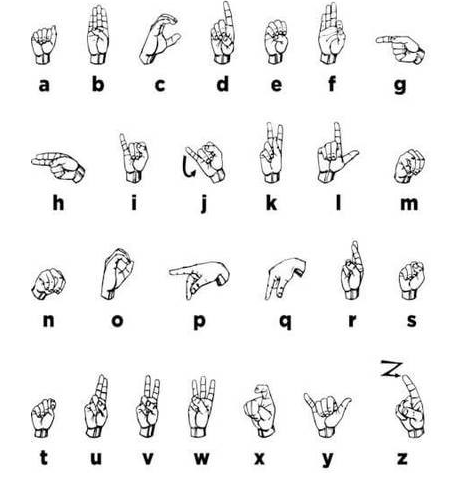
**Introduction:**

Pakistan Sign Language (PSL) is a complete, natural language that is expressed using the movement of hands and face. PSL provides the deaf community a way to interact within the community itself as well as to the outside world. However, not everyone knows about signs and gestures used in the sign language. With the advent of [Artificial Neural Networks](https://medium.com/technology-invention-and-more/everything-you-need-to-know-about-artificial-neural-networks-57fac18245a1) and [Deep Learning](https://www.mathworks.com/discovery/deep-learning.html), it is now possible to build a system that can recognize objects or even objects of various categories (like red vs green apple). Utilizing this, here we have an application that uses a deep learning model trained on the ASL Dataset to predict the sign from the sign language given an input image or frame from a video feed.

To build PSLD (Pakistan sign language detection) we must need **3** things:

* **Dataset**
* **Model (In our project we train model using CNN)**
* **Python (For input image from user)**

**Alphabet signs in Pakistan Sign Language are shown below:**



**Dataset**

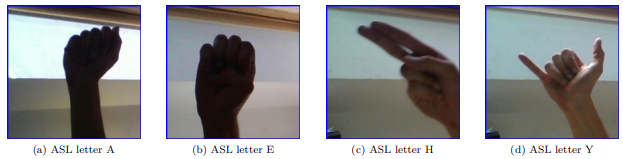
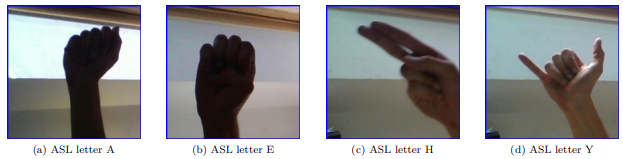
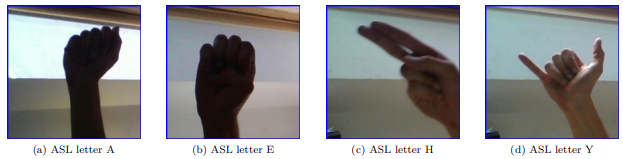
To develop the first prototype of this system was used a dataset of 24 static signs from Alphabet.



**Sources of Data:**

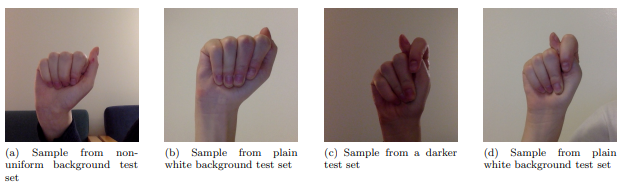
**Data Collection**: The primary source of data for this project was the compiled dataset of Pakistan Sign Language (PSL). The dataset is comprised of images which are 200x200 pixels. There are 29 classes, 26 for the letters A-Z and 3 for space, delete and nothing.

A collage of hands

Description automatically generated with low confidence   

These photos were then cropped, rescaled, and labelled for use. Examples of images from the Kaggle dataset used for training. Note difficulty of distinguishing fingers in the letter E. A self-generated test set was created in order to investigate the neural network’s ability to generalize. Five different test sets of images were taken with a webcam under different lighting conditions, backgrounds, and use of dominant/non-dominant hand. These images were then cropped and preprocessed.

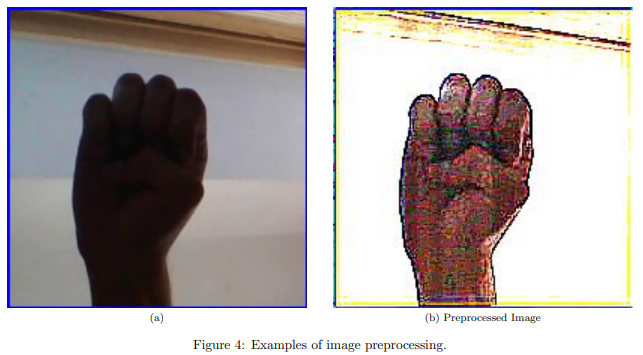
**Data Pre-processing**



**Image Enhancement**: A combination of brightness, contrast, sharpness, and color enhancement was used on the images. For example, the contrast and brightness were changed such that fingers could be distinguished when the image was very dark.

**Edge Enhancement:** Edge enhancement is an image-filtering techniques that makes edges more defined. This is achieved by the increase of contrast in a local region of the image that is detected as an edge. This has the effect of making the border of the hand and fingers, versus the background, much more-clear and distinct. This can potentially help the neural network identify the hand and its boundaries.

**Image Whitening**: ZCA, or image whitening, is a technique that uses the singular value decomposition of a matrix. This algorithm decorrelates the data, and removes the redundant, or obvious, information out of the data. This allows for the neural network to look for more complex and sophisticated relationships, and to uncover the underlying structure of the patterns it is being trained on. The covariance matrix of the image is set to identity, and the mean to zero.



**Learning/Modeling**

We used a Convolutional Neural Network, or CNN, model to classify the static images in our first dataset. Our first goal when building the neural network was to define our input layer. A 28x28 image contains 784 pixels each represented by a grayscale value ranging from 0 (black) to 1 (white). By converting each image to a series of numbers, we transform the data into a format the computer can read.

Once the input layer has been prepared, it can be processed by the neural network’s hidden layers. The architecture of our neural network can be seen below.

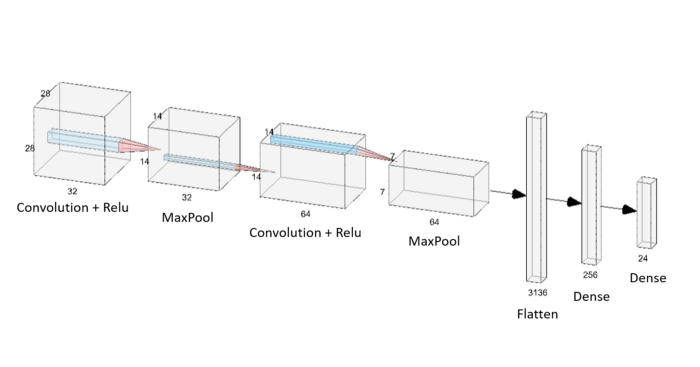


Figure: The architecture of the Convolutional Neural Network, [Public Domain](https://www.kaggle.com/datamunge/sign-language-mnist)