**Model architecture and implementation details**

* 1. Generate a vocabulary from given data using TextVectorization method of Keras.
  2. Download Pre-Trained GloVe embeddings and generate a mapping from each word of the vocabulary to its GloVe embedding.
  3. Model architecture and implementation:
     1. Each sentence is converted into an array of word embeddings (1 embedding for each word) of size (25483, 300) where 25483 is the length of the longest sentence in the dataset (all smaller sentences are padded to increase length) and 300 is the length of the GloVe embedding for each word. From now, we will call this array sentence embedding.
     2. Each of the sentence embedding in the pair is fed into identical Siamese subnetworks.
     3. The identical Siamese subnetworks consist of a CNN layer with 50 filters of size 9 each and ReLu activation, followed by 2 BiLSTM layers with 200 neurons each.
     4. The CNN layer is used to gather local context of the word by taking into consideration the information from surrounding words.
     5. This local context is concatenated to the corresponding word embedding in the sentence embedding and then fed to the first BiLSTM layer.
     6. The output from the final timestep of the final BiLSTM layer is treated as an encoding for the sentence.
     7. We calculate the L1 distance between the encoding generated by the twin Siamese networks (between encoding of the 2 sentences in the pair) and feed it into a dense layer.
     8. The output from this final dense layer is our desired similarity score.
  4. Due to limited time and computing resources, I could not train the model for a larger number of epochs. The model has potential to perform even better if trained for a larger number of epochs.
  5. Advantages (reason for implementing):
     1. Siamese networks are really good at capturing the similarity/difference between inputs by generating encodings for each input and comparing them.
     2. The CNN layer can capture context of each word by leveraging data from surrounding words and therefore may result in richer encodings for the word.
     3. The BiLSTM, again, gathers some context from surrounding words and may generate a richer sentence embedding.
     4. Can scale to any length of input sentence without losing out on data/semantic information.
  6. Disadvantages:
     1. As there are relatively large number of parameters, it takes a relatively huge amount of time to train (approximately 2hr 40min/epoch).