Hamza Oudrhiri

K1910115

**C17520: Machine Learning and Artificial Intelligence**

**Coursework 2: Neural Network / Artificial Intelligence System**

**Introduction**

The theme we were assigned was text classification. The dataset we had to study was the Reuters dataset from the Keras library. It is a dataset that contains 11228 samples that are classified over 46 classes. It is important to note that each sample is only assigned one class, therefore it is not a multi-label classification problem but a multi-class classification problem.

The classes are the following (1):

{'copper': 6, 'livestock': 28, 'gold': 25, 'money-fx': 19, 'ipi': 30, 'trade': 11, 'cocoa': 0, 'iron-steel': 31, 'reserves': 12, 'tin': 26, 'zinc': 37, 'jobs': 34, 'ship': 13, 'cotton': 14, 'alum': 23, 'strategic-metal': 27, 'lead': 45, 'housing': 7, 'meal-feed': 22, 'gnp': 21, 'sugar': 10, 'rubber': 32, 'dlr': 40, 'veg-oil': 2, 'interest': 20, 'crude': 16, 'coffee': 9, 'wheat': 5, 'carcass': 15, 'lei': 35, 'gas': 41, 'nat-gas': 17, 'oilseed': 24, 'orange': 38, 'heat': 33, 'wpi': 43, 'silver': 42, 'cpi': 18, 'earn': 3, 'bop': 36, 'money-supply': 8, 'hog': 44, 'acq': 4, 'pet-chem': 39, 'grain': 1, 'retail': 29}

We were asked to find 2 pre-existing model, then tweak them to improve their performance and in the last part we had to design our own deep learning model.

**PART I:**

The dataset we had to use was part of the Keras library, it was loaded using a preset function. The training and testing tests were divided into 80% and 20% respectively.

As for the parameters. we chose to select only the 10000 most used words in the newswire.

We then vectorized (2) the samples into tensors of shape (length, dimensions) where the number of dimensions was the number of words we chose to select (10000). This basically gave us 11228 tensors that were arrays of 10000 dimensions that contained 1 wherever a word was inside the newswire in the appropriate index, and 0s for the rest.

Similarly, we encoded the labels to be arrays of 46 dimensions that contained 0s everywhere except a 1 for the appropriate class number.

The samples were readable by using a dictionary of integer and word, which we used to convert the array of integers into text. Please note, since we only use the 10000 most common words the decoded newswire might not make perfect sense.

**PART II:**

The **first model** we found for part II was found in the following website:

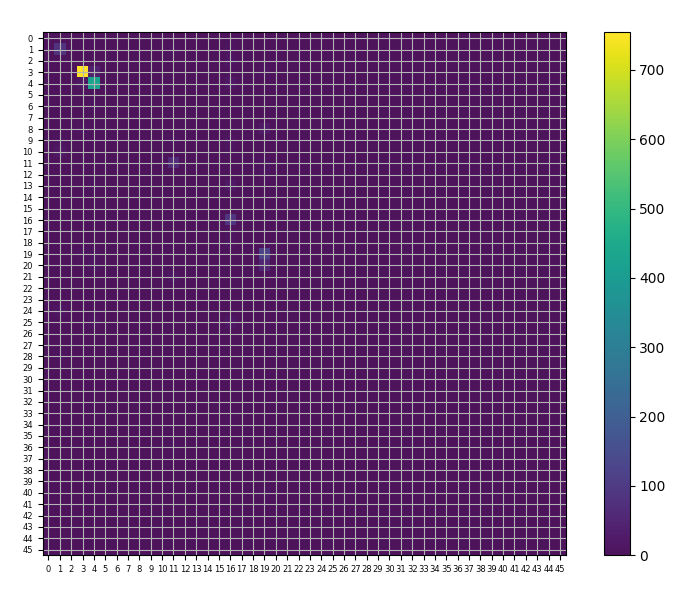
<https://www.kaggle.com/drscarlat/reuters-document-classification-with-keras-tf>

The network is a multi-layer network that contains 2 fully connected layers and has 10 epochs with a batch-size of 512.

The first layer takes as input an array of shape (8982, 10000) and outputs an array of (8982, 256). Dropout of 0.5 is added to both layers, in order to avoid overfitting

The first model had an accuracy of 0.69 and an f1score of 0.67

Below is the confusion matrix:



The second model we used was found here:

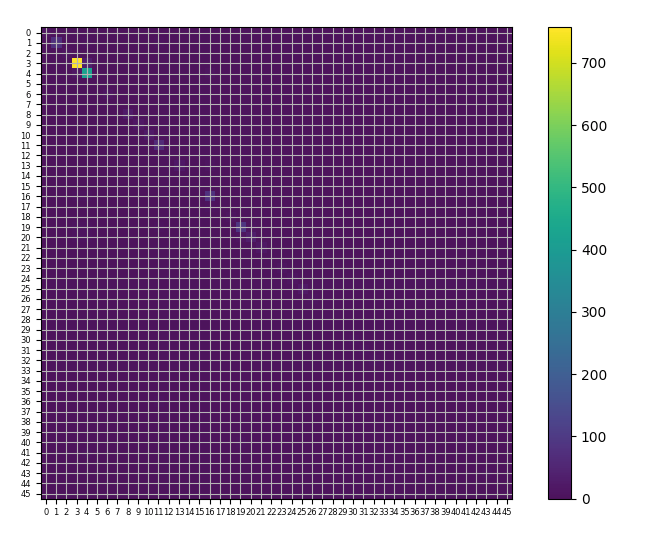
<https://towardsdatascience.com/text-classification-in-keras-part-1-a-simple-reuters-news-classifier-9558d34d01d3>

It is a single-layer neural network that contains a Dense layer that outputs an array of (8982,512). No dropout was added.

It was running over 3 epochs with a batch size of 32.

The second model had an accuracy of 0.80 and an f1score of 0.80

Below is the confusion matrix for the 2nd model:



**Part IV:**

For this part, we had to create our own deep learning neural network. We created a recurrent neural network, which is ideal for text processing. A recurrent neural network processes sequences by iterating through the sequence elements and maintaining a state containing information relative to what it has seen so far (3).

Our model had an accuracy of 0.36 and an f1score of 0.36

As we can see, the 2nd model had the best performance. It predicted samples in 44 different classes whereas the 1st model only predicted samples in 6 classes.

**References:**

1. <https://github.com/keras-team/keras/issues/12072>
2. F.Chollet, *Deep learning with Python (2018),* p.78
3. F.Chollet, *Deep learning with Python (2018),* p.196