# Deep Learning - Homework 7:

Due Monday April 20, 2020

Use your favorable deep learning software to implement 1) an LSTM network and 2) a GRU network for the IMDB Movie reviews sentiment classification dataset listed below. Train your networks and present your best results in a table that includes the minibatch size, the number of hidden layers, the numbers of hidden units, the total number of parameters **excluding** the embedding layer (see below), the optimization methods used, the learning rate, the initializer, the regularization if used, the number of training epochs, the final training accuracy, the final testing accuracy, and the ratio of the test accuracy to the 10th root of the total number of parameters. Your test accuracy should be at least 85%. The ratio will be used to score your results. Also, include the plots of the convergence curve of the loss function, the training accuracy and the testing accuracy against the training epochs.

IMDB dataset (see `http://ai.stanford.edu/~amaas/data/sentiment/`):
The dataset contains texts of movie reviews and a label of positive or negative for each review. The goal is to train a neural network to classify a text review as being positive or negative - hence a binary classification problem. The dataset consists of 25,000 reviews for training and another 25,000 for testing.

1. In Keras, download the dataset using these options with `vocab_size=8000` and `maxLen =250`
   ```
   (x_train, y_train), (x_test, y_test) =
   tf.keras.datasets.imdb.load_data(path="imdb.npz", num_words=vocab_size,
   skip_top=0, maxlen=maxLen, seed=113, start_char=1, oov_char=2, index_from=3)
   ```
2. Pad the sequences to the same length of `maxLen=250` using
   ```
   x_train = tf.keras.preprocessing.sequence.pad_sequences(x_train, maxlen=maxLen)
   x_test = tf.keras.preprocessing.sequence.pad_sequences(x_test, maxlen=maxLen)
   ```
3. Use word embedding to represent words in a dense vector representation; see `https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/` for an explanation. In Keras, this is implemented by using an embedding layer as the first layer for inputs:
   ```
   tf.keras.layers.Embedding(vocab_size, 128, input_length=maxLen)
   ```
   Basically, this converts words labeled by integers to one-hot vectors of dimension of vocabulary size (`vocab_size`) and then maps them to 128-dimensional vectors with a trainable weight matrix. Do not count the parameters of this embedding layer in your total parameter counts. This layer should be the same for everyone.
4. A Tensorflow LSTM implementation is available at `https://www.tensorflow.org/tutorials/text/text_classification_rnn` but the dataset is generated with slightly different options from above. Be sure to use the options described above.

**Notes:**

- Submit your codes and results through canvas in a single PDF file.
- Grading: 9 points out of 10 will be given for achieving a testing accuracy of at least 85% and the scoring ratio $r$ of at least 0.25 for both problems. 10 points will be given to the highest ratio of the class.

**Project for graduate students only:**

Implement a CNN (1D convolutional layers) network for this problem. Submit a report that should include a brief introduction to the problem, a description of each of the three network architectures you used, and discussions on experiments leading to your architectures. Also discuss your conclusions with respect to advantages and disadvantages of the three architectures.