Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 00 (2023) 1–19
Computer Vision

Optimized ResNet Framework for *Multilingual* Sign Language Detection

Yehudit Brickner[a], Or Haim Anidjar[a,b,c,d,*]

[a]*School of Computer Science, Ariel University, Golan Heights 1, 4077625, Ariel, Israel.*
[b]*Ariel Cyber Innovation Center, Ariel University, Golan Heights 1, 4077625, Ariel, Israel.*
[c]*Kinematics and Computational Geometry Lab (K&CG), Ariel University, Golan Heights 1, 4077625, Ariel, Israel.*
[d]*Data Science and Artificial Intelligence Research Center, Ariel University, Golan Heights 1, 4077625, Ariel, Israel.*

Abstract

This paper presents a method for accurately classifying the static letters of both the American and Israeli sign language using transfer learning of a Convolutional Neural Network (CNN), specifically ResNet. The proposed approach outperforms a state-of-the-art method for the same problem on American sign language letters. To evaluate the effectiveness of the proposed method, two datasets were used. The first dataset included larger and fewer images of approximately 45 different people using sign language, while the second dataset was a cropped version of the first dataset. The proposed model achieved an accuracy of 94% on the first dataset and an impressive accuracy of 99.65% on the second dataset after performing data augmentation techniques. These results demonstrate the potential of the proposed method for accurately classifying the static letters of American and Israeli sign language and bridging the communication gap between individuals who rely on sign language as their primary mode of communication, and people who do not know sign language. Moreover, the two datasets are real-world ones, with high variability in terms of the number of individuals and skin shades.

Keywords: Convolutional Neural Networks, ResNet, Hard Of Hearing, American Sign Language, Israeli Sign Language, YOLO.

1. Introduction

Sign Language is a language that uses hand gestures Holdengreber et al. (2021), facial expressions, and body movements Kosmidou et al. (2011); Elsayed et al. (2022); Shin et al. (2021), where a critical part being finger spelling. It is primarily used by people who are deaf or Hard Of Hearing (HOH) and their family members Padden & Gunsauls (2003). Additionally, it is also used by people who are mute or have difficulty speaking Salvin et al. (1977). According to the United Nations (UN[1]), there are over 300 different sign languages used around the world. Some of the better-known sign languages that are used in a lot of scientific research papers are American Sign Language (ASL) Shin et al. (2021); Chuan et al. (2014); Abu-Jamie & Abu-Naser (2022); Zhang et al. (2021); Mannan et al. (2022); Kumar et al. (2021); Ma et al. (2022); Prateek et al. (2018), British Sign Language Deuchar (2013); Liwicki & Everingham (2009); Mocialov et al. (2020), and Indian Sign Language Ekbote & Joshi (2017); Hore et al. (2017); Ghotkar et al. (2012). Understanding sign languages and their structure is crucial for enhancing communication and ensuring accessibility for people who use these languages as their primary means of communication. In this paper, we aim to explore

*Corresponding author: Or Haim Anidjar, School of Computer Science, Ariel University, Golan Heights 1, 4077625, Ariel, Israel.
Email addresses: diti.yb@gmail.com (Yehudit Brickner), orhaim@ariel.ac.il (Or Haim Anidjar)
[1]Please visit the UN website.

and analyze the structure of sign language fingerspelling and propose methods for improving their recognition and interpretation using machine learning techniques.

ASL is primarily used in the United States and Canada, while in other countries where English is the predominant language, such as England and Australia, they use British Sign Language Deuchar (2013) and Australian Sign Language Johnston & Schembri (2007), respectively. The reason for these differences is that sign languages were created independently of each other or branched off of other sign languages that were created independently Johnston (2003); Stokoe Jr (2005). Each sign language has its own unique grammar and syntax, which may differ from the grammar and syntax of spoken languages. For example, both ASL and Israeli Sign Language (ISL) Meir & Sandler (2007) have different grammar and syntax than English Janzen & Shaffer (2002) and Hebrew Meir (1999), respectively.

According to the World Health Organization (WHO[2]), there are 1.5 billion people globally with hearing loss, and this number is expected to increase in the future. Additionally, 70 million people worldwide use sign language as their primary form of communication, according to the United Nations. Therefore, there is a pressing need to bridge the communication gap in sign language, just as there is a need to bridge gaps in other languages.

Hearing loss affects people in different ways, and not all deaf people have the same level of hearing loss Kosmidou et al. (2011). There are five levels of hearing loss: (i) mild, where the person cannot hear anything quieter than 20-40 decibels; (ii) moderate, where the person cannot hear anything quieter than 41-55 decibels; (iii) moderately severe, where the person cannot hear anything quieter than 56-70 decibels; (iv) severe, where the person cannot hear anything quieter than 71-95 decibels; and (v) profound, where the person cannot hear anything quieter than 95 decibels.

The average human speech has a volume of about 60 decibels Karpf (2011), meaning that people with level (iii) hearing loss or higher may have difficulty hearing conversations. To help with hearing, people who are deaf or HOH may use hearing aids or cochlear implants. However, even with these devices, hearing can be challenging and may require a lot of concentration and lip-reading to understand spoken language Desjardins & Doherty (2014); Loizou (1999). Therefore, deaf people may use sign language as a more accessible and expressive way of communication Kyle et al. (1988).

In 2014, approximately 6% of the population in Israel was deaf or HOH, with 500,000 HOH people and 7000 deaf people Tannenbaum-Baruchi et al. (2014). This number is expected to increase according to a report by the World Health Organization (WHO) on hearing loss. For those who use sign language as their primary form of communication, Israeli Sign Language (ISL) is often used Tannenbaum-Baruchi et al. (2014). ISL uses the Hebrew alphabet for finger spelling, which consists of 23 normal letters and 5 final letters Treiman et al. (2007). To represent each of these letters, ISL uses 28 unique hand gestures. Figure 1 illustrates the sign for each Hebrew letter in ISL, along with the corresponding phonetic sound in English. Interestingly, ISL and American Sign Language (ASL) share many similarities in their signs, with many of the letters looking exactly the same or very similar.

People who use sign language as their main form of communication often have difficulties communicating with people who do not know sign language. This research helps to solve this issue and bridge the communication gap, by using a Convolutional Neural Network (CNN) Albawi et al. (2017); O'Shea & Nash (2015); Wu (2017) to translate images of sign language letters into written language letter by letter. The main objective of this research paper was to create a classifier for the ISL alphabet that can manage to classify real-world images, that are diverse and with all different backgrounds.

A CNN is a type of deep learning algorithm used for identifying patterns in images. The main component of CNNs is convolutional layers, that can find patterns in images O'Shea & Nash (2015). There are many CNN models that exist such as ResNet Wu et al. (2019); Tai et al. (2017); Targ et al. (2016) and VGG Simonyan & Zisserman (2014); Dutta et al. (2016); Tammina (2019), each with their own unique architecture. Transfer learning is a good way to utilize an existing model, by using the weights of a pre-trained network for the initialization of the network before training on a new dataset.

In Abu-Jamie & Abu-Naser (2022), the authors used a ResNet architecture to classify ASL letters from a Kaggle dataset Mavi (2020). Their data contains a lot more images, with all of them having a plain background and the hand in the center of the image, making it easier to classify those images. Also, all of the images appear to be of the same person. On the other hand, the dataset used in this paper is from over 45 different people, with a lot fewer images per class, and yet our architecture achieved better results.

[2]Please visit the WHO website.

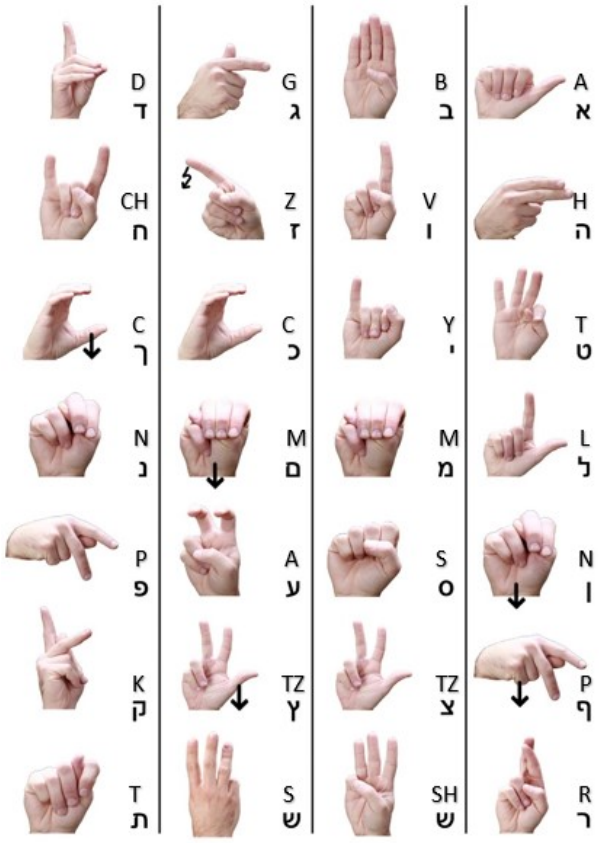


Figure 1. Illustration of ISL alphabet; next to each sign, is the Hebrew letter and its corresponding phonetic sound in English. The illustration was edited from Levana sign language Website.

In this paper, we aim to tackle the problem of sign language recognition and translation, specifically the classification of the ISL alphabet. In section 5, we will present a comprehensive evaluation of our proposed approach. To demonstrate the effectiveness of our method, we will compare our results with the ones achieved by the approach presented in Abu-Jamie & Abu-Naser (2022) on the very same problem. Our results show that our approach outperforms the previous approach in terms of accuracy Yozevitch et al. (2014) and robustness to diverse and complex image backgrounds. Next, we present the contributions of this paper as presented in Section 1.1.

1.1. Our Contribution

In this work, we make the following contributions:

- **Real-life dataset -** We demonstrate that our proposed models achieve very good accuracies despite using a diverse set of images from over 45 different people. The dataset contains images of both male and female hands with varying skin colors and with accessories such as nail polish, rings, watches, and bracelets. Additionally, the dataset includes images of both the right and left hands, making it more challenging for the models.
- **Israeli Sign Language -** We classify Israeli Sign Language letters, a sign language that has not received much attention in the literature. Although ISL has similar letters to ASL, it is a distinct language, and our proposed approach can be extended to other sign languages with similar letter forms.
- **Small dataset and the YOLO algorithm -** We use the YOLO Jiang et al. (2022) algorithm to crop images around the hands to remove background noise, and this helps to make the dataset bigger. Despite having a relatively small dataset, our approach achieves high accuracies, demonstrating its robustness to noise and variations in the input data.

- **Augmentations Exploitation -** We employ various data augmentations techniques while training our models, which helps to make them more robust and achieve higher accuracies. Our results show that data augmentation is an effective way to mitigate overfitting and improve model generalization.

1.2. Paper Structure

The remainder of this paper is structured as follows: Section 2 surveys related work regarding the classification of letters or numbers in sign languages. Section 3 discusses the datasets used in this paper and the pre-processing procedure. Section 4 presents the approach employed in this paper of classifying letters in ISL. Section 5 presents the process made to get to the final model, including running the model on augmented data. Finally, Section 6 concludes the paper and offer some directions for future work. For ease of reading, Table 1 provides a list of abbreviations that are commonly used in this paper.

Abbreviation	**Meaning**
ASL	American Sign Language
C	Clockwise
CC	Counter-Clockwise
CNN	Convolutional Neural Network
FC	Fully Connected
HOH	Hard Of Hearing
ISL	Israeli Sign Language
KNN	K Nearest Neighbors
NN	Neural Network
SVM	Support Vector Machine
UN	United Nations
WHO	World Health Organization
YOLO	You Only Look Once

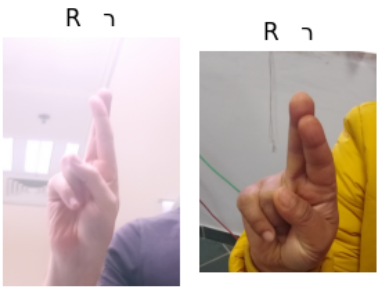
Table 1. List of Abbreviations.

2. Related Work

Previous studies have employed various machine learning algorithms such as Support Vector Machines (SVM) Jakkula (2006); Noble (2006); Wang (2005) to classify sign language letters. However, these models generally achieved lower accuracy than current state-of-the-art results with neural networks. With the emergence of Convolutional Neural Networks (CNNs), which are designed to detect patterns in images, most recent research in image classification of sign language letters has focused on utilizing CNNs Jordan & Mitchell (2015); Zhou (2021). Most of these previous studies on the classification of sign language letters have employed datasets that do not resemble real-world data. These datasets are limited to a small number of individuals with the same skin color, and the images mostly focus on the hand with little background or noise. Furthermore, the majority of the datasets only include images of the right hand and do not consider the left hand.

Real-life data presents certain difficulties that are not encountered in laboratory data. Each person signs differently, as can be observed in Figures 2, where the sign for the letters *M* (English) and *Mem* (Hebrew) is executed with the thumb placed under the first three fingers, not under the pinky finger. Depending on the hand size, the thumb can be seen over the pinky. Another example is the sign for the letters *R* (English) and Reish (Hebrew), which involves crossing the middle finger over the pointer finger. However, not everyone can get their middle finger completely over their pointer finger. These slight variations between individuals' signing styles introduce an additional level of complexity to the dataset. Moreover, people finger-spell with their dominant hand; thus, the models must be able to classify signs from both left and right hands. This can cause letters that look very different when signed with the same hand to suddenly appear very similar. For example, the letters D (English) and Vav (Hebrew) resemble *I* (English) and *Yud* (Hebrew) when they are signed with opposite hands. Humans can quickly distinguish between them, but this task poses a greater challenge for a Neural Network (NN).

(a) Comparison of hand signs for the letter *M*, showing the variation in thumb placement between different signers.



(b) The sign for the letter *R*, with a clear distinction between the left image where the middle finger is crossing over the pointer finger, and the right image where the middle finger is barely over the pointer finger.

(c) The left-hand sign is the letter *D*, whereas the right-hand sign is the letter *I*. Although they have a similar shape, humans can differentiate them by the finger standing up. However, it is challenging for computers to distinguish between them.

Figure 2. Real-life dataset: The dataset used in this study represents real-life data, which poses several challenges for the classification of sign language letters. Each person signs differently, resulting in small variations in hand gestures that add complexity to the dataset. Furthermore, people finger-spell with their dominant hand, making it necessary to classify signs from both left and right hands. These differences in hand shape and orientation make it challenging for a computer model to accurately classify the letters.

2.1. Classic Machine-Learning for Sign Language

Among advanced previous studies that have employed classic machine-learning techniques for the sign-language identification problem, one can find the following ones: Shin et al. (2021); Chuan et al. (2014); Ekbote & Joshi (2017); Pugeault & Bowden (2011);

As reported in Shin et al. (2021), the authors used three different datasets, including Kaggle Mavi (2020), Massey Shin et al. (2021) and the Finger spelling dataset Aryanie & Heryadi (2015), to classify 26 ASL letters. They utilized mediapipe hands Zhang et al. (2020) to extract 190 features that correspond to distances between joints and 630 features that correspond to the angles between the joints. They trained an SVM and Light Gradient Boosting Machine Taha & Malebary (2020); Alzamzami et al. (2020) on the extracted features and achieved their best results on the Massey dataset with both feature types. Both the SVM and Light Gradient Boosting Machine models achieved accuracy rates of 99.39% and 97.8%, respectively, and performed best on all features.

Chuan et al. Chuan et al. (2014) classified 26 ASL letters using a Leap Motion sensor Guzsvinecz et al. (2019) to obtain 3D information about hand and finger positions. They utilized SVM and KNN Zhang & Zhou (2007); Zhang et al. (2017); Guo et al. (2003) classifiers on these features. The SVM achieved 79.83% accuracy, while KNN achieved 72.78%.

In another study, Ekbote et al. Ekbote & Joshi (2017) addressed the classification of Indian Sign Language digits from a dataset of 1000 images, containing 100 images per class, captured from 10 people. The authors converted RGB images into YCbCr color space Ahirwal et al. (2007) and utilized three different feature extraction techniques, namely shape descriptors, Scale Invariant Feature Transform Lindeberg (2012), and Histogram of Oriented Gradients Tomasi (2012). They trained four SVM models on different features and a fully connected neural network model with two hidden layers. Results showed that the model trained on only the Histogram of Oriented Gradients features outperformed other models with 99% accuracy. Interestingly, the fourth model that used all the extracted features performed worse than the third model that only used the Histogram of Oriented Gradients features, demonstrating the potential negative effect of adding more features to the model.

In a study by Pugeault and Bowden Pugeault & Bowden (2011), a dataset of over 48000 images, consisting of 24 letters in ASL, excluding J and Z, was classified using Microsoft Kinect. Depth images were captured from different angles and viewpoints, representing real-life data, and each letter class had over 500 samples. They used a random forest Breiman (2001); Biau & Scornet (2016); Chuan et al. (2014) model to classify the images, creating three different models: intensity, depth, and a combination of both. The accuracy results were 73%, 69%, and 75%, respectively.

2.2. CNN-based approaches for Sign Language

2.2.1. ResNet-based approaches

Various studies have utilized convolutional neural networks (CNNs) for sign language recognition. These studies include Abu-Jamie & Abu-Naser (2022); Zhang et al. (2021); Mannan et al. (2022); Kumar et al. (2021); Ma et al. (2022); Prateek et al. (2018). The methods employed in these studies involve different types of CNNs and image pre-processing techniques to classify images of ASL.

In their work on sign-language detection, several studies employed ResNet models, including Abu-Jamie & Abu-Naser (2022); Zhang et al. (2021); Ma et al. (2022). The ResNet models have various numbers of layers, with the smaller to medium-size models having 18, 34, 50, 101, and 152 layers.

In particular, Abu-Jamie & Abu-Naser (2022) utilized a pre-trained ResNet50 model on ImageNet Recht et al. (2019) to classify all 26 letters of ASL and three additional classes: 'delete', 'space', and 'blank'. Their experiments used a subset of the Kaggle dataset consisting of 1,000 images per class, out of the total of 3,000 images per class. They achieved their best result with an accuracy of 99.87

In Zhang et al. (2021), the authors aimed to classify 24 out of the 26 letters in ASL, excluding *J* and *Z* due to their dynamic signs. They utilized the ASL fingerspelling dataset, which consisted of 65,774 images, with approximately 2,750 images per class. Since the size of the dataset was relatively small and the images were complex, they increased the number of images by augmenting the original images with rotations and Gaussian noise. This resulted in almost doubling the size of the dataset to 12,000 images. They employed the ResNet18 model and achieved a 99% accuracy in their classification task.

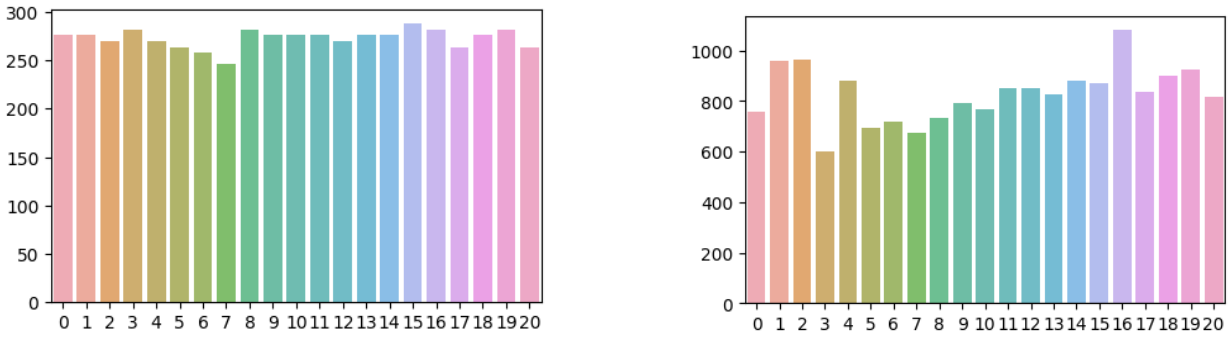
In their work,Ma et al. (2022) aimed to classify all 26 letters of ASL, as well as three additional classes, namely, 'delete', 'space', and 'blank'. They collected images from the Kaggle and MNISTCohen et al. (2017) datasets, which resulted in a total of 121,627 images. To further augment their dataset, they applied various transformations, such as rotation, scaling, and translation. They utilized four different CNN models, namely, ResNet18 Shen et al. (2019), ResNet50 Rezende et al. (2017), AlexNet Shanthi & Sabeenian (2019), and LeNet Xu et al. (2022). In order to improve the classification of the letters J and Z, which have dynamic signs, they trained eight networks using the aforementioned models with and without a two-stream mixed method. This method involves combining two images of the same class, such that if the sign is dynamic, both images can be seen in the new image, and if the sign is not dynamic, the two images almost totally overlap and look like one image. Their best result was achieved using the ResNet50 Two-stream mixed method, with an accuracy of 97.57%, as well as a model that made the fewest errors for the letters *J* and *Z*.

2.2.2. Classic CNN-based approaches

Regarding classic CNN architectures for sign-language detection, Prateek et al. Prateek et al. (2018) classified all 26 letters of ASL. They created their own dataset using 32,400 images extracted from videos of people signing. The images selected had minimal background noise, and skin tones were filtered to produce images with a black background and a white hand. Canny edge detector Rong et al. (2014) was then applied to detect the outline of the hand in each image. These new images were used to train their CNN model, which achieved the best accuracy of 98.66%. As the final images used for training contained less noise, it is speculated that this contributed to their high accuracy results.

The work of Mannan et al. (2022) focused on classifying 24 out of 26 letters in ASL, with the exclusion of *J* and *Z* due to their dynamic signs. They employed the MNIST Cohen et al. (2017) dataset containing 35,000 images and augmented the data with rotations and scaling techniques during the training of their CNN model to improve its robustness. Their CNN architecture comprised convolutional, max pooling, Fully-Connected (FC), and dropout layers. The CNN consisted of three convolution blocks, each containing a convolution layer and a max pooling layer. The output of the third convolution block was flattened and fed into the FC layers, which comprised of three FC layers, with a dropout layer used after the second FC layer. The final FC layer was used for classification, and they achieved an accuracy of 99.67%.

In another related work, Kumar et al. Kumar et al. (2021) addressed the problem of classifying all the letters in ASL. To do so, they constructed their own dataset, which included around 35,000 images. To improve the size of their dataset, they employed several data augmentation techniques, such as rotations, shifts in height and width, random zooming, and flips. They also designed a CNN model with 3 convolutional blocks and 2 fully-connected layers.



(a) Original dataset, images per class histogram.

(b) Cropped dataset, images per class histogram.

Figure 3. ISL dataset class-distribution.

Specifically, the first and third convolutional blocks contained convolutional batch normalization and max pooling layers, while the second convolutional block included a dropout layer. The output of the third convolutional block was flattened and used as the input for the first fully-connected layer. After applying a dropout layer, they used the final fully-connected layer to classify the images. Their proposed model achieved an accuracy of 99.63%.

Finally, and based on our understanding, our study focuses on the most recent and cutting-edge approach to the sign-language identification problem, as introduced in the study by Abu et al.Abu-Jamie & Abu-Naser (2022). We provide a detailed comparison of our proposed method with this state-of-the-art approach in Section5.

3. Dataset

The initial dataset contained around 45 images per class, which were augmented to generate five additional images each. The first transformation involved flipping the images along their Y-axis to create a mirror image, thereby representing people signing with both hands instead of just the right hand. This was also useful for classifying images taken by front-facing cameras that automatically flip the image to its mirror image. The second to fifth transformations involved rotating the original and flipped images randomly between 1-25 degrees clockwise and counter-clockwise, to increase the amount of data and train the models on images that are not at the perfect angle. This process resulted in a dataset of approximately 5,670 images, with around 270 images per class. The distribution of each letter in the dataset is displayed in Fig. 3a, indicating that the letters are distributed evenly. This augmented dataset will henceforth be referred to as the *'original dataset'*.

The initial dataset had images with a lot of background noise surrounding the hand. To crop and center the images around the hand, the "You Only Look Once" (YOLO) Jiang et al. (2022) algorithm was employed, which is available in the YOLO-Hand Reposirtory. The images were passed through YOLO four times with different buffer sizes of 700, 400, 200, and 100 pixels. However, the algorithm was not always successful in detecting the hand, as it sometimes found ears or faces instead. The cropped images were manually verified to ensure that the majority of the hand was present, and five additional images were created for each cropped image using the same augmentation process as the original dataset. The final size of the cropped dataset is approximately 17,382 images, with about 700-900 images per class. Note that there is a difference in the size of each class due to the YOLO algorithm's variable success rate. Fig 3b shows the distribution of each letter in the cropped dataset, which is close to the average number of instances per class, except for classes 3 and 16. This dataset will be referred to as the *'cropped dataset'*.

In this study, the datasets used for the experiments in Section 5 were split into 80% for training, 10% for validation, and 10% for testing. Table 2 provides a detailed summary of the number of images in each dataset.

4. Framework Proposed

As previously mentioned, the focus of this work is on detecting sign language, specifically both ASL and ISL. In the case of ISL, some letters, namely the final letters, were not classified due to their dynamic nature - the sign for these letters requires movement and cannot be represented with a single image. The letter *Zain* (i.e., *Z*) was also

Dataset	**Original**	**Cropped**
Training	4,584	13,905
Validation	573	1,739
Testing	573	1,738

Table 2. Datasets size used in Section 5.

not classified for the same reason. Additionally, the letter Sin (shown in the bottom row, column 3 of Fig 1) was not classified, as it makes a sound that is rarely used in spoken language, and in written Hebrew it looks the same as the letter Shin (shown in the bottom row, column 2 of Fig 1), which is used for both letters.

This study utilized pre-trained ResNet models from PyTorch to develop the deep learning framework. Specifically, ResNet18, 34, 50, 101, and 152 models, which had been previously trained on the ImageNet Recht et al. (2019) dataset, were selected for this purpose. The approach of using pre-trained neural networks that are further trained on new data is known as transfer learning. There are two primary benefits to using transfer learning in many projects. First, transfer learning can decrease the number of epochs required to train the model Chelghoum et al. (2020). Second, it can help mitigate the challenge of working with small datasets Weiss et al. (2016).

Next, pre-trained ResNet models from PyTorch were used as a starting point for transfer learning. The imported models had a final FC Albawi et al. (2017) layer with an output of 1,000 classes since they were trained on the Imagenet dataset with 1,000 classes. However, since the current project had only 21 classes, the output of the final FC layer for all the NN models used in this paper was changed to 21. Some models were also experimented with having three FC layers to improve the classification results (discussed in detail in this section, as well as in Section 5).

The reason for using CNNs in this paper is that the convolutional layers are the main component of these models. The first layers in the CNN identify small patterns in the input image, such as vertical and horizontal lines. The subsequent convolutional layers combine the patterns detected in the previous layers to identify larger patterns in the image. The FC layers at the end of the CNN then use the information about the detected features and patterns to classify the input image O'Shea & Nash (2015).

The ResNets were trained on the images of the training set using the Adam optimizer Kingma & Ba (2014); Bock & Weiß (2019) with a learning rate of 0.0001 for 10 epochs. The log-softmax De Brebisson & Vincent (2015) activation function was used in the architecture instead of the softmax Jang et al. (2016) activation function. The reason for using the log-softmax activation function is that it produces an output in the range $(-\infty, 0]$, which helps to spread out the classes and simplify the image classification. During training, the model was validated after each epoch, and if the validation loss of the current epoch was lower than that of the previous epoch, the model was saved. The last saved model was loaded after the final epoch, and the test data was run on it to reduce overfitting.

In this study, the ResNets were fed with images in batches of 16. Upon loading each batch, the images were transformed into tensors. Two transformers were utilized: one for the train dataset and another for the validation and test datasets. Both transformers applied color adjustments to the images, setting their mean to $(0.485, 0.456, 0.406)$ and standard deviation to $(0.229, 0.224, 0.225)$. These mean and standard deviation values are tuples containing 3 values each, representing the RGB channels. The reason for changing the mean and standard deviation of each image is that the imported models were initially trained on ImageNet images, and these parameters correspond to those of the ImageNet dataset Dornadula et al.; Milton (2018).

The experimentation conducted in Section 5 involved applying various augmentations to the images, including Gaussian noise Mandelbrot & Wallis (1969), blur Laroche et al. (2023), rotation Khorov et al. (2022), color jitter Treu et al. (2021), and ISO Clark et al. (2021) noise. All of these augmentations, along with the color changes, were implemented using the Albumentations software package.

For each model that utilized augmentations, every combination of up to three augmentations was tested, as detailed in Tables 5, 6, 7, and 8. The augmentations were applied exclusively to the training dataset via the train loader. Rotation was applied to every image, while the other augmentations were applied randomly with a probability of 50%.

4.1. Data Augmentation

Next, we discuss each of the augmentations techniques that were in use as part of the framework presented in this paper as follows:

- **Gaussian Noise.** - This augmentation added noise to the pixel values of all RGB channels. The noise was sampled from a normal distribution with default parameters: variance limit between 10 and 50, and mean 0. The noise added to each channel was different.
- **Blurring.** - This augmentation applied a convolution matrix kernel to the image, resulting in a blurred image. The default kernel size used by the model was between 3 and 7, resulting in a square kernel of dimensions 3x3, 5x5 or 7x7.
- **Rotation.** - This augmentation rotated the image up to 15 degrees clockwise or counterclockwise. This was done to train the model to recognize the letter at different angles. The dataset already contained images rotated up to 25 degrees. With this augmentation, images could be rotated up to 40 degrees clockwise or counterclockwise. It was also possible for rotations to contradict each other.
- **Color Jitter.** - This augmentation modifies an image's brightness (makes all the pixels have a higher or lower value), contrast (difference between the lowest and highest pixel value), saturation (color intensity), and hue (color tone). The brightness, contrast, and saturation are altered by a random value between [0.8, 1.2], whereas the hue is adjusted by a random value between [-0.2, 0.2]. The default value used for this augmentation is 0.2.
- **ISO Noise.** - This augmentation introduces ISO noise to the image, which simulates the noise that occurs in an image when taken with a high ISO value in low light conditions. The ISO value controls the amount of light that enters the camera sensor, and a higher ISO value leads to more noise in the image. The default setting for this augmentation introduces ISO noise to the image.

Finally, 4 presents an illustration of the full framework suggested in this paper.

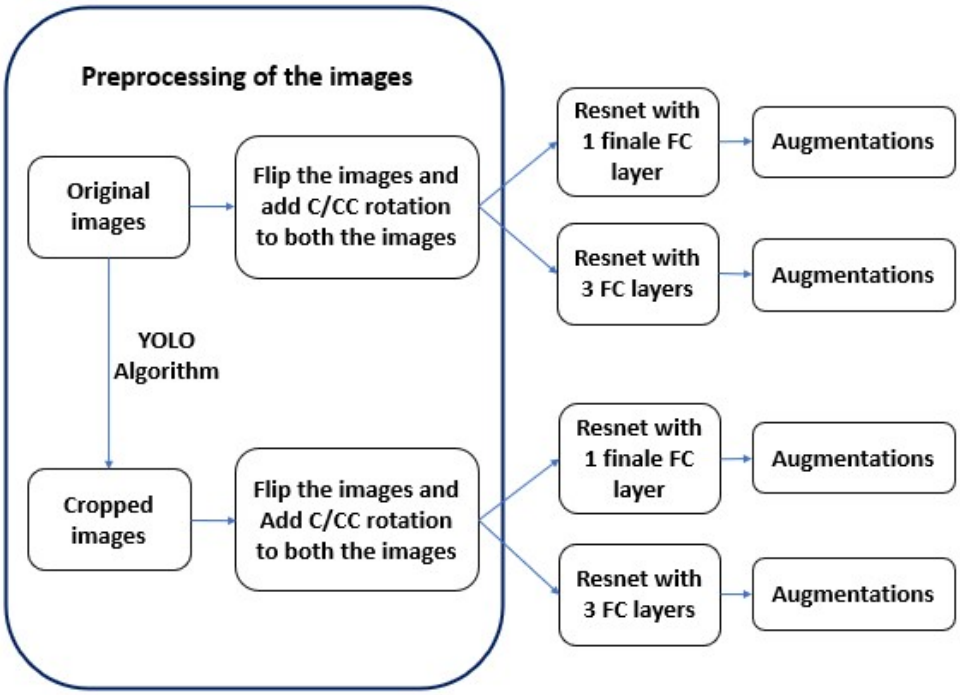


Figure 4. Depiction of the overview of the proposed framework. The pre-processing pipeline, as described in section 3, is visualized along with the experimentation process.

5. Experimental Evaluation and Results

This section aims to validate our hypothesis that incorporating augmentations and cropped images, as well as architectural modifications to the VGG, can result in improved results compared to classical CNNs.

Computational Resources. The experiments were performed on an HP Pavilion Gaming laptop 16-a0xxx, running Windows 10 64-bit OS with an Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz, 16 GB RAM, NVIDIA GeForce GTX 1650 Ti GPU, using PyTorch (v1.11.0+cu113) and scikit-learn (v1.1.1).

Finally, this section is organized as follows:

- Section 5.1 introduces how the classification performance of the trained models is assessed using well-known evaluation metrics, such as Accuracy, Precision, Recall, and F1-Score Lipton et al. (2014). These metrics are used throughout the entire experimental evaluation, which includes Sections 5.2, 5.3, and 5.4.
- Section 5.2 presents several ResNet-based model architectures for the Original and Cropped Dataset of the ISL. The goal is to determine the best model architecture that balances accuracy metrics, as discussed in Section 5.1, with computational resource consumption. We evaluate the performance of each architecture and analyze their tradeoffs.
- Section 5.3 presents the $Power - Set$ experimental set, which examines the performance of the Augmentation-based models outlined in Section 4.1.
- Section 5.4 presents a comparative analysis that showcases the superior performance of the best models proposed in this study compared to those in Abu-Jamie & Abu-Naser (2022).

5.1. Evaluation Metrics

This paper exploits the following four well-known Anidjar et al. (2023) metrics:

- **Accuracy.** gives us an indication of how many instances were correctly classified out of all of the instances. The accuracy is defined as in Eq.(1):

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

- **Precision.** gives us an indication of how many instances of a certain class were correctly classified, out of all the instances that were classified to that class. The Precision is defined as in Eq.(2):

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

- **Recall.** gives us an indication of how many instances of a certain class were correctly classified, out of all the instances from that class. The Recall is defined as in Eq.(3):

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

- **F1-Score.** this score is the harmonic average of precision and recall. If either the Precision or Recall is very low it will significantly lower the score, And if either is zero then the score is 0. Since this score is the harmonic average of precision and recall it should be between them. The F1-Score is defined as in Eq.(4):

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

In the case of imbalanced datasets, evaluating the performance of the model solely based on accuracy may not be sufficient. Metrics such as Recall, Precision, and F1-Score are more appropriate for measuring how well the model is performing. This is because high accuracy can be achieved even if one class is not predicted at all, as in the case where there are 99 instances of class A and only 1 instance of class B. However, by calculating Precision and Recall, taking into account TP (True Positive) instances of class B classified as B, both Precision and Recall would be 0, resulting in an F1-score of 0. On the other hand, when the dataset is balanced, accuracy can be a reliable metric to evaluate model performance, as the model must learn both classes to achieve high accuracy.

The datasets used in this study, namely ASL and ISL, are well-balanced. Fig. 3a shows the bar plot of the original dataset, which demonstrates that the number of instances per class is nearly equal, with only slight variation. On the other hand, Fig. 3b depicts the bar plot of the cropped dataset. Although the number of instances in each class varies more than in the original dataset, it still hovers around the average number of instances per class, with the minimum being approximately 600 and the maximum around 1100 for class 3 and class 16, respectively.

To evaluate the performance of our model, we calculated various metrics using *scikit-learn.metrics*, including accuracy_score, precision_score, recall_score, and f1_score. Since our dataset consists of 21 classes, all the scores were computed using the 'weighted' average. However, it is worth noting that using a weighted average may cause the F1-score to deviate from the average of the precision and recall scores, as indicated by scikit-learn[3].

5.2. ResNet Architectures for Original and Cropped Dataset

In this experimental section, we trained 5 ResNets on the ISL datasets to determine the best model for subsequent experiments. The ResNet models used were ResNet18, 34, 50, 101, and 152, each having 1 FC layer.

Table 3 reports the results of the first experiment on the original dataset. In Ent.(3), ResNet50, achieved the best results with an accuracy of 88.65%, considering model complexity. We selected ResNet50 as the best model for the following experiments for the following reasons:

(1). Ent.(3)'s accuracy is more than 8% higher than that of Entries (1) and (2).

(2). Ent.(3)'s accuracy is 3% higher than that of Ent.(4), and ResNet50 is less complex than ResNet101 with 50 fewer layers.

(3). Although Ent.(5) has a higher accuracy by just over 1%, ResNet152 has over 100 more layers than ResNet50, making it significantly more complex.

ResNet models **with no extra FC layers** trained on the **original** dataset					
Entry #	Model	Accuracy	Precision	Recall	F1 Score
1	Res18	80.45	81.45	80.45	80.58
2	Res34	77.48	80.88	77.48	77.96
3	Res50	**88.65**	**89.47**	**88.65**	**88.63**
4	Res101	85.86	87.25	85.86	85.83
5	Res152	89.87	91.06	89.87	90.07

Table 3. Comparison results of different ResNet models on the original dataset.

Next, the results of the first experiment on the cropped dataset in Table 4 are presented. Five different ResNets were trained, and each ResNet had 1 FC layer. The model with the best results compared to the complexity of the model is Ent.(65), ResNet50, with 99.36% accuracy. The reason that Ent.(8) was decided as the best model in this table are:

(1). Ent.(8) accuracy is higher than Entries (6)-(7), and Ent.(9) accuracy's.

(2). Ent.(10 has a higher accuracy, but ResNet152, has over 100 more layers than ResNet50, which makes it a lot more complex. This finding suggests that ResNet50 is the best model for the cropped dataset, and we will use it in the following experiments in this section.

[3]Please visit the scikit-learn website.

ResNet models **with no extra FC layers** trained on the **cropped** dataset					
Entry #	Model	Accuracy	Precision	Recall	F1 Score
6	Res18	98.84	98.86	98.84	98.84
7	Res34	98.90	98.92	98.90	98.90
8	Res50	**99.36**	**99.38**	**99.36**	**99.36**
9	Res101	99.02	99.03	99.02	99.02
10	Res152	99.65	99.65	99.65	99.65

Table 4. Comparison results of different ResNet models on the cropped dataset.

5.3. Power − Set of Augmentation-Based Models

In this section, the performance of the ResNet50 model is tested by adding augmentations during training to improve the results. The ResNet50 model was trained with various combinations of augmentations up to size three. The noise augmentations were used to enhance the noise in the images and make it easier for the models to learn how to deal with different types of noise. It was observed that all models trained with the augmentation of blurring performed worse than the others. The downsizing of the images to 224x224 may have caused essential features to be smoothed out by the blurring kernel. Therefore, the results of these models will not be presented in this section but can be found in the appendix.

5.3.1. Original ISL Dataset

The results of the second experiment on the original dataset with ResNet50, having a single FC layer of size 2048-21 with augmentations, are presented in Table 5. The best-performing model was Entry 13, the model with the augmentation of rotation, achieving an accuracy of 94%. The high accuracy can be attributed to the fact that $\frac{2}{3}$ of the dataset are images that were rotated C or CC. Hence, adding augmentations during the training helped the model perform better. It was noticed that all models with three augmentations showed poor results because they learned how to classify images with a lot of noise. Thus, when the model tried to classify clean images, it struggled.

ResNet50-Based Training on Data Augmentation and **1** FC, **original** dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
3	none	88.65	89.97	88.65	88.63
11	i	89.70	90.76	79.70	89.72
12	iii	88.48	89.08	88.48	88.51
13	iv	**94.06**	**94.55**	**94.06**	**94.01**
14	v	87.26	88.24	87.26	87.18
15	i iii	89.17	90.14	89.17	89.23
16	i iv	92.84	93.09	92.84	92.85
17	i v	87.43	88.32	87.43	87.56
18	iii iv	87.26	88.13	87.26	87.32
19	iii v	89.17	90.13	89.17	89.33
20	iv v	91.27	91.79	91.27	91.32
21	i iii iv	23.03	24.64	23.03	22.24
22	i iii v	20.24	19.71	20.24	18.96
23	i iv v	23.03	24.28	23.03	22.47
24	iii iv v	22.33	23.62	22.33	21.68

Table 5. Comparison results of the ResNet50 model with 1 FC layer on the original dataset with all the augmentations except blurring. The results of the augmentations containing blurring can be found in Appendix A.11. The augmentations are: i = Gaussian, iii = color jitter, iv = rotate, v = ISO.

The third experiment on the original dataset is presented in Table 6, where ResNet50 is trained with 3 FC layers along with augmentations. The sizes of the FC layers are 2048-1050, 1050-210, and 210-21. Among all models, Entry 35 achieved the highest accuracy of 90.92% with the augmentations of rotation and ISO. We hypothesize that the model with the augmentation of rotations was able to perform better in this experiment as well, as it helped the model to learn the rotated versions of the images.

Additionally, the ISO noise augmentation helped the model learn how to cope with noise that is present in all images. Similar to Table 5, models with three augmentations had poor results due to overfitting on the noisy images.

ResNet50-Based Training on Data Augmentation and **3** FC, **original** dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
25	none	89.17	89.93	89.17	89.16
26	i	89.87	90.29	89.87	89.80
27	iii	84.81	86.44	84.81	85.07
28	iv	89.87	90.22	89.87	89.71
29	v	85.16	86.14	85.16	84.95
30	i iii	85.51	87.18	85.51	85.53
31	i iv	89.87	90.72	89.87	89.84
32	i v	84.99	86.38	84.99	85.20
33	iii iv	88.13	89.33	88.13	88.06
34	iii v	82.72	83.93	82.72	82.52
35	iv v	**90.92**	**91.34**	**90.92**	**90.86**
36	i iii iv	45.72	47.84	45.72	45.64
37	i iii v	37.52	39.62	37.52	35.72
38	i iv v	41.71	44.30	41.71	41.00
39	iii iv v	42.93	44.04	42.93	41.95

Table 6. Comparison results of ResNet50 model with 3 FC layers on the original dataset with all the augmentations except blurring. Those results can be found in the appendix A.12 The augmentations are: i = Gaussian, iii = color jitter, iv = rotate, v = ISO.

5.3.2. Cropped ISL Dataset

The results of the second experiment on the cropped dataset are presented in Table 7, where ResNet50 was trained with a single FC layer of size 2048-21 with augmentations. Notably, all rows with up to two augmentations achieved an accuracy above 98.5%, with Entries (42)-(43) and (45), and Ent.(49) having the highest accuracies of 99.59%. This study will focus on Entries (45) and (49) as they use two augmentations instead of one, which adds complexity. Ent.(45) used Gaussian noise and rotation as augmentations, while Entry 49 used rotation and We hypothesize that these models performed the best due to their augmentation of adding noise, which aids the model in handling noise in the test images.

Additionally, the rotation augmentation is helpful as $\frac{2}{3}$ of the dataset has images with some rotation. It is noteworthy that the models with three augmentations had significantly lower results, which is due to the same reason mentioned above.

In the third experiment on the cropped dataset, Res50 was trained with 3 FC layers and augmentations, as shown in Table 8. The FC layers had sizes of 2048-1050, 1050-210, and 210-21. The results are comparable to those in Table 7, with all rows having up to two augmentations achieving an accuracy above 98.5%. Entry 64, which employed the augmentations of rotation and ISO, had the highest accuracy of 99.65%. We speculate that this model performed well for the same reasons as Entry 35 in Table 6. Additionally, note that using three augmentations resulted in lower accuracy, as discussed earlier.

An important observation from the experimental results presented in Tables 5, 6, 7, and 8 is that all models trained with three augmentations consistently perform worse compared to those with fewer augmentations. However, it is worth noting that more complex models and better datasets lead to significantly better results. For instance, in Table 5, all entries with three augmentations achieved approximately 22% accuracy, while in Table 8, the corresponding entries obtained an accuracy ranging between 75-80%. Therefore, it can be inferred that using a larger dataset with images centered around hands and small backgrounds could potentially lead to better accuracy for models trained with three augmentations, resulting in improved model robustness.

5.4. Multilingual Comparison - State-of-the-Art Outperformance on ASL & ISL

This section presents an experimental evaluation of the performance of our proposed models in comparison to the results reported in Abu-Jamie & Abu-Naser (2022). We denote our architecture by **MLSLD**, which stands for

ResNet50-Based Training on Data Augmentation and **1** FC, **cropped** dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
8	none	99.36	99.38	99.36	99.36
40	i	99.42	99.42	99.42	99.42
41	iii	99.48	99.48	99.48	99.48
42	iv	**99.59**	**99.60**	**99.59**	**99.59**
43	v	**99.59**	**99.60**	**99.59**	**99.59**
44	i iii	99.48	99.48	99.48	99.47
45	i iv	**99.59**	**99.60**	**99.59**	**99.59**
46	i v	99.42	99.43	99.42	99.42
47	iii iv	99.36	99.37	99.36	99.36
48	iii v	98.61	98.63	98.61	98.61
49	iv v	**99.59**	**99.60**	**99.59**	**99.59**
50	i iii iv	59.83	60.56	59.82	58.34
51	i iii v	57.82	59.31	57.82	56.23
52	i iv v	61.33	61.83	61.33	60.15
53	iii iv v	63.46	64.33	63.46	62.15

Table 7. Comparison results of ResNet50 with 1 FC layer on the cropped dataset with all the augmentations except blurring. Those results can be found in the appendix A.13 The augmentations are: i = Gaussian, iii = color jitter, iv = rotate, v = ISO.

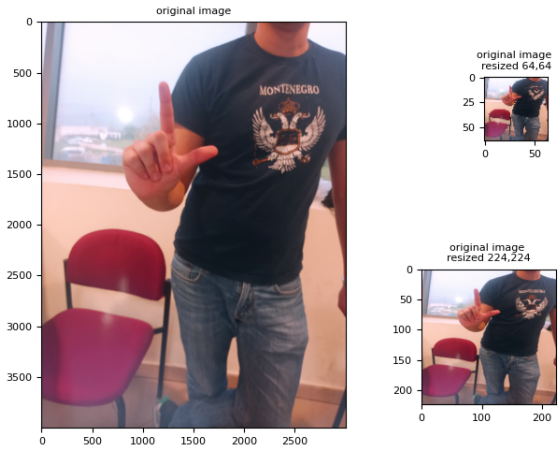
ResNet50-Based Training on Data Augmentation and **3** FC, **cropped** dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
54	none	99.48	99.49	99.48	99.48
55	i	99.30	99.32	99.30	99.30
56	iii	99.42	99.43	99.42	99.42
57	iv	99.36	99.38	99.36	99.36
58	v	99.07	99.10	99.07	99.08
59	i iii	99.36	99.37	99.36	99.36
60	i iv	99.19	99.22	99.19	91.19
61	i v	99.36	99.37	99.36	99.36
62	iii iv	98.67	98.74	98.67	98.68
63	iii v	98.96	98.98	98.96	98.96
64	iv v	**99.65**	**99.65**	**99.65**	**99.65**
65	i iii iv	80.20	80.66	80.20	79.83
66	i iii v	76.69	77.01	76.69	76.37
67	i iv v	80.03	80.64	80.03	79.78
68	iii iv v	82.16	82.46	82.16	81.87

Table 8. Comparison results of ResNet50 with 3 FC layers on the cropped dataset with all the augmentations except blurring. Those results can be found in the appendix A.14 The augmentations are: i = Gaussian, iii = color jitter, iv = rotate, v = ISO.

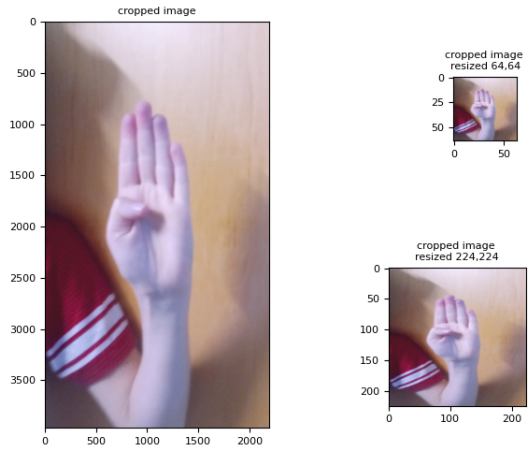
Multi**L**ingual **S**ign **L**anguage **D**etection, and present the comparison between the approaches as follows:

Table 9 summarizes the results of our best models (Entries (13), (35), (45), (49), (64)) and the ResNet50 architecture with ImageNet weights trained on the datasets used in this study (Entries (60) and (70)-(72)) by Abu-Jamie & Abu-Naser (2022). It is observed that the results reported by Abu-Jamie & Abu-Naser (2022) were considerably lower than the results obtained in our study for each dataset. Notably, their architecture employed RGB images of size 64x64 without additional FC layers, and no augmentation was utilized during the training phase.

Our experimental evaluation included a hypothesis that the low accuracy observed in Entries (69) and (71), in comparison to the best models in our study, could be attributed to the small image dimensions used in the architecture proposed by Abu-Jamie & Abu-Naser (2022). To test our hypothesis, we ran the Abu-Jamie & Abu-Naser (2022) architecture with images of size 224x224, similar to the architecture used in this study. Entries (70) and (72) show improved results, but they still fall short of the performance achieved by our models. Fig 5a depicts the differ-



(a) An original image of the letter Lamed (Fig 1 row 4 column 1) resized to 224x224 and 64x64, as the image gets smaller it gets more blurry and its harder to differentiate features



(b) A cropped image of the letter Bet (Fig 1 row 1 column 2) resized to 224x224 and 64x64, as the image gets smaller it gets more blurry and its harder to differentiate features

Figure 5. resized images

ence in image sizes for the original dataset, demonstrating that the 64x64 image lacks details, particularly the hand shape. Similarly, Fig 5b displays the difference in image sizes for the cropped dataset with a background-rich image, highlighting the challenge of seeing the hand's full detail in the 64x64 image.

The results in Table 9 confirm that our hypothesis regarding image size is valid. However, it does not explain why the models trained on larger images (Entries (70), (72)) still underperformed compared to our models. We hypothesize that our models' superior performance could be due to the use of augmentations during the training phase, which enhanced the models' robustness to variations in the data.

Comparison: Abu-Jamie & Abu-Naser (2022) and MLSLD - original and cropped datasets - ISL dataset						
Entry #	Dataset	Architectural Details	Accuracy	Precision	Recall	F1 Score
69	Original	Abu-Jamie & Abu-Naser (2022) 64x64	22.33	24.98	22.33	23.25
70	Original	Abu-Jamie & Abu-Naser (2022) 224x224	54.62	54.64	54.62	54.66
13	Original	MLSLD, Augmentations, original images, 1 FC	**94.06**	**94.55**	**94.06**	**94.01**
35	Original	MLSLD, Augmentations, original images, 3 FC	90.92	91.34	90.92	90.86
71	Cropped	Abu-Jamie & Abu-Naser (2022) 64x64	89.34	89.30	89.34	89.27
72	Cropped	Abu-Jamie & Abu-Naser (2022) 224x224	96.95	96.96	96.95	96.93
45, 49	Cropped	MLSLD, Augmentations, cropped images, 1 FC	99.59	99.60	99.59	99.59
64	Cropped	MLSLD, Augmentations, cropped images, 3 FC	**99.65**	**99.65**	**99.65**	**99.65**

Table 9. Comparison between MLSLD approach and the one from Abu-Jamie & Abu-Naser (2022) on ISL dataset.

Next, Table 10 is presented, and compares three sets of results:

- Our results of the best models for both datasets (Entries (13), (35), (45), (49), (64)).
- Our results of the best models after training on the ASL dataset that was used in Abu-Jamie & Abu-Naser (2022) (Entries (73)-(77)).
- The reported results from Abu-Jamie & Abu-Naser (2022) (Entry 78).

As can be seen, Ent.(78) has a higher accuracy compared to Entries (13) and (35), which can be attributed to the difference in dataset complexity, where the ASL dataset is less complex. Similarly, Ent.(64) has a slightly higher accuracy than Entries (45), (49), and (64), which is also due to the complexity of the data. However, Entries (45), (49), and (64) were able to compensate for this complexity by training with augmentations.

Furthermore, Entries (73)-(76), and Ent.(77) achieved the highest accuracy of 99.96%, and these models were trained on the ASL data. We hypothesize that these models performed better than the rest because they were trained on less complex data and used augmentations during training. This could also explain why these results were better than the results in Abu-Jamie & Abu-Naser (2022) (Ent.(78)), which used smaller images as inputs.

Comparison between Abu-Jamie & Abu-Naser (2022) and MLSLD - original and cropped datasets - ASL dataset					
Entry #	Architectural Details	Accuracy	Precision	Recall	F1 Score
73	ASL on MLSLD derived from Ent.(13)	**99.96**	99.96	99.96	99.96
74	ASL on MLSLD derived from Ent.(35)	**99.96**	99.96	99.96	99.96
13	Augmentations, original images, 1 FC	94.06	94.55	94.06	94.01
35	Augmentations, original images, 3 FC	90.92	91.34	90.92	90.86
75	ASL data on MLSLD derived from Ent.(45)	**99.96**	99.96	99.96	99.96
76	ASL data on MLSLD derived from Ent.(49)	**99.96**	99.96	99.96	99.96
77	ASL data on MLSLD derived from Ent.(64)	**99.96**	99.96	99.96	99.96
45, 49	Augmentations, cropped images, 1 FC	99.59	99.60	99.59	99.59
64	Augmentations, cropped images, 3 FC	99.65	99.65	99.65	99.65
78	Abu-Jamie & Abu-Naser (2022)	99.87			

Table 10. Comparison of the MLSLD approach and the one from Abu-Jamie & Abu-Naser (2022) on ASL dataset (cropped images). Note that Ent.(78) only contains the Accuracy metric, as presented by Abu-Jamie & Abu-Naser (2022). One can note that Entries (73)-(77) have achieved the highest accuracy since they represent a trained model on the ASL dataset, which is easier than the ISL. In addition, the architecture of these models is augmentations-oriented, which are beneficial to achieve higher metrics scores.

6. Conclusions and Future Work

In this work, we presented a deep learning approach for the classification of the Israeli Sign Language alphabet, using ResNet50 neural networks pre-trained on the ImageNet dataset. Our approach achieves high accuracies despite the challenges of a diverse real-life dataset containing images of both male and female hands, varying skin colors, and accessories. Furthermore, we demonstrate the effectiveness of the YOLO algorithm in cropping images around the hands to remove background noise and augmenting data to improve model robustness and accuracy.

Our approach achieved 94% accuracy on the original, more complex dataset and 99.65% accuracy on the cropped dataset with more images. We attribute this success to the use of augmentations while training the networks and using the YOLO algorithm to crop the original dataset. However, the paper also acknowledges that there are 28 letters in the Israeli Sign Language alphabet, and we only classified 21 letters in this work. Therefore, future work should classify all 28 letters using more than one image or video to capture dynamic signs.

In addition, the contributions of this work include the use of a real-life dataset that is diverse and challenging, the classification of a sign language that has not received much attention in the literature, the exploitation of augmentations to improve model generalization and the YOLO algorithm to augment the dataset, and demonstrating the robustness of our approach despite a relatively small dataset.

Our proposed approach can be extended to other sign languages with similar letter forms, and we believe that the Natural Language Processing approach will be helpful in the future to overcome mistakes in the classification of the letters by checking each word and making corrections using the BERT Tenney et al. (2019) algorithm. Furthermore, our work may be beneficial for individuals with hearing impairments who rely on sign language for communication.

In conclusion, our proposed approach demonstrates the effectiveness of deep learning in the classification of the Israeli Sign Language alphabet. We believe that this work has significant contributions to sign language recognition, and our approach can be extended to other sign languages. We hope our work inspires further research in this area, leading to improved communication and better access to information for individuals with hearing impairments.

7. Acknowledgments

This work was supported by the Ariel Cyber Innovation Center in conjunction with the Israel National Cyber directorate in the Prime Minister's Office, and Ariel Data Science and Artificial Intelligence Research Center.

In addition, the authors wish to thank Dr. Roi Yozevitch, for sharing and inspiring a subset of his outstanding learning techniques, presented in his books.

As aforementioned, the ISL dataset has been gathered by undergraduate students in the School of Computer Science at Ariel University. Clearly, the authors wish to thank to anyone who accepted to donate for collecting the ISL dataset.

References

Abu-Jamie, T. N., & Abu-Naser, S. S. (2022). Classification of sign-language using deep learning by resnet, .

Ahirwal, B., Khadtare, M., & Mehta, R. (2007). Fpga based system for color space transformation rgb to yiq and ycbcr. In *2007 International Conference on Intelligent and Advanced Systems* (pp. 1345–1349). IEEE.

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1–6). Ieee.

Alzamzami, F., Hoda, M., & El Saddik, A. (2020). Light gradient boosting machine for general sentiment classification on short texts: a comparative evaluation. *IEEE access*, *8*, 101840–101858.

Anidjar, O. H., Barak, A., Ben-Moshe, B., Hagai, E., & Tuvyahu, S. (2023). A stethoscope for drones: Transformers-based methods for uavs acoustic anomaly detection. *IEEE Access*, *11*, 33336–33353. doi:10.1109/ACCESS.2023.3262702.

Aryanie, D., & Heryadi, Y. (2015). American sign language-based finger-spelling recognition using k-nearest neighbors classifier. In *2015 3rd International Conference on Information and Communication Technology (ICoICT)* (pp. 533–536). IEEE.

Biau, G., & Scornet, E. (2016). A random forest guided tour. *Test*, *25*, 197–227.

Bock, S., & Weiß, M. (2019). A proof of local convergence for the adam optimizer. In *2019 international joint conference on neural networks (IJCNN)* (pp. 1–8). IEEE.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Chelghoum, R., Ikhlef, A., Hameurlaine, A., & Jacquir, S. (2020). Transfer learning using convolutional neural network architectures for brain tumor classification from mri images. In *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part I 16* (pp. 189–200). Springer.

Chuan, C.-H., Regina, E., & Guardino, C. (2014). American sign language recognition using leap motion sensor. In *2014 13th International Conference on Machine Learning and Applications* (pp. 541–544). IEEE.

Clark, C., Gjestland, T., Lavia, L., Notley, H., Michaud, D., & Morinaga, M. (2021). Assessing community noise annoyance: A review of two decades of the international technical specification iso/ts 15666: 2003. *The Journal of the Acoustical Society of America*, *150*, 3362–3373.

Cohen, G., Afshar, S., Tapson, J., & Van Schaik, A. (2017). Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)* (pp. 2921–2926). IEEE.

De Brebisson, A., & Vincent, P. (2015). An exploration of softmax alternatives belonging to the spherical loss family. *arXiv preprint arXiv:1511.05042*, .

Desjardins, J. L., & Doherty, K. A. (2014). The effect of hearing aid noise reduction on listening effort in hearing-impaired adults. *Ear and hearing*, *35*, 600–610.

Deuchar, M. (2013). *British sign language*. Routledge.

Dornadula, A., Pokle, A., & Yerukola, A. (). Defendr: A robust model for image classification, .

Dutta, A., Gupta, A., & Zissermann, A. (2016). Vgg image annotator (via). *URL: http://www. robots. ox. ac. uk/˜ vgg/software/via*, .

Ekbote, J., & Joshi, M. (2017). Indian sign language recognition using ann and svm classifiers. In *2017 International conference on innovations in information, embedded and communication systems (ICIIECS)* (pp. 1–5). IEEE.

Elsayed, N., ElSayed, Z., & Maida, A. S. (2022). Vision-based american sign language classification approach via deep learning. *arXiv preprint arXiv:2204.04235*, .

Ghotkar, A. S., Khatal, R., Khupase, S., Asati, S., & Hadap, M. (2012). Hand gesture recognition for indian sign language. In *2012 International Conference on Computer Communication and Informatics* (pp. 1–4). IEEE.

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). Knn model-based approach in classification. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings* (pp. 986–996). Springer.

Guzsvinecz, T., Szucs, V., & Sik-Lanyi, C. (2019). Suitability of the kinect sensor and leap motion controller—a literature review. *Sensors*, *19*, 1072.

Holdengreber, E., Yozevitch, R., & Khavkin, V. (2021). Intuitive cognition-based method for generating speech using hand gestures. *Sensors*, *21*, 5291.

Hore, S., Chatterjee, S., Santhi, V., Dey, N., Ashour, A. S., Balas, V. E., & Shi, F. (2017). Indian sign language recognition using optimized neural networks. In *Information Technology and Intelligent Transportation Systems: Volume 2, Proceedings of the 2015 International Conference on Information Technology and Intelligent Transportation Systems ITITS 2015, held December 12-13, 2015, Xi'an China* (pp. 553–563). Springer.

Jakkula, V. (2006). Tutorial on support vector machine (svm). *School of EECS, Washington State University*, *37*, 3.

Jang, E., Gu, S., & Poole, B. (2016). Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, .

Janzen, T., & Shaffer, B. (2002). Gesture as the substrate in the process of asl grammaticization. *Modality and structure in signed and spoken languages*, (pp. 199–223).

Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A review of yolo algorithm developments. *Procedia Computer Science*, *199*, 1066–1073.

Johnston, T. (2003). Bsl, auslan and nzsl: three signed languages or one? In *Theoretical Issues in Sign Language Research Conference (7th: 2000)* (pp. 47–69). Signum.

Johnston, T., & Schembri, A. (2007). *Australian Sign Language (Auslan): An introduction to sign language linguistics*. Cambridge University Press.

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*, 255–260.

Karpf, A. (2011). *The human voice: The story of a remarkable talent*. Bloomsbury Publishing.

Khorov, E., Kureev, A., Levitsky, I., & Akyildiz, I. F. (2022). A phase noise resistant constellation rotation method and its experimental validation for noma wi-fi. *IEEE Journal on Selected Areas in Communications*, *40*, 1346–1354.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, .

Kosmidou, V. E., Petrantonakis, P. C., & Hadjileontiadis, L. J. (2011). Enhanced sign language recognition using weighted intrinsic-mode entropy and signer's level of deafness. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *41*, 1531–1543.

Kumar, M., Gupta, P., Jha, R. K., Bhatia, A., Jha, K., & Shah, B. K. (2021). Sign language alphabet recognition using convolution neural network. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1859–1865). IEEE.

Kyle, J. G., Kyle, J., Woll, B., Pullen, G., & Maddix, F. (1988). *Sign language: The study of deaf people and their language*. Cambridge university press.

Laroche, C., Almansa, A., & Tassano, M. (2023). Deep model-based super-resolution with non-uniform blur. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 1797–1808).

Lindeberg, T. (2012). Scale invariant feature transform, .

Lipton, Z. C., Elkan, C., & Narayanaswamy, B. (2014). Thresholding classifiers to maximize f1 score. *arXiv preprint arXiv:1402.1892*, .

Liwicki, S., & Everingham, M. (2009). Automatic recognition of fingerspelled words in british sign language. In *2009 IEEE computer society conference on computer vision and pattern recognition workshops* (pp. 50–57). IEEE.

Loizou, P. C. (1999). Introduction to cochlear implants. *IEEE Engineering in Medicine and Biology Magazine*, *18*, 32–42.

Ma, Y., Xu, T., & Kim, K. (2022). Two-stream mixed convolutional neural network for american sign language recognition. *Sensors*, *22*, 5959.

Mandelbrot, B. B., & Wallis, J. R. (1969). Computer experiments with fractional gaussian noises: Part 1, averages and variances. *Water resources research*, *5*, 228–241.

Mannan, A., Abbasi, A., Javed, A. R., Ahsan, A., Gadekallu, T. R., & Xin, Q. (2022). Hypertuned deep convolutional neural network for sign language recognition. *Computational Intelligence and Neuroscience*, *2022*.

Mavi, A. (2020). A new dataset and proposed convolutional neural network architecture for classification of american sign language digits. *arXiv preprint arXiv:2011.08927*, .

Meir, I. (1999). A perfect marker in israeli sign language. *Sign Language & Linguistics*, *2*, 43–62.

Meir, I., & Sandler, W. (2007). *A language in space: The story of Israeli Sign Language*. Psychology Press.

Milton, M. A. A. (2018). Evaluation of momentum diverse input iterative fast gradient sign method (m-di2-fgsm) based attack method on mcs 2018 adversarial attacks on black box face recognition system. *arXiv preprint arXiv:1806.08970*, .

Mocialov, B., Turner, G., & Hastie, H. (2020). Transfer learning for british sign language modelling. *arXiv preprint arXiv:2006.02144*, .

Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, *24*, 1565–1567.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, .

Padden, C. A., & Gunsauls, D. C. (2003). How the alphabet came to be used in a sign language. *Sign Language Studies*, (pp. 10–33).

Prateek, S., Jagadeesh, J., Siddarth, R., Smitha, Y., Hiremath, P. S., & Pendari, N. T. (2018). Dynamic tool for american sign language finger spelling interpreter. In *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)* (pp. 596–600). IEEE.

Pugeault, N., & Bowden, R. (2011). Spelling it out: Real-time asl fingerspelling recognition. In *2011 IEEE International conference on computer vision workshops (ICCV workshops)* (pp. 1114–1119). IEEE.

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do imagenet classifiers generalize to imagenet? In *International conference on machine learning* (pp. 5389–5400). PMLR.

Rezende, E., Ruppert, G., Carvalho, T., Ramos, F., & De Geus, P. (2017). Malicious software classification using transfer learning of resnet-50 deep neural network. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 1011–1014). IEEE.

Rong, W., Li, Z., Zhang, W., & Sun, L. (2014). An improved canny edge detection algorithm. In *2014 IEEE international conference on mechatronics and automation* (pp. 577–582). IEEE.

Salvin, A., Routh, D. K., Foster, R. E., & Lovejoy, K. M. (1977). Acquisition of modified american sign language by a mute autistic child. *Journal of autism and childhood schizophrenia*, *7*, 359–371.

Shanthi, T., & Sabeenian, R. (2019). Modified alexnet architecture for classification of diabetic retinopathy images. *Computers & Electrical Engineering*, *76*, 56–64.

Shen, M., Han, K., Xu, C., & Wang, Y. (2019). Searching for accurate binary neural architectures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (pp. 0–0).

Shin, J., Matsuoka, A., Hasan, M. A. M., & Srizon, A. Y. (2021). American sign language alphabet recognition by extracting feature from hand pose estimation. *Sensors*, *21*, 5856.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, .

Stokoe Jr, W. C. (2005). Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of deaf studies and deaf education*, *10*, 3–37.

Taha, A. A., & Malebary, S. J. (2020). An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE Access*, *8*, 25579–25587.

Tai, Y., Yang, J., & Liu, X. (2017). Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3147–3155).

Tammina, S. (2019). Transfer learning using vgg-16 with deep convolutional neural network for classifying images. *International Journal of Scientific and Research Publications (IJSRP)*, *9*, 143–150.

Tannenbaum-Baruchi, C., Feder-Bubis, P., Adini, B., & Aharonson-Daniel, L. (2014). Emergency situations and deaf people in israel: communication obstacles and recommendations. *Disaster health*, *2*, 106–111.

Targ, S., Almeida, D., & Lyman, K. (2016). Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029*, .
Tenney, I., Das, D., & Pavlick, E. (2019). Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, .
Tomasi, C. (2012). Histograms of oriented gradients. *Computer Vision Sampler*, (pp. 1–6).
Treiman, R., Levin, I., & Kessler, B. (2007). Learning of letter names follows similar principles across languages: Evidence from hebrew. *Journal of Experimental Child Psychology*, *96*, 87–106.
Treu, M., Le, T.-N., Nguyen, H. H., Yamagishi, J., & Echizen, I. (2021). Fashion-guided adversarial attack on person segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 943–952).
Wang, L. (2005). *Support vector machines: theory and applications* volume 177. Springer Science & Business Media.
Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big data*, *3*, 1–40.
Wu, J. (2017). Introduction to convolutional neural networks. *National Key Lab for Novel Software Technology. Nanjing University. China*, *5*, 495.
Wu, Z., Shen, C., & Van Den Hengel, A. (2019). Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, *90*, 119–133.
Xu, Z., Guan, H., Kang, J., Lei, X., Ma, L., Yu, Y., Chen, Y., & Li, J. (2022). Pavement crack detection from ccd images with a locally enhanced transformer network. *International Journal of Applied Earth Observation and Geoinformation*, *110*, 102825.
Yozevitch, R., Ben-Moshe, B., & Dvir, A. (2014). Gnss accuracy improvement using rapid shadow transitions. *IEEE Transactions on Intelligent Transportation Systems*, *15*, 1113–1122.
Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, .
Zhang, H.-w., Hu, Y., Zou, Y.-j., & Wu, C.-y. (2021). Fingerspelling identification for american sign language based on resnet-18. *International Journal of Advanced Networking and Applications*, *13*, 4816–4820.
Zhang, M.-L., & Zhou, Z.-H. (2007). Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, *40*, 2038–2048.
Zhang, S., Li, X., Zong, M., Zhu, X., & Cheng, D. (2017). Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *8*, 1–19.
Zhou, Z.-H. (2021). *Machine learning*. Springer Nature.

Appendix A. Tables

The following Appendix section corresponds with the rows that were excluded from Tables 5, 6, 7 and 8 in Section 5.3. The Entry count was restarted from 1, and the tables are corresponding in the same order as the ones in Section 5.3.

Augmemtations containing blurring, ResNet50 with 1 FC layers, original dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
1	ii	86.56	88.32	86.56	86.68
2	i ii	87.60	89.13	87.60	87.63
3	ii iii	87.60	88.79	87.60	87.76
4	ii iv	**90.75**	**91.42**	**90.75**	**90.75**
5	ii v	84.29	86.69	84.29	84.45
6	i ii iii	19.72	22.04	19.72	18.72
7	i ii iv	20.06	21.46	20.06	18.64
8	i ii v	20.59	19.96	20.59	18.97
9	ii iii iv	19.72	19.46	19.72	18.23
10	ii iii v	19.54	19.66	19.54	18.30
11	ii iv v	20.06	21.43	20.06	19.18

Table A.11. Comparison results of ResNet50 with 1 FC layer on the original dataset with augmentations that contain blurring. The augmentations are: i = Gaussian, ii = blur, iii = color jitter, iv = rotate, v = ISO.

Augmemtations containing blurring, ResNet50 with 3 FC layers, original dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
12	ii	87.78	88.63	87.78	87.89
13	i ii	88.48	89.04	88.48	88.41
14	ii iii	87.78	88.21	87.78	87.71
15	ii iv	**89.87**	**90.42**	**89.87**	**89.89**
16	ii v	87.43	88.11	87.43	87.49
17	i ii iii	36.47	37.49	36.47	35.60
18	i ii iv	41.71	44.18	41.71	40.47
19	i ii v	36.12	36.61	36.12	34.65
20	ii iii iv	38.56	40.23	38.56	37.83
21	ii iii v	33.15	35.59	33.15	31.65
22	ii iv v	37.69	38.55	37.69	36.54

Table A.12. Comparison results of ResNet50 with 3 FC layers on the original dataset with augmentations that contain blurring. The augmentations are: i = Gaussian, ii = blur, iii = color jitter, iv = rotate, v = ISO.

Augmemtations containing blurring, ResNet50 with 1 FC layers, cropped dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
23	ii	**99.53**	**99.54**	**99.53**	**99.53**
24	i ii	99.36	99.37	99.36	99.36
25	ii iii	99.19	99.22	99.19	99.19
26	ii iv	99.36	99.37	99.36	99.36
27	ii v	99.07	99.10	99.07	99.07
28	i ii iii	57.24	58.22	57.24	56.04
29	i ii iv	57.13	58.26	57.13	55.68
30	i ii v	58.22	58.82	58.82	57.04
31	ii iii iv	59.03	60.21	59.03	57.76
32	ii iii v	56.55	57.89	56.55	55.14
33	ii iv v	57.71	58.84	57.71	56.44

Table A.13. Comparison results of ResNet50 with 1 FC layer on the cropped dataset with augmentations that contain blurring. The augmentations are: i = Gaussian, ii = blur, iii = color jitter, iv = rotate, v = ISO.

Augmemtations containing blurring, ResNet50 with 3 FC layers, cropped dataset					
Entry #	Augmentations	Accuracy	Precision	Recall	F1 Score
34	ii	99.25	99.25	99.25	99.25
35	i ii	**99.53**	**99.54**	**99.53**	**99.53**
36	ii iii	98.79	98.86	98.79	98.79
37	ii iv	98.96	98.98	98.96	98.96
38	ii v	99.36	99.37	99.36	99.36
39	i ii iii	77.10	77.46	77.10	96.99
40	i ii iv	79.40	80.28	79.40	79.31
41	i ii v	75.94	76.73	75.94	75.59
42	ii iii iv	77.10	78.17	77.10	76.88
43	ii iii v	77.15	77.65	77.15	76.87
44	ii iv v	78.48	79.17	78.48	78.26

Table A.14. Results of ResNet50 with 3 FC layers on the cropped dataset with augmentations that contain blurring. The augmentations are: i = Gaussian, ii = blur, iii = color jitter, iv = rotate, v = ISO.