Introduction:

The goal of this project is to train a model to classify the sign presented in each image. There are 29 different signs in the dataset. Twenty-six of them represent the American Sign Language (A to Z), and three signs of "space, delete, and nothing". In this project I was responsible of creating a model by combining different types of layers runned with a variety of parameters, I also created and combined the accuracy and the test graphs.

My work:

I started with creating a model using Keras(in Colab platform), then I modified this model by adding different layers including: convolutional, max-pooling, and Dropout layers. After several trials I was able to get a very good model. In this model I included 12 convolutional layers followed by 2 fully connected layers. I normalized all the convolutional layers, after every four conv layers I included a max-pooling layer except the last four layers. In the development of this model I tried to follow the conventions reported by well known AI researchers like Andrew NJ. In the original model I created in Keras, I did not include drop out layers for some convolutional layers, because when I was developing the model I noticed that including several(2-4) conv layers with the same number of input channels and different kernel, stride, and padding sizes yields a better accuracy results than the drop out layers. In the Pytorch model I could not implement the same model I created in Keras(which reached an accuracy of 98.3% in val_set after 40 epochs) because I was confused with the Pytorch framework. Here is a part of the model I created in Keras which performed very well:

```
my_model = Sequential()
my_model.add(Conv2D(32, kernel_size=3, strides=1, activation='relu', padding='same', kernel_constraint=max_norm(3), input_shape=target_dims))
my_model.add(BatchNormalization())
my_model.add(Dropout(0.2))
my_model.add(Conv2D(32, kernel_size=3, strides=1, activation='relu', padding='same'))
my_model.add(BatchNormalization())
my_model.add(Conv2D(32, kernel_size=4, strides=2, activation='relu'))
my_model.add(BatchNormalization())
my_model.add(Conv2D(32, kernel_size=5, strides=1, activation='relu', kernel_constraint=max_norm(3)))
my_model.add(BatchNormalization())
my_model.add(Dropout(0.2))
my_model.add(MaxPooling2D(pool_size = (2, 2),  padding='same'))
```

Summary and conclusion:

Before I started working in this project I took the deep learning courses available for free in Coursera and Youtube, but when I started implementing the model I encountered different problems and I had to search thoroughly on the internet in order to solve them. Some of what is recommended in the blog posts are not reliable like this post which advised not to use the the dropout layers except in the fully connected layers and replace them with Batch normalization which is the opposite what the renowned researchers advised people to do. I tried replacing the dropout with the batch normalisation in the convolutional layers but it didn't workout as suggested by the blogger.

Given the nature(simple) of this problem and the data(clear images) associated with it, I believe that this problem can be solved with a few number of layers 8 to 12 layers, because it does not need to recognize very complicated structures. For future improvement I would suggest to experiment with different architectures while using a small number of layers.

I copied around 70% of the code from the internet and from the professor's code provide for the exam.

References:

https://towardsdatascience.com/dont-use-dropout-in-convolutional-networks-81486c823c16

https://www.kaggle.com/grassknoted/asl-alphabet

https://www.youtube.com/channel/UCcIXc5mJsHVYTZR1maL5l9w

https://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf