Final Report of Internship Program 2020

On

"Classifying ASL letters using Machine Learning"

MEDTOUREASY, NEW DELHI

27th January
2021

ACKNOWLDEGMENTS

The internship opportunity that I had with MedTourEasy was a great change for learning and understanding the intricacies of the Machine Learning; and also, for personal as well as professional development. I am very obliged for having a chance to interact with so many professionals who guided me throughout the internship project and made it a great learning curve for me.

Firstly, I express my deepest gratitude and special thanks to the Training Head of MedTourEasy, Mr. Ankit Hasija who gave me an opportunity to carry out my internship at their esteemed organization. Also, I express my thanks to him for making me understand the details of the Machine Learning profile and training me in the same so that I can carry out the project properly and with maximum client satisfaction and also for spearing his valuable time in spite of his busy schedule.

I would also like to thank the team of MedTourEasy and my colleagues who made the working environment productive and very conducive.

TABLE OF CONTENTS

ABSTRACT

American Sign Language (ASL) is a complete, natural language that has the same linguistic properties as spoken languages, with grammar that differs from English. ASL is expressed by movements of the hands and face. It is the primary language of many North Americans who are deaf and hard of hearing, and is used by many hearing people as well..

ASL is a language completely separate and distinct from English. It contains all the fundamental features of language, with its own rules for pronunciation, word formation, and word order. While every language has ways of signaling different functions, such as asASL is a language completely separate and distinct from English. It contains all the fundamental features of language, with its own rules for pronunciation, word formation, and word order. While every language has ways of signaling different functions, such as asking a question rather than making a statement, languages differ in how this is done. For example, English speakers may ask a question by raising the pitch of their voices and by adjusting word order; ASL users ask a question by raising their eyebrows, widening their eyes, and tilting their bodies forward.king a question rather than making a statement, languages differ in how this is done. For example, English speakers may ask a question by raising the pitch of their voices and by adjusting word order; ASL users ask a question by raising their eyebrows, widening their eyes, and tilting their bodies forward.
This Project will classify the ASL letters based on the images from the dataset and by using advanced machine learning algorithms to predict the values of newer images.

1.1 About the Company

MedTourEasy, a global healthcare company, provides you the informational resources needed to evaluate your global options. It helps you find the right healthcare solution based on specific health needs, affordable care while meeting the quality standards that you expect to have in healthcare. MedTourEasy improves access to healthcare for people everywhere. It is an easy to use platform and service that helps patients to get medical second opinions and to schedule affordable, high-quality medical treatment abroad.

1.2 About the Project

This project is developed to create a system that can predict the classify the ASL letters based on the images of input dataset . the project helps in translation of images to a layman who dosent understand the ASL letters. The steps taken in the project:

1. Understanding ASL
2. Visualizing the data
3. Data preprocessing
4. Creating the model
5. Training the model
6. Testing the model
7. Visualizing the model

1.3 Objectives and Deliverables

This project focuses on creating a proper classification model that can classify the ASL letters based on the images . It uses the concepts of neural networks which is a part of machine learning . It outputs the accuracy of the model and also the wrong predictions.

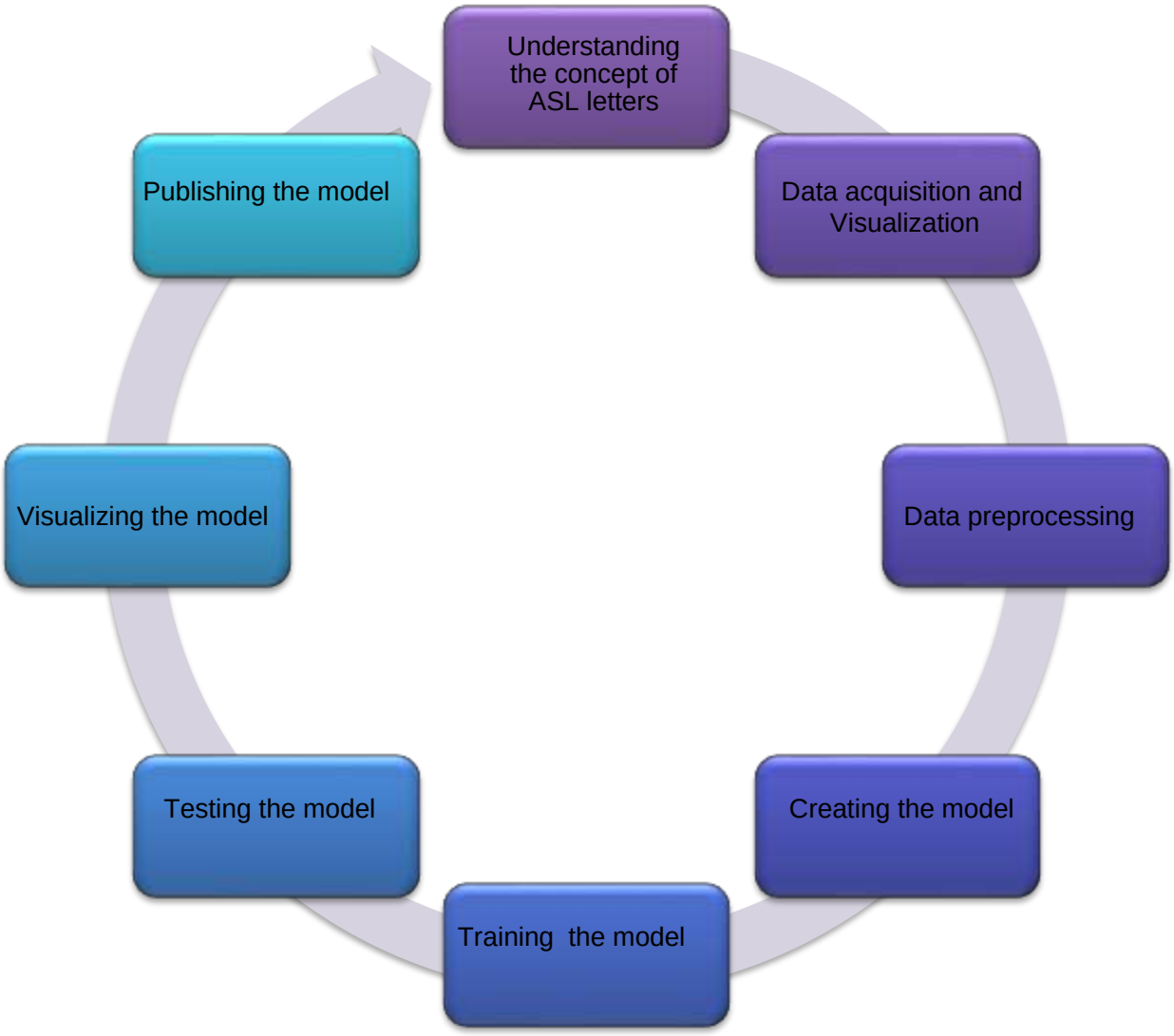
The project consists of 2 deliverables detailed as follows:

a. **Visualizing the dataset:** this output shows the images present in the input dataset
b. **Visualizing the model:** this output shows the wrong predictions of the model and their respective correct predictions.
c. It also shows the accuracy of the model on training and testing data.

I. METHODOLOGY

2.1 Flow of the Project

The project followed the following steps to accomplish the desired objectives and deliverables. Each step has been explained in detail in the following section.



2.2 Language and Platform Used

2.2.1 Language: Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

2.3.2 IDE: Jupter Notebook (Anaconda IDE)

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

2.3.3 Packages: Keras , Tenserflow,Pandas

- Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides.
- TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.
- pandas is a fast, powerful, flexibl and easy to use opensource data analysis and manipulation tool, built on top of the Python programming language.

```
# Import packages and set numpy random seed
import numpy as np
np.random.seed(5)
import tensorflow as tf
tf.random.set_seed(2)
from datasets import sign_language
import matplotlib.pyplot as plt
%matplotlib inline

# Load pre-shuffled training and test datasets
(x_train, y_train), (x_test, y_test) = sign_language.load_data()
```

II. IMPLEMENTATION

3.1 Understanding the ASL letters and dataset

This is the first step wherein the information about ASL is understood and the variations of the letters in the dataset are visualized and the letters are segregated.

3.2 Data Collection and Importing

The data is present in form of images that are uploaded using the predefined functions of laod_data present in separate .py file. The various required libraries and packages are also inported in order to be able to use the functionalities of some functions present in them in generating the model..

3.3 b

Once the data has been imported into the jupyter notebook, it is important to process the data . As the data present is in the form of images we need to extract the information from the images and convert it into numerical data on which the model will be built .

3.4 Model creation

Selecting the model is the most important task when it comes to creating a new system as it will define how well your system works on test cases and also shows the efficieny it can provide. Chosing the wrong model will result in lesser accuracy and non proper implementation of the dataset .

3.5 Training the model:

The Model is fitted on the training data and the accuracy is put as the metrics to measure the goodness of the model.

3.6 Testing the model :

The model is then validated on the testing dataset (the 20% of the whole dataset was separated on random) to find the accuracy of the model .

3.7 Visualizing the model:

The wrongly predicted data is visualized so as to see the wrong predictions and the models ability to predict.

III. SAMPLE SCREENSHOTS AND OBSERVATIONS

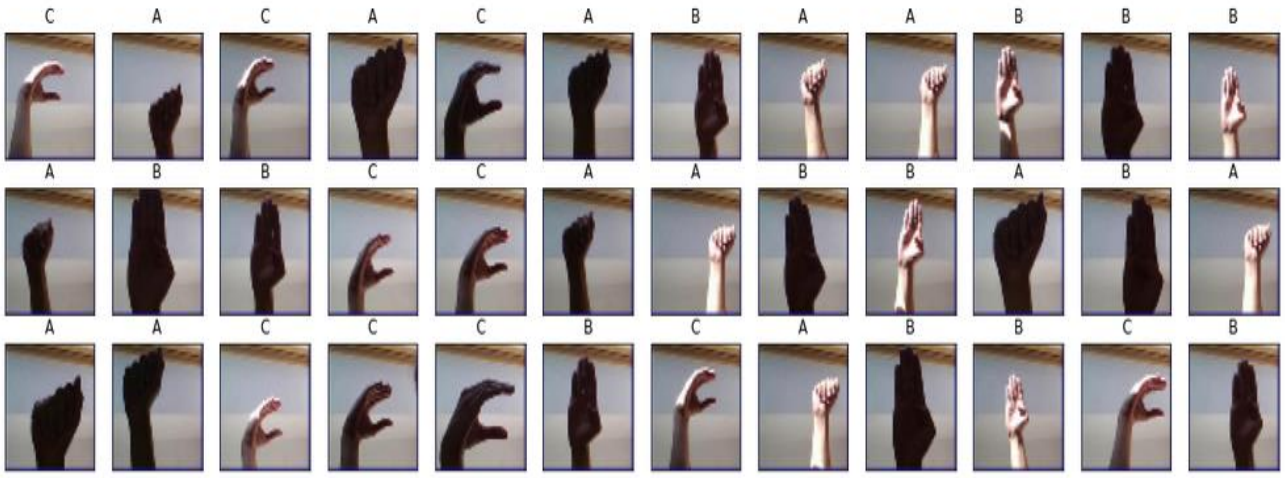
4.1 The code screenshots:

```
# Import packages and set numpy random seed
import numpy as np
np.random.seed(5)
import tensorflow as tf
tf.random.set_seed(2)
from datasets import sign_language
import matplotlib.pyplot as plt
%matplotlib inline

# Load pre-shuffled training and test datasets
(x_train, y_train), (x_test, y_test) = sign_language.load_data()
```

2. Visualize the training data

Now we'll begin by creating a list of string-valued labels containing the letters that appear in the dataset. Then, we visualize the first several images in the training data, along with their corresponding labels.

```
# Store labels of dataset
labels = ['A','B','C']

# Print the first several training images, along with the labels
fig = plt.figure(figsize=(20,5))
for i in range(36):
    ax = fig.add_subplot(3, 12, i + 1, xticks=[], yticks=[])
    ax.imshow(np.squeeze(x_train[i]))
    ax.set_title("{}".format(labels[y_train[i]]))
plt.show()
```



```
# Number of A's in the training dataset
num_A_train = sum(y_train==0)
# Number of B's in the training dataset
num_B_train = sum(y_train==1)
# Number of C's in the training dataset
num_C_train = sum(y_train==2)

# Number of A's in the test dataset
num_A_test = sum(y_test==0)
# Number of B's in the test dataset
num_B_test = sum(y_test==1)
# Number of C's in the test dataset
num_C_test = sum(y_test==2)

# Print statistics about the dataset
print("Training set:")
print("\tA: {}, B: {}, C: {}".format(num_A_train, num_B_train, num_C_train))
print("Test set:")
print("\tA: {}, B: {}, C: {}".format(num_A_test, num_B_test, num_C_test))
```

```
Training set:
        A: 540, B: 528, C: 532
Test set:
        A: 118, B: 144, C: 138
```

- `0` is encoded as `[1, 0, 0]`,
- `1` is encoded as `[0, 1, 0]`, and
- `2` is encoded as `[0, 0, 1]`.

```
from keras.utils import np_utils
import keras
# One-hot encode the training labels
y_train_OH = np_utils.to_categorical(y_train)

# One-hot encode the test labels
y_test_OH = np_utils.to_categorical(y_test)
```

```
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Flatten, Dense
from keras.models import Sequential

model = Sequential()
# First convolutional layer accepts image input
model.add(Conv2D(filters=5, kernel_size=5, padding='same', activation='relu',
                 input_shape=(50, 50, 3)))
# Add a max pooling layer
model.add(MaxPooling2D())
# Add a convolutional layer
model.add(Conv2D(filters=15,kernel_size=5 , padding='same',activation='relu'))
# Add another max pooling layer
model.add(MaxPooling2D())
# Flatten and feed to output layer
model.add(Flatten())
model.add(Dense(3, activation='softmax'))

# Summarize the model
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 50, 50, 5)         380
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 25, 25, 5)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 25, 25, 15)        1890
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 15)        0
_________________________________________________________________
flatten (Flatten)            (None, 2160)              0
_________________________________________________________________
dense (Dense)                (None, 3)                 6483
=================================================================
Total params: 8,753
Trainable params: 8,753
Non-trainable params: 0
_________________________________________________________________
```

6. Compile the model

After we have defined a neural network in Keras, the next step is to compile it!

```
# Compile the model
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics='accuracy')
```

7. Train the model

Once we have compiled the model, we're ready to fit it to the training data.

```
# Train the model
hist = model.fit(x_train,y_train_OH,epochs=2,batch_size=32)
```

```
Epoch 1/2
50/50 [==============================] - 1s 24ms/step - loss: 0.1464 - accuracy: 0.9569
Epoch 2/2
50/50 [==============================] - 1s 24ms/step - loss: 0.0655 - accuracy: 0.9831
```

8. Test the model

To evaluate the model, we'll use the test dataset. This will tell us how the network performs when classifying images it has never seen before!

If the classification accuracy on the test dataset is similar to the training dataset, this is a good sign that the model did not overfit to the training data.

```
# Obtain accuracy on test set
score = model.evaluate(x=x_test,
                       y=y_test_OH,
                       verbose=0)

print('Test accuracy:', score[1])
```

```
Test accuracy: 0.9900000095367432
```

```
# Get predicted probabilities for test dataset
y_probs = model.predict(x_test)

# Get predicted labels for test dataset
y_preds = np.argmax(y_probs,axis=-1)


# Indices corresponding to test images which were mislabeled
bad_test_idxs  = [i for i,v in enumerate(y_preds) if y_preds[i]!=y_test[i]]


# Print mislabeled examples
fig = plt.figure(figsize=(25,4))
for i, idx in enumerate(bad_test_idxs):
    ax = fig.add_subplot(2, np.ceil(len(bad_test_idxs)/2), i + 1, xticks=[], yticks=[])
    ax.imshow(np.squeeze(x_test[idx]))
    ax.set_title("{} (pred: {})".format(labels[y_test[idx]], labels[y_preds[idx]]))
```

IV. CONCLUSION AND FUTURE SCOPE

Future Scope:

This project can be combined with live video to provide a real life translation and also NLP can be included to convert the audio into proper translation tools for the specially challenged . also this can provide a great tool for learning for special needs.

Conclusion:

The sign languages are very difficult for the layman to understand it would take lot of effort for a person with special need to be able to convey a message this tool/project when combined with its future scope would greatly improve the efficiency and also make lives easier.