CAPSTONE PROPOSAL

Machine Learning Project

On

SIGN LANGUAGE GESTURE RECOGNITION FROM VIDEO FRAMES

OCTOBER 29, 2020

UDACITY Nanodegree Program

Authored by: Pankhuri Bhatnagar

SIGN LANGUAGE GESTURE RECOGNITION FROM VIDEO FRAMES

Domain Background:

Inability to speak is considered to be true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language.

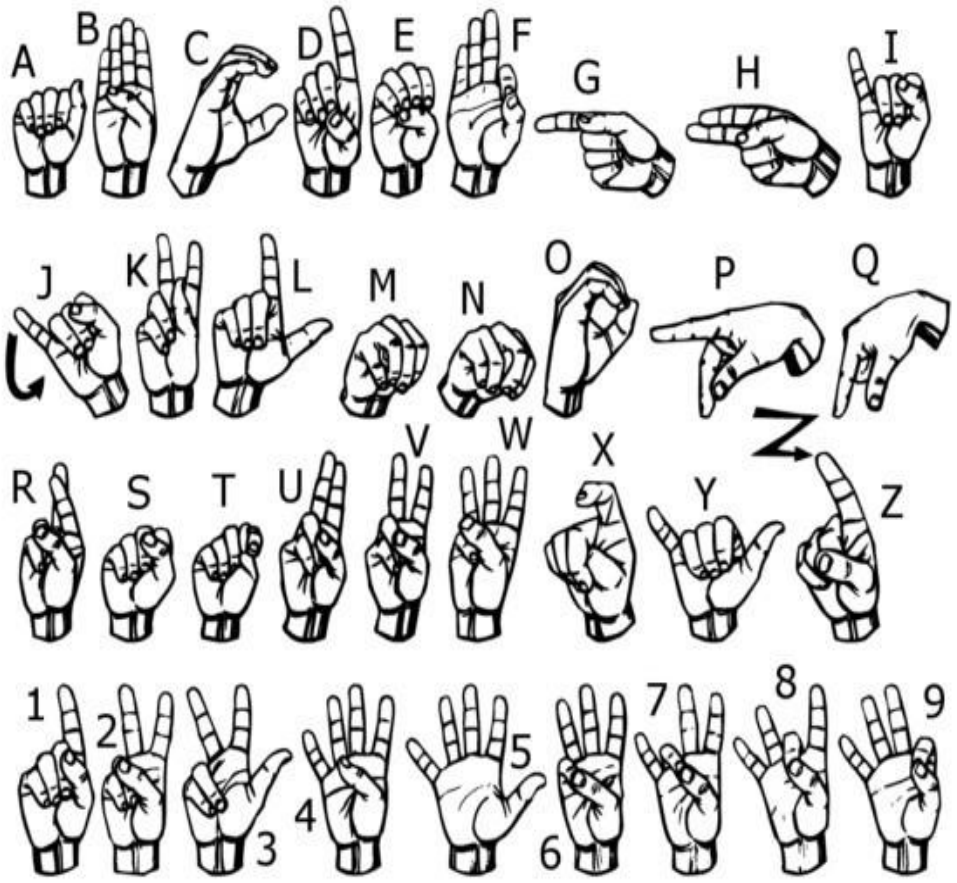
Developing sign language application for deaf people can be very important, as they'll be able to communicate easily with even those who don't understand sign language. Our project aims at taking the basic step in bridging the communication gap between normal people, deaf and dumb people using sign language.

Problem Statement:

The main focus is to create a vision-based system to identify sign language gestures from the video sequences. The reason for choosing a system based on vision relates to the fact that it provides a simpler and more intuitive way of communication between a human and a computer.

Video sequences contain both the temporal as well as the spatial features. So, we can use two different models to train both the

temporal as well as the spatial features. To train the model on the spatial features of the video sequences we can use Inception model which is a deep CNN (convolutional neural net).



Datasets and Input:

LSA64: A Dataset for Argentinian Sign Language:

The sign database for the **Argentinian Sign Language**, created with the goal of producing a dictionary for LSA and training an automatic sign recognizer, includes ***3200*** *videos where 10 non-expert subjects executed 5 repetitions of* ***64*** *different types of signs*. Signs were selected among the most commonly used ones in the LSA lexicon, including both verbs and nouns.

Signs

The following table lists the id, name and hands (H column) used in each sign. The column H specifies whether the sign was performed with the right hand (R) or both hands (B).

ID	Name	H	ID	Name	H	ID	Name	H	ID	Name	H
01	Opaque	R	17	Call	R	33	Hungry	R	49	Yogurt	B
02	Red	R	18	Skimmer	R	34	Map	B	50	Accept	B
03	Green	R	19	Bitter	R	35	Coin	B	51	Thanks	B
04	Yellow	R	20	Sweet milk	R	36	Music	B	52	Shut down	R
05	Bright	R	21	Milk	R	37	Ship	R	53	Appear	B
06	Light-blue	R	22	Water	R	38	None	R	54	To land	B
07	Colors	R	23	Food	R	39	Name	R	55	Catch	B
08	Red	R	24	Argentina	R	40	Patience	R	56	Help	B
09	Women	R	25	Uruguay	R	41	Perfume	R	57	Dance	B
10	Enemy	R	26	Country	R	42	Deaf	R	58	Bathe	B
11	Son	R	27	Last name	R	43	Trap	B	59	Buy	R
12	Man	R	28	Where	R	44	Rice	B	60	Copy	B
13	Away	R	29	Mock	B	45	Barbecue	B	61	Run	B
14	Drawer	R	30	Birthday	R	46	Candy	R	62	Realize	R
15	Born	R	31	Breakfast	B	47	Chewing-gum	R	63	Give	B
16	Learn	R	32	Photo	B	48	Spaghetti	B	64	Find	R

Solution Statement:

To train the model on the spatial features of the video sequences we can use Inception model which is a deep CNN. CNN can be trained on the frames obtained from the video sequences of train data. We can then use RNN (recurrent neural network) to train the model on the temporal features. Trained CNN model will be used to make predictions for individual frames to obtain a sequence of predictions or pool layer outputs for each video.

Now this sequence of prediction or pool layer outputs will be given to RNN to train on the temporal features.

Benchmark Model:

Many approaches have been implemented over the past few years. From a lot of binary and multi classification algorithms to extensive use of Computer Vision, a tremendous attention has been given in this area to help humans break the physical barriers and express themselves like everyone can.
A benchmark deep learning model using YOLO algorithm with 80% accuracy will be compared to that of the model using CNN and RNN Neural Networks.

A detailed Approach for our model:

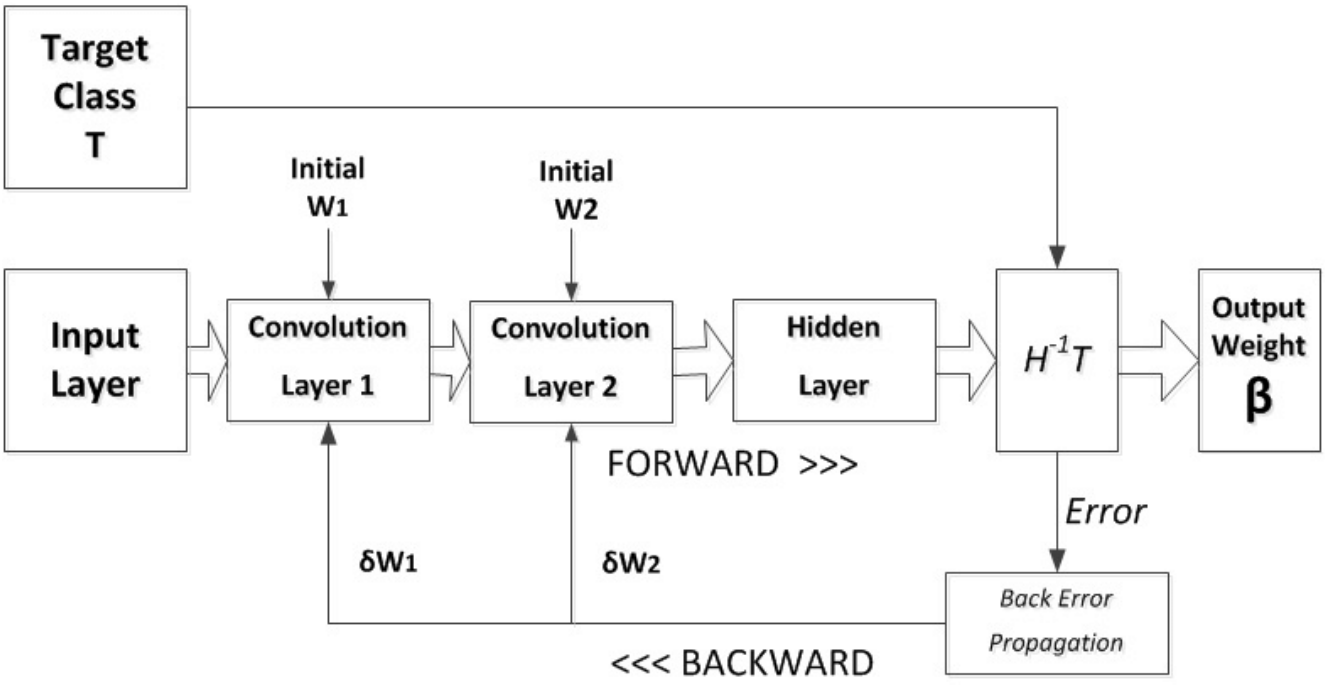
1. Firstly, we should use CNN for feature extraction from images using Google's Inception V3 model. InceptionV3 is a CNN that is trained on various images.
2. InceptionV3 will produce the Bottleneck files which are one of the last stage files that are to be given for training. By converting Images to their suitable Bottlenecks, we reduce the no. of features to train on without any loss.
3. Then, we create a CNN model with the output layer having 29 classification units for Gesture Classification.
4. For testing, OpenCV can be used and the live feed from the camera will be converted to images that can further be classified by our model using tensorflow and the general TFGraph File.

Evaluation Metrics:

The project evaluation metric for this project is the **accuracy.**
Based on the accuracy we will be able to determine the level proficiency for the model performance on classifying the pictures.
The accuracy will be determined as follows:
Accuracy = (The number of correctly classified gestures / The total number of examples) x 100

Project Design:

There are four main steps in CNN: convolution, subsampling, activation and full connectedness.



Summarized Steps discussed in this proposal: