another key limitation.

Our challenges highlight opportunities for improvement with additional time and resources. Originally, we aimed to use our models for dynamic classification, but trials revealed that this approach was too time-consuming, computationally demanding and often caused our local machines to freeze, leading us to change our focus to static classification. Recognising these limitations in handling dynamic gestures, exploring 3D CNNs is recommended, but with awareness of potential overfitting, increased computational demands, and the need for specific volumetric data. Moreover, incorporating additional convolutional layers in the CNN model and adopting a smaller kernel size would improve the architecture's capacity to capture finer details, consequently enhancing its ability to discern smaller features and ultimately elevating overall inference performance.Ultimately, with further refinement in datasets, exploration of advanced techniques, and potential model enhancements, both the keypoints and the CNN model could contribute more effectively to real-world ASL prediction scenarios.

Reference

[1] Adeyanju, I. A., Bello, O. O., & Adegboye, M. A. (2021). Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, *12*, 200056.

[2] Shi, G., Li, Z., Tu, K., Jia, S., Cui, Q., & Jin, Y. (2013). Sign language translation system based on micro-inertial measurement units and ZigBee network. Transactions of the Institute of Measurement and Control, 35(7), 901-909.

[3] tecperson (2017). Sign Language MNIST. https://www.kaggle.com/datasets/datamunge/sign-language-mnist

[4] Akash Nagaraj (2018). ASL Alphabet. *https://www.kaggle.com/datasets/grassknoted/asl-alphabet*

[5] F. Zhang, C.-L. Chang, M. G. Yong, J. Lee et al. (2019). MediaPipe: A framework for building perception pipelines, arXiv:1906.08172