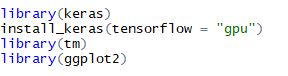
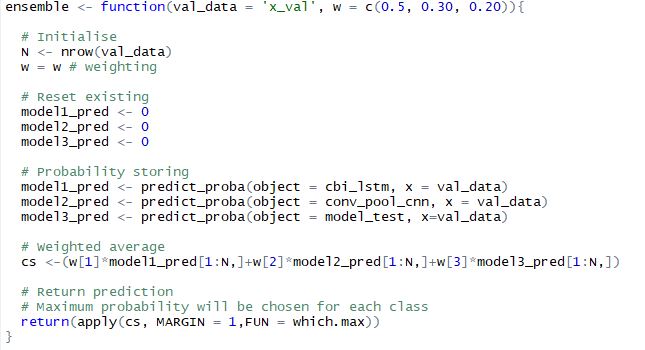
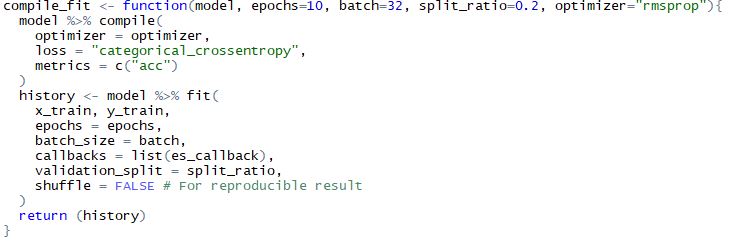
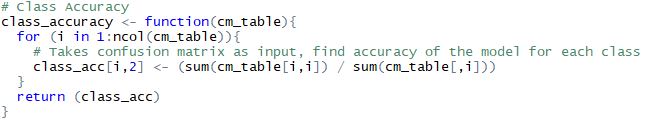
1. Introduction

We were given data containing 106,445 documents that would be used as the training

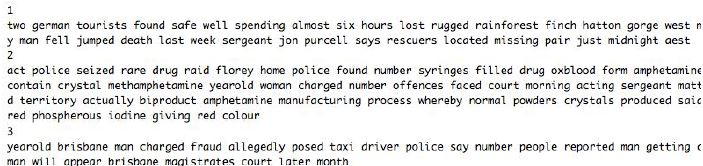
data for our models. The training labels for each document range from Class 1 to Class 23. My application contains 3 models:

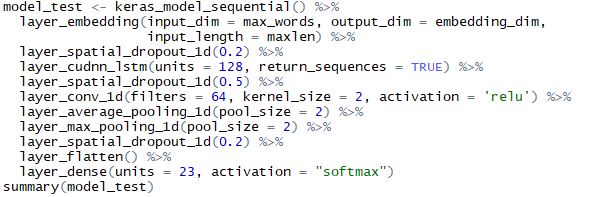
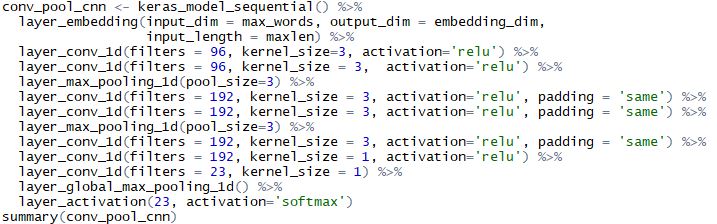
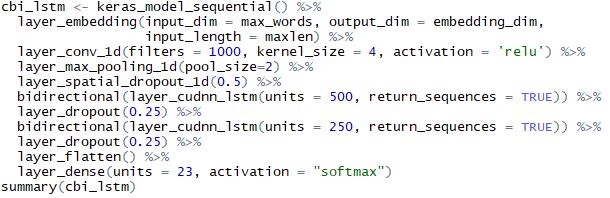
* Convolutional Bi-directional LSTM
* ConvPool-CNN-C
* LSTM Conv Net
* And finalized by an ensemble model

1. Installation

* Required libraries:
  + 
* Required functions:
  + This function aims to calculate macro f1 score as a validation score on testing model performance
  + This function is ensemble model, which combine other models into single one then predict based on validation data.
  + Compiler function is where we define loss function and optimizer for our models.
  + Class accuracy aims to calculate a table to report how our model will perform on each class.
* Data pre-processing:
  + Remove numbers
  + Remove punctuations
  + Remove stop words
  + Strip redundant whitespace

First, we created a corpus from all the document texts. This corpus saves the text from all the news articles and means that we are able to begin pre-processing our articles using the TM package. The first things we did was (in order) remove numbers, lowercase, remove punctuation, remove stopwords, and then strip whitespace. There were other steps that we had tried in the pre-processing such as stemming and TF-IDF but they actually decreased the accuracy of our model, hence they were removed from the pre-processing. Our process leaves us with a text dataframe that looks like below.



* Create models:
  + LSTM Convolutional Neural Network (CNN) 
  + ConvPool CNN-C using stride 1 of standard convolution layers instead of using pooling layers 
  + Convolutional Bi-Directional Long Short-Term Memory (LSTM )

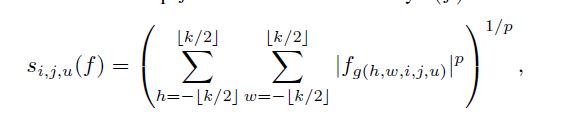
1. Model explanation

* Convolutional Bi-Directional Long Short-Term Memory (LSTM)

Convolutional Bi-Directional LSTM model consists of 2 mains stages. A convolutional stage and bi-directional LSTM stage. The basic idea of this approach is that once the embedding layer reads a word vector, the convolutional layer creates convolved features using 1000 filters and ‘relu’ activation function. The Max pooling layer then receives feature vectors from the convolutional layer, extracts important features from taking the max values from each activation map. Once we have extracted the important features, two layers of Bidirectional LSTM is used to learn the ordering of the features. Bi-directional implies that information flows both forward and backwards simultaneously. Also dropout is used to prevent overfitting problem in neural network (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014).

* Conv Pool CNN-C

ConvPool CNN-C using stride 1 of standard convolution layers instead of using pooling layers. Based on reference (Springenberg, Dosovitskiy, Brox, & Riedmiller, 2014) increase size stride of convolution layer then it can perform as pooling layer with this mapping function:

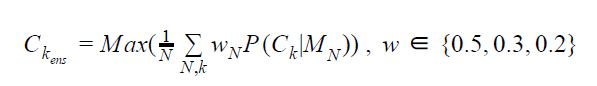


3-D array input size WxHxN will be sampled with size of pooling k and stride r will map feature from s to f (both s and f are positions in input vector space) with mapping function g and p-norm order p can be infinite. In our experiment we allow the network to overlap by using r = 1 and will be used as feature extraction with p-norm as activation function.

* LSTM Convolutional Neural Network (CNN)

Basic CNNs work by feeding multidimensional data to a convolutional layer, this layer is comprised of multiple filters that are used to learn different features regarding whatever input you have given it (Sosa, 2017). The reason we use a CNN with text is that when given text such as our news articles, this is very structured and organised. The CNN can discover and learn patterns that would otherwise be lost in a feed-forward network (Sosa, 2017). For example, a CNN can learn and recognise the difference between the phrases ‘under the weather’ and ‘the weather is good’ even though there is a use of a synonym in the two phrases, this of course makes it excellent for sentiment and textual analysis. The LSTM-CNN model works by first creating a word embedding for each word in our input (in this case, the news articles) (Lai, Xu, Liu & Zhao 2015). Here, the model picks up the input as well as memorises contextual information.

* Ensemble approach using all 3 models



We had run three different models separately and received a wide range of training accuracies which can be seen in Appendix 6.1. Our overall classification model was a weighted average ensembling model which applies the following weights to each models’ probabilistic prediction (Opitz, D, & Maclin, R. 1999): 0.5 to Model 1, 0.3 to Model 2 and 0.2 to Model 3. These weights were chosen by trial and error given model 1 had the best performance in general, and represented the combination that provided the highest level of accuracy. The weights work by looking at the probability of the class predicted for each model and then selecting the highest result after multiplying by the weights. The ensembling process for us worked as it took the ‘best’ features from each model and provided us with a higher accuracy than had we used any of the three models on their own. The use of all three models is warranted as each model has a different method of processing and offer different results to each other.