Classification of texts by neural networks.

In my final project, I used neural networks to determine the topics of news from the AG's News Topic Classification Dataset. AG is a collection of more than 1 million news articles. News articles have been gathered from more than 2000 news sources by ComeToMyHead in more than 1 year of activity. ComeToMyHead is an academic news search engine which has been running since July, 2004. The dataset is provided by the academic comunity for research purposes in data mining (clustering, classification, etc), information retrieval (ranking, search, etc), xml, data compression, data streaming, and any other non - commercial activity.

I built 3 NN models: 1D Convolutional neutral network, LSTM and GRU neural networks. And compared their performance on test date set.

First, we start with getting training data and text tokenization. Text tokenization is the process of breaking down a given text into smaller units called tokens. These tokens can be individual words, characters, or subwords, depending on the specific tokenization technique used.

Getting train data

```
train_set = pd.read_csv('train.csv',
                        header=None,
                        names=['class', 'title', 'text'])
```

Choosing data for training

```
news = train_set['text']
```

Getting correct answers

```
y_train = utils.to_categorical(train_set['class'] - 1, nb_classes)
```

Text tokenization

```
tokenizer = Tokenizer(num_words=num_words)
tokenizer.fit_on_texts(news)
```

After we convert news to a numeric representation.

Converting news to a numeric representation

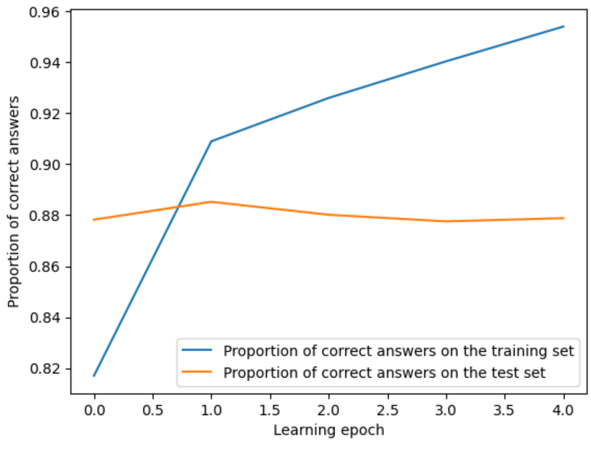
```
sequences = tokenizer.texts_to_sequences(news)

x_train = pad_sequences(sequences, maxlen=max_news_len)
```

From now we can build our NN models. Here is what they look like.

Convolutional Neural Network

```
model_cnn = Sequential([
    Embedding(num_words, 32, input_length=max_news_len),
    Conv1D(250, 5, padding='valid', activation='relu'),
    GlobalMaxPooling1D(),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(32, activation='relu'),
```

```
    Dropout(0.2),
    Dense(4, activation='softmax')])

model_cnn.summary()

Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_10 (Embedding)    (None, 30, 32)            320000

 conv1d_3 (Conv1D)           (None, 26, 250)           40250

 global_max_pooling1d_3 (Glo  (None, 250)              0
 balMaxPooling1D)

 dense_16 (Dense)            (None, 64)                16064

 dropout_1 (Dropout)         (None, 64)                0

 dense_17 (Dense)            (None, 32)                2080

 dropout_2 (Dropout)         (None, 32)                0

 dense_18 (Dense)            (None, 4)                 132

=================================================================
Total params: 378,526
Trainable params: 378,526
Non-trainable params: 0
_________________________________________________________________
```



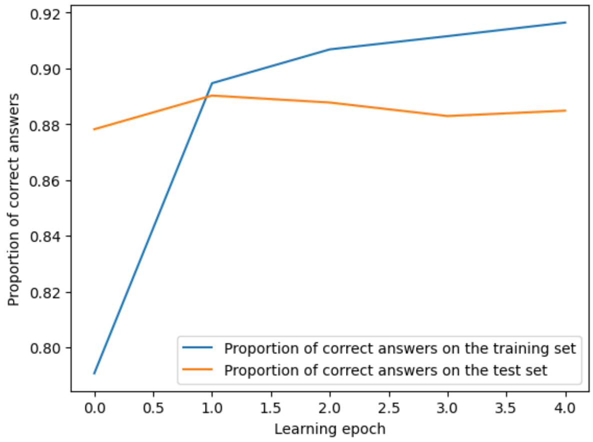
LSTM NN

```
model_lstm = Sequential([
    Embedding(num_words, 32, input_length=max_news_len),
    SpatialDropout1D(0.2),
    LSTM(16, dropout=0.2, recurrent_dropout=0.2),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(16, activation='relu'),
```

```
    Dropout(0.2),
    Dense(4, activation='softmax')])
```

```
model_lstm.summary()
```

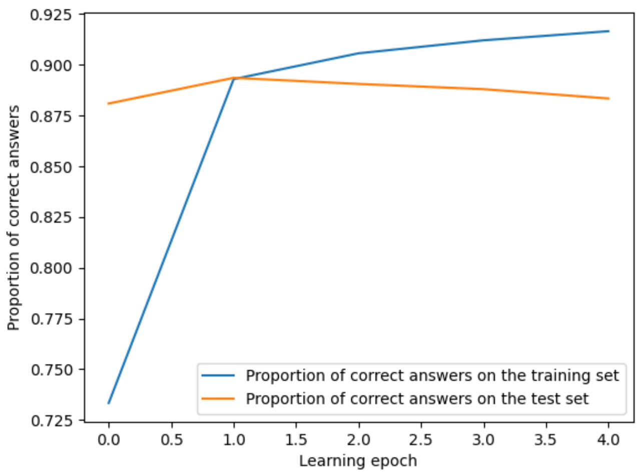
```
Model: "sequential_10"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_11 (Embedding)    (None, 30, 32)            320000

 spatial_dropout1d_5 (Spatia  (None, 30, 32)           0
 lDropout1D)

 lstm_4 (LSTM)               (None, 16)                3136

 dense_19 (Dense)            (None, 32)                544

 dropout_3 (Dropout)         (None, 32)                0

 dense_20 (Dense)            (None, 16)                528

 dropout_4 (Dropout)         (None, 16)                0

 dense_21 (Dense)            (None, 4)                 68

=================================================================
Total params: 324,276
Trainable params: 324,276
Non-trainable params: 0
_________________________________________________________________
```



GRU NN

```
model_gru = Sequential([
    Embedding(num_words, 32, input_length=max_news_len),
    SpatialDropout1D(0.2),
    GRU(16, dropout=0.2, recurrent_dropout=0.2),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(16, activation='relu'),
    Dropout(0.2),
    Dense(4, activation='softmax')])
```

```
model_gru.summary()

Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding_12 (Embedding)    (None, 30, 32)            320000    
                                                                 
 spatial_dropout1d_6 (Spatia  (None, 30, 32)           0         
 lDropout1D)                                                     
                                                                 
 gru_2 (GRU)                 (None, 16)                2400      
                                                                 
 dense_22 (Dense)            (None, 32)                544       
                                                                 
 dropout_5 (Dropout)         (None, 32)                0         
                                                                 
 dense_23 (Dense)            (None, 16)                528       
                                                                 
 dropout_6 (Dropout)         (None, 16)                0         
                                                                 
 dense_24 (Dense)            (None, 4)                 68        
                                                                 
=================================================================
Total params: 323,540
Trainable params: 323,540
Non-trainable params: 0
_________________________________________________________________
```



Adding more number of layers increased accuracy for train data set more for CNN rather than LSTM and GRU. At some point, number of layers almost didn't change the accuracy for LSTM and GRU models. Also, it caused overfitting on test set for all of them. Even adding dropout layers didn't change the situation much. The current number of layers is one of stable variants of models.

We can use current model in news companies such as The Guardian, The Economist and so on to save human resources and correctly identify news categories and topic. Furthermore, by tweaking the model, we can also use it to look for controversial words and topics to either avoid publishing news or fixing the language of it. It allows easier censorship and saves time reading whole article and looking for such words.