In this project, you are asked to build the mutation effect prediction models. This project can be more challenging than the previous ones.

You are given two datasets (beta-lactamase and IF1). For each dataset, preprocess.py takes in the multiple sequence alignment (training data) and the deep mutational scan data (testing data), and returns the following:

* From the multiple sequence alignment (.a2m), (1) train\_data.npy: multiple sequence alignment (MSA) sequences in one-hot encoding. The MSA was obtained by running a homology search tool against a target protein sequence.
* From the deep mutational scan data (.csv), (2) test\_data.npy: mutated sequences having single mutations with respect to the target protein sequence and (3) target\_values.npy: mutation effect scores for the mutated test sequences.

You need to build two deep autoregressive generative models, | | ,…, ). You can use dilated causal convolution layers, tf.keras.layers.Conv1D, <https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv1D> for both proteins. The only difference between the two models is the optimal kernel size and the number of dilated causal convolution layers. This is because the MSA of each protein family has a different sequence length.

For a kernel size=*k* and number of dilated causal convolution layers=*n*, a model with exponentially increasing dilation rate (1, 2, 4, 8, ...) will have a receptive field of 1+(*k*-1)x(2*n*-1). For more information, <https://medium.com/the-artificial-impostor/notes-understanding-tensorflow-part-3-7f6633fcc7c7>. The receptive field has to be greater than or equal to the sequence length. We recommend using (*k*, *n*) = (5, 6) for beta-lactamase and (*k*, *n*) = (6, 4) for IF1. Other hyperparameters can be the same.

When training the model, you should use negative log likelihood (softmax cross entropy) between the input and the output.

When testing the model, you should compute the spearman rank correlation (“from scipy.stats import spearmanr”) between the target values and the model-predicted mutation effects (you can use the keras callback for evaluating the spearman correlation during training, <https://github.com/keras-team/keras/issues/2548>). The model-predicted mutation effects are computed using the log likelihood (logp) of the mutant sequence, which is equivalent to the negative of the loss.

Make sure to pad the sequence at the beginning of the model, and truncate the sequence at the end of the model. This is to make sure that the output at time step L is only influenced by the inputs at time steps 1, ..., (L-1). This can be done using the keras Lambda function. Refer to: <https://www.tensorflow.org/api_docs/python/tf/keras/backend/temporal_padding>

Also make sure to use residual connections in the model. For example, “h2 = tf.keras.layers.Add()([h2, h1])”