Data Science For Industry - Assignment 2

University of Cape Town - Department of Statistical Sciences

Prinavan Pillay(PLLPRI017), Simon Katende(KTNSIM001), Dirren Subbiah(SBBDIR001)

20/10/2018

Part 1 - President Predictor using Neural Networks

1. Introduction

The State of the Nation Address (SONA) is an annual event in South Africa which drives significant political and economic impact, since the president reports on the current status of the country. Gaining deeper insight into the various speeches from different presidents can provide a foundation for political prediction engines. A convolution neural network is used to predict which president was the source of a statement, given a particular sentence as an input. This can provide insight into the similarities and trends between different presidents' speeches, and determining possible overlap between key statements issues annually.

2. Convolutional Neural Network

Neural networks have gained popularity in recent times for complex classifications, due to advancements in distributed computing environments. Furthermore, neural nets allow one to artificially model the way the human brain makes decisions, which address complex classification tasks such as vision, speech, and natural language processing.

The President Predictor is built using the `R Keras` neural network library.

2.1 Data Preprocessing

Firstly, the data is read in and converted into an appropriate dataframe form. This involves storing the entire speech as a row entry of a dataframe.

```
txt_files <- list.files("/Users/prinpillay/Desktop/DSFI - Assignment 2/Data/sona-text-1994-2018/")
sona <- data.frame(filename = as.character(), speech = as.character())
for(i in txt_files){
  file_name <- paste0("/Users/prinpillay/Desktop/DSFI - Assignment 2/Data/sona-text-1994-2018/", i)
  # import text as single character string (can also read.table but the "seperator" causes problems)
  this_speech <- readChar(file_name,
                          nchars = file.info(file_name)$size)
  # make data frame with metadata (filename contains year and pres) and speech
  this_sona <- data.frame(filename = i, speech = this_speech, stringsAsFactors = FALSE)
  # make a single dataset
  sona <- rbind(sona, this_sona)
}
```

We then perform extra processing to determine the Year and the name of the President for a particular speech.

```
#Extract the year
sona$year <- str_sub(sona$filename, start = 1, end = 4)
#Extract all names removing txt
```

```
sona$president <- str_sub(sona$filename, start = 6, end=-5)
#Now we remove pre and post election prefixes
sona$president[str_detect(sona$pre,'post')]=
  str_sub(sona$president[str_detect(sona$pre, 'post')], start = 16)
sona$president[str_detect(sona$pre,'pre')]=
  str_sub(sona$president[str_detect(sona$pre, 'pre')], start = 15)
```

We then use tokenization to unnest the speech into sentences which can be used as inputs to the neural network model.

```
library(tidytext)
tidy_sona <- sona %>% unnest_tokens(text, speech, token = "sentences")
```

Exploratory analysis is used to determine the appropriate hyperparameters that will be used in the CNN model. This involves determining the unique number of words in all the speeches, the longest sentence and the frequency of particular words. The process of string manipulation involves unnesting the speech row into words, and then extracting the required features.

```
# Apply a sentence ID to the dataframe
tidy_sona$ID <- seq.int(nrow(tidy_sona))
#Unnest the data into words
tidy_sona2 = tidy_sona %>% unnest_tokens(text, text, token = "words")
#Check the most frequent words in all the text, according to sentence ID
tidy_sona2 %>% group_by(ID, text)  %>% summarize(count = n()) %>% arrange(desc(count))
#Check for the longest sentence
tidy_sona2 %>% group_by(ID)  %>% summarize(count = n()) %>% arrange(desc(count))
#Check for all unique words (count)
length(unique(tidy_sona2$text))
```

2.2 Model Architecture

Once baseline parameters surrounding the data are established, we proceed with initializing the CNN with hyperparameters required for the model. CNNs are particularly useful in the case of text mining, since they exploit not only the presence of certain words as predictors, but also the relationship between words. A part of this initialization phase includes setting the number of features (popular words), longest sentence, and any exclusions for particularly short sentences. Another important feature of neural nets is setting the embedding dimensions. This maps the text input to a higher dimensional feature space, which almost acts as a lookup table for the classifier.

```
#choose max_features most popular words
max_features <- 10000
# exclude sentences shorter than this
minlen <- 5
# longest sentence (for padding)
maxlen <- 340
# Number of unique words
input_dimensions <- 10000
#Random output dimension space
embedding_dims = 100
# Longest sentence
input_length = 340
```

We now need to transform the inputs and outputs into a form which can be fed into the CNN. This involves mapping the sentences into digits, with each digit representing a unique word across all sentences in speeches. This assists the CNN in numerically mapping the words as tensors which can be used to perform numerical

estimations. In terms of the outputs, the names of the various presidents need to be converted to categorical information.

```
#Convert text to digits based on the maximum number of features
tokenizer = text_tokenizer(num_words = max_features)
fit_text_tokenizer(tokenizer, tidy_sona$text)
sequences = tokenizer$texts_to_sequences(tidy_sona$text)

#Remove short sentences
seq_ok <- unlist(lapply(sequences, length)) > minlen
lengthIs <- function(n) function(x) length(x)>n
sequences <- Filter(lengthIs(minlen), sequences)

#Convert outputs to multiclass integer
y <- as.numeric(as.factor(tidy_sona$president[seq_ok]))
y = as.integer(y)
```

The next important step in model construction is splitting the data into a training and validation set. This is valuable to determine the out-of-sample error obtained when performing hyperparameter tuning for optimal model conditions. For this particular sample, a 90/10 split was used where 90% of the data is used for training, and 10% of the data is used for validation. Considering that text mining is a relatively difficult classification problem, we want to make as much data available for the training process to enhance the prediction accuracy.

```
#Generate test and train set
train <- list()
test<- list()
#Perform a 90/10 split for training/validation
train_id <- sample(1:length(sequences),
                   size = 0.9*length(sequences),
                   replace=F)
test$x <-  sequences[-train_id]
train$x <- sequences[train_id]

#We now also perform a split on the output
train$y <- y[train_id]
test$y <-  y[-train_id]
```

We also pad the inputs to the length of the longest sentences, to ensure all inputs are the same size.

```
x_train <- train$x %>% pad_sequences(maxlen = maxlen)
x_test <- test$x %>% pad_sequences(maxlen = maxlen)
```

Another important step is transforming the numerical outputs into a binary classification matrix as required by the Keras CNN.

```
#Transform train and test outputs into binary classification matrix
y_train= to_categorical(train$y)
y_test <- to_categorical(test$y)
```

We now use `Keras` to build the actual model. In terms of the model architecture, a 1D convolutional Neural Net is used with dropout regularization to prevent overfitting. A grid based hyper-parameter tuning technique is used, where a range of values were tested for optimal accuracy. Certain features were selected by experimentation (output dimensionality, kernel size, dropout ration, etc.) and other variables were determined due to the nature of the data. For example, softmax activation function was used due to mutually exclusive categorical classifications, the loss function used categorical cross entropy due to the binary classification output provided, etc.

```
model <- keras_model_sequential()

#Define model hyperparameters - determined experimentally using grid-based technique
model %>%
  # embedding layer maps all the features to the higher dimensional space for lookup
  layer_embedding(max_features, embedding_dims, input_length = maxlen) %>%
  # add some dropout
  layer_dropout(0.3) %>%
  # convolutional layer
  layer_conv_1d(
    filters = 250, #Output dimensionality
    kernel_size = 3, #Length of CNN window
    padding = "valid",  # padding was already done
    activation = "relu", #Rectified linear activation function for this layer
    strides = 1 #The amount by which the CNN filter shifts
  ) %>%
  layer_global_max_pooling_1d() %>% #Useful for sequential data where ordering is important ie. text
  layer_dense(128) %>%
  layer_dropout(0.3) %>%
  layer_activation("relu") %>%
  layer_dense(7) %>%   # 7 possibilities for output layer (6 presidents and a row of 0s - NULL)
  layer_activation("softmax") #Useful when classes are mutually exclusive

#Compile the model with loss function and other metrics
model %>% compile(
  loss = "categorical_crossentropy",
  optimizer = "adam",
  metrics = "accuracy"
)
```

2.3 Model Implementation

The training data is now used to fit a model and the accuracy is tested with the validation set. Th

```
#Fit model
model %>%
  fit(
    x_train, y_train,
    batch_size = 32,
    epochs = 6,
  )
#Validate model
model %>% evaluate(x_test, y_test)
```

We then reverse the output to get meaningful results from the classification process

```
Predictions=model %>% predict(x_test)
#Reverse output into president numbers
Prediction_num=apply(Predictions, 1, function(row) which(row==max(row))-1)
#Obtain the mapping between president names and numbers
pres_names=unique(y)
pres_nums=unique(tidy_sona$president)
#pres_names
#pres_nums
```

```
Prediction_num[Prediction_num==3]='Mandela'
Prediction_num[Prediction_num==2]='deKlerk'
Prediction_num[Prediction_num==4]='Mbeki'
Prediction_num[Prediction_num==6]='Zuma'
Prediction_num[Prediction_num==1]='Motlanthe'
Prediction_num[Prediction_num==5]='Ramaphosa'
#Final Predictions
Final_pred= Prediction_num
```

3. Discussion of Results

As shown above, after best accuracy after experimentally performing hyperparameter tuning was around 63.2%. Considering that the system uses a locally trained model with a 6 class classification, this represents a successful first line approximation at building a SONA President predictor given any input sentence. Furthermore, considering that there was an unbalanced dataset since there were much more training sentences for certain president, there might have been inherent bias in the dataset which could have possibly skewed the results and accuracy. Future recommendations would include testing the system on a more balanced set, and employing text generation techniques to fill in sentences for presidents with much lower sentence counts. In terms of the model building process, future recommendations would be to run the model generation process on a distributed computing environment for a thorough hyperparameter search to further optimize the accuracy.

4. Conclusion

A successful Neural network was designed, implemented and tested to obtain a validation accuracy of 63.2% when provided a sentence to predict a president from a set of SONA speeches used as the training data.

Part 2 - Descriptive Analysis

Exploratory Data Analysis

The data set consists of 30 State of the Nation Address (SONA) speech transcripts from all the presidents from February 1994 til recently in February 2018.

A hypothesis can be made that sentiments of speeches differ depending on the political season.There are 3 political seasons :

1. Pre-Election
2. Post-Election
3. Normal Term

Below is a summary of all the speeches as per the various presidents categorized by the 3 political seasons :

Period	Presidents	Pre-Election	Post-Election	Normal Term	**Total**
1994	de-Klerk	1			**1**
1994-1999	Mandela	1	2	4	**7**
2000-2008	Mbeki	1	1	8	**10**
2009	Motlante	1			**1**
2009-2017	Zuma	1	2	7	**10**
2018	Ramaphosa			1	**1**
Total		**5**	**5**	**20**	**30**

Table 1: The number of SONA per president and arranged by political seasons

From the table above the following remarks can be made :

- de Klerk,Motlante and Ramaphosa all have one speech each which will make it extremely difficult to accurately predict given the far higher number of speeches from their counterpart presidents.
- There are far more speeches done during the normal season which will inherently bias the training data towards that season
- Mbeki and Mandela dominate the number of speeches with 10 apiece.This will also inherently bias the training data towards them.
- Pre-Election speeches are evenly distributed across 5 of the 6 presidents whilst post election speeches are dominated by Mandela and Zuma.

Word Distribution

Below are the most frequently used words of all the 30 presidential speeches rescaled according to their respective political seasons :

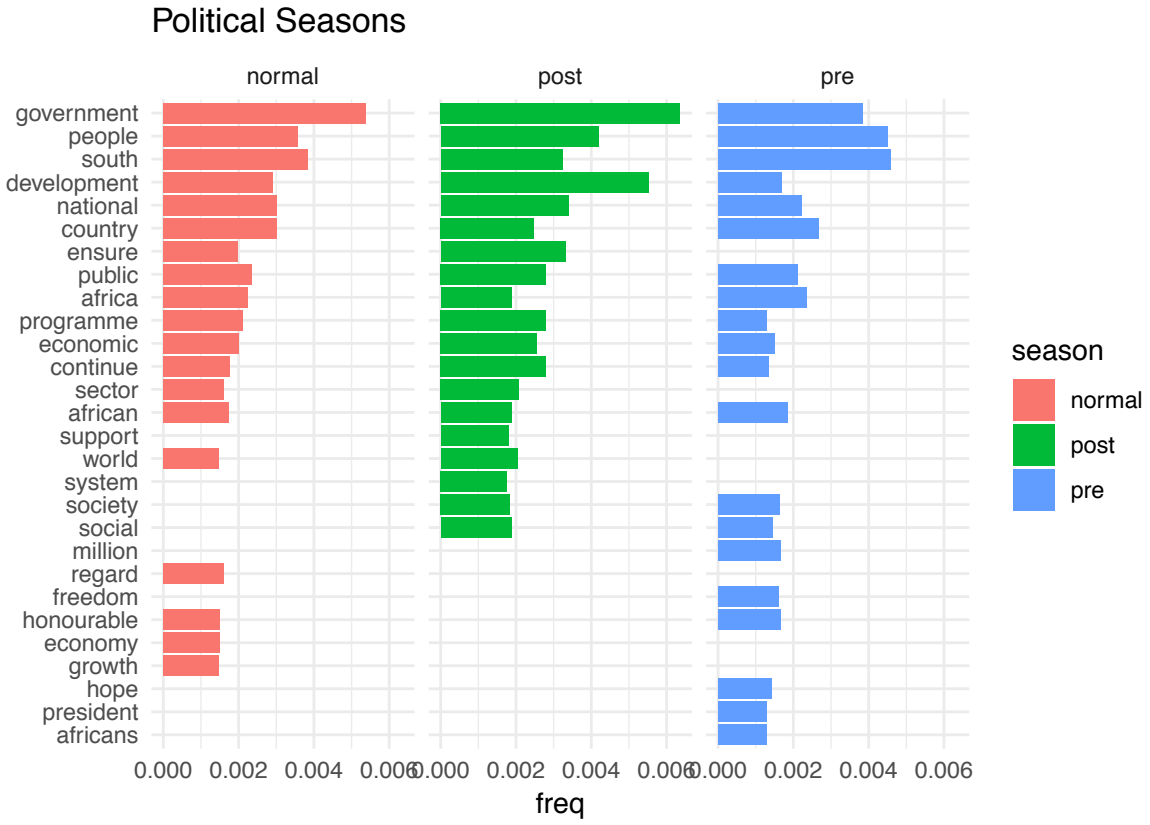

Figure 1: Frequently used words across political seasons

From the illustration above the following remarks can be made :

- Notable words commonly used across all political seasons are **south**, **africa**,**government**,**development**,**people**,**country** words will potentially not assist in distinguishing the respective presidents.
- Comparing pre and post to normal political seasons ,notable words like **economy** and **growth** are introduced into their speeches .The utilization of these words (which could imply economic growth) are understandable given that these are typical themes that need to be addressed constantly throughout the normal period of presidential terms.
- Comparing pre to post political seasons,uplifting words like **freedom**,**hope**,**africans** are used before and not after elections.
- Comparing pre to post political seasons,notable words introduced are **support**,**system**,**ensure**.These words convey a theme of action and execution which is expected after coming from an election.

Below are the most frequently used words of all the 30 presidential speeches rescaled according to the respective presidents :

Figure 2: Top words used by all presidents in SONA

From the illustration above the following remarks can be made :

- Looking at all the presidents frequently used words ,de Klerk has the least common words.This is due to the fact that firstly there is only one speech in the dataset and secondly given that the speech was before the first democratic elections,the content will be far different to the speeches made by the presidents that preceded in the democratic era of South Africa.
- Commonly used words across all presidents are **south**,**government**,**national** which are also common words across political seasons.
- Notably words commonly used across all presidents excluding deKlerk are **people**,**country**,**public**,**ensure**,**development**

Clustering by Term Similarity

Words from the respective speeches we aggregated by the respective presidents.Words greater than 4 letters were considered so as to focus primarily on descriptive words.Sparse elements that did not reflect at least 40% of the time in the resultant document matrix were removed. The resultant word counts were then normalized to avoid biases of presidents with more speeches .

K means clustering was then conducted and a $k = 4$ was selected based on the 'elbow rule'.

The objective is to see what frequent common words do the presidents use and what potential themes to these similar words posses.The resultant visualization can be viewed below:

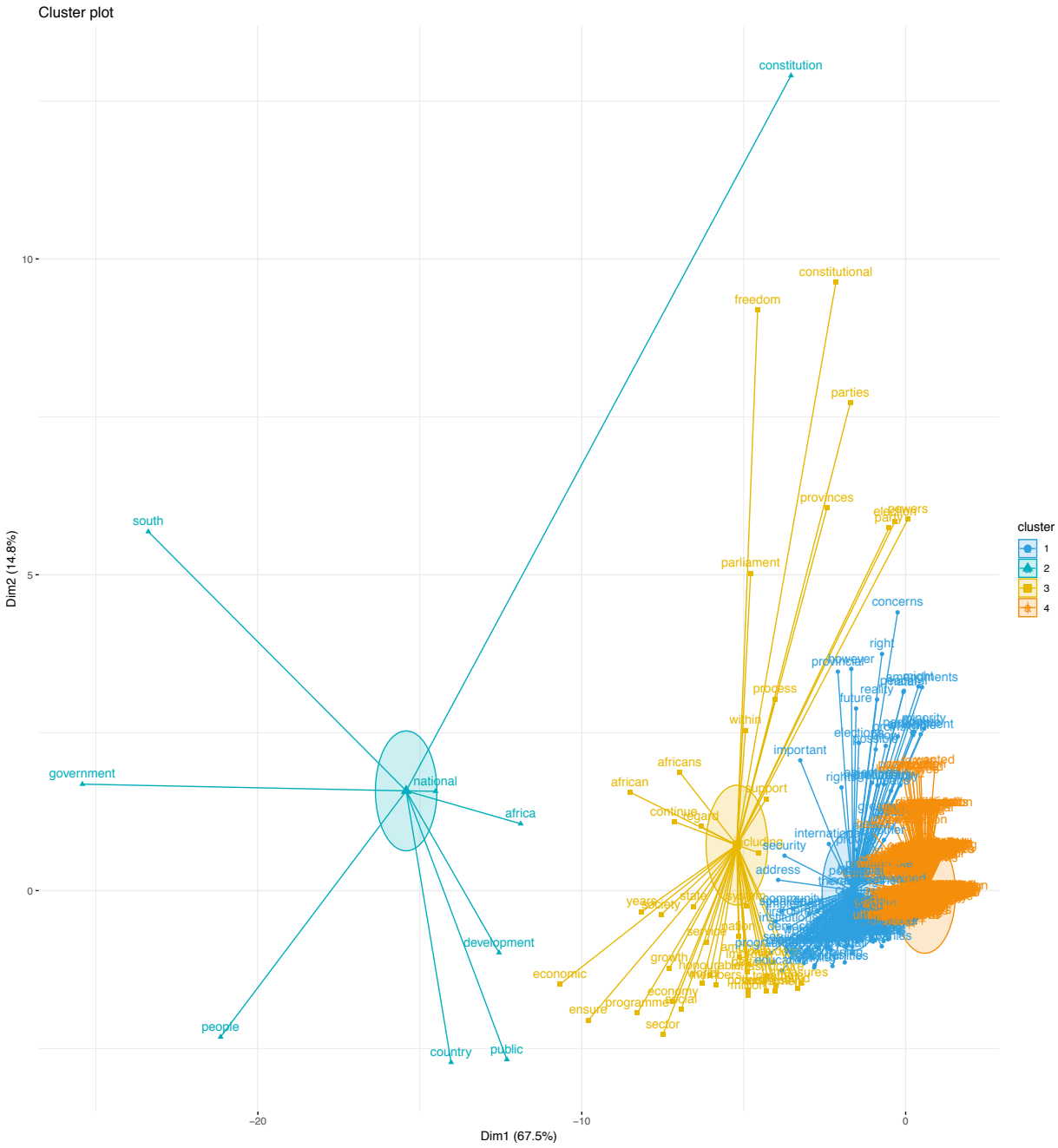

Figure 3: K means clustering of speech words of the 5 presidents

From the illustration above the following remarks can be made :

- Approximately 84 percent of the variability resides in the first two principal components.
- Cluster 2 has words like **south africa** ,**country**, synonymous with SONA introduction/concluding paragraphs that all presidents will generally mention.
- Cluster 1 portrays a very serious tone.The theme that comes to mind is pertinent issues that need to be addressed.Words include **concerns**,**security**,**education**,**address**.

- Cluster 3 portrays a more warm theme with a slight patriotism undertone.Notable words include **africa**,**growth**,**freedom**,**economic**,**service**.

Sentiment Analysis Across Political Seasons

The SONA of before and after the elections over the years are compared to determine whether there is an inherent tone difference.The *bing* lexicon was utilized in this analysis.The sentiment of the words in the lexicon were summed up to determine the net sentiment which will be referred to as polarity.Below are the results

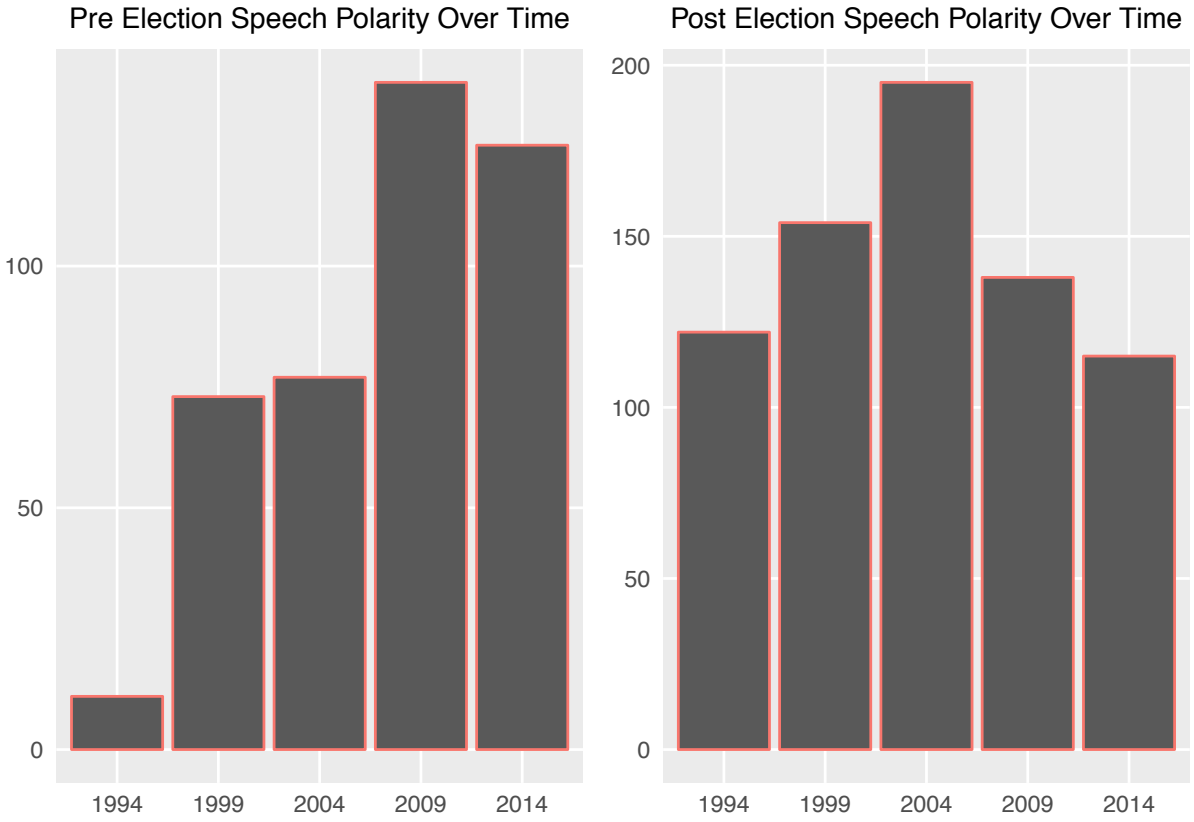

Figure 4: Pre vs Post Election Speech Sentiment over time

From the illustration above the following remarks can be made :

- There is an interesting behavior in the first 3 election years (1994,1999,2004) there was an improvement in overall sentiment after the elections,however in the more recent 2 election years (2009,2014) there has been a more cautionary tone after elections
- The biggest overall sentiment disparities before and after elections occur in 2004 and 2009.The 2004 pre/post comparison is interesting - both speeches were done by Mbeki and it showed the highest increase in net sentiment in any given year.This could be due to the fact that Mbeki had just won his second election and wanted to reassure the country with a positive speech.The 2009 pre/post comparison is counter intuitive- one would expect that given that when Zuma was elected into power for the first time that he would have a higher net sentiment then that of the his predecessor who was there on a temporary basis.

Sentiment Analysis Across Presidential Terms

Sentiment analysis was conducted on presidents with at least a full presidential term to observe where there is seasonality on with their net sentiment over their respective terms.The presidents were:

1. Mandela
2. Mbeki
3. Zuma

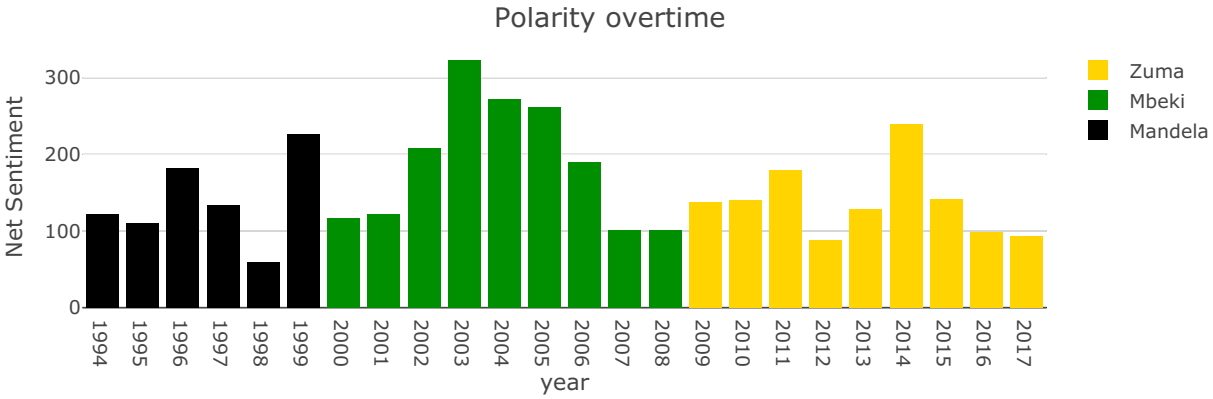

Figure 5: Polarity of Presidents across their terms

From the illustration above the following remarks can be made :

- Net Sentiment across the 3 presidents tend to trend up in their first 3 years of service followed by a downward trend .
- The most startling change in sentiment was Mandela in his last year of service who finished off on a vary high sentiment note.

Conclusion

The objective of this section was to conduct an exploratory text analysis of the SONA speeches.This was conducted with the additional of looking at political seasons and how that influences sentinemt. 5/6 Presidents utilize the same common words which are generally topical themes in South Africa. It has been observed that there is an improvement in net sentiment of speeches after elections. Presidents with at lease one term tend to exhibit an increase in sentiment in their first 3 years followed by a decrease in sentiment.

Group Work Contribution

All group members were involved in the entire design and implementation, with certain members having key focus areas as listed below:

Part A: Neural Network

Prinavan Pillay

Dirren Subbiah

Part B: Descriptive Analysis

Simon Katende

Collaboration Tool

This report was compiled by all group members equally. Code collaboration was conducted using the following GitHub repo:

https://github.com/katendencies/datasci_fi_assignment

Appendix

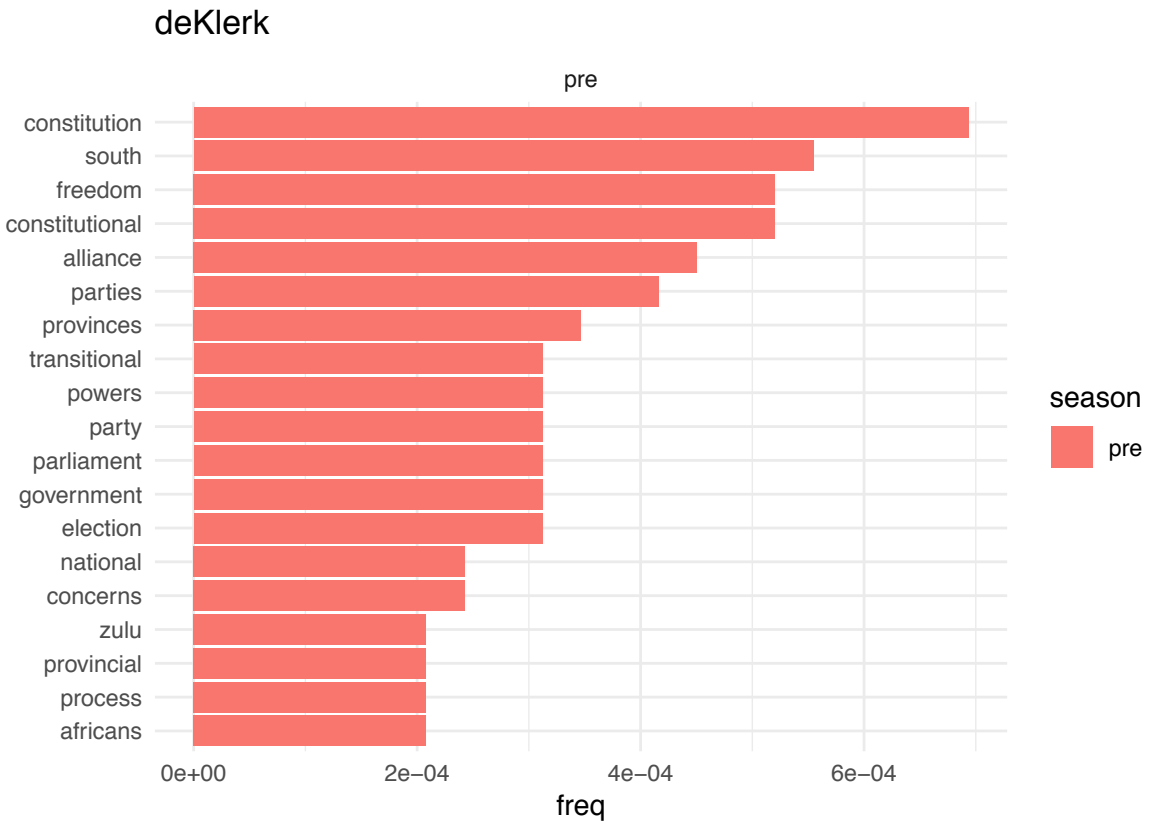

Figure 6: Top Words used by de Klerk

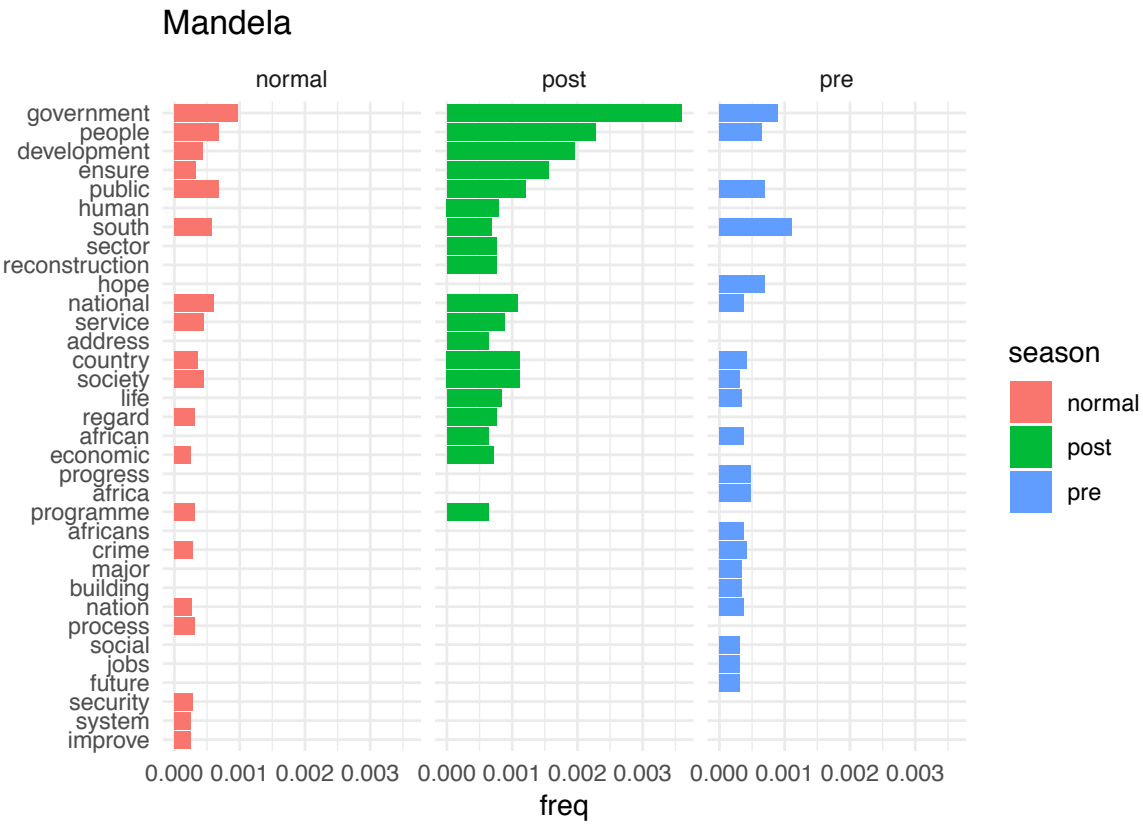

Figure 7: Top Words used by Mandela

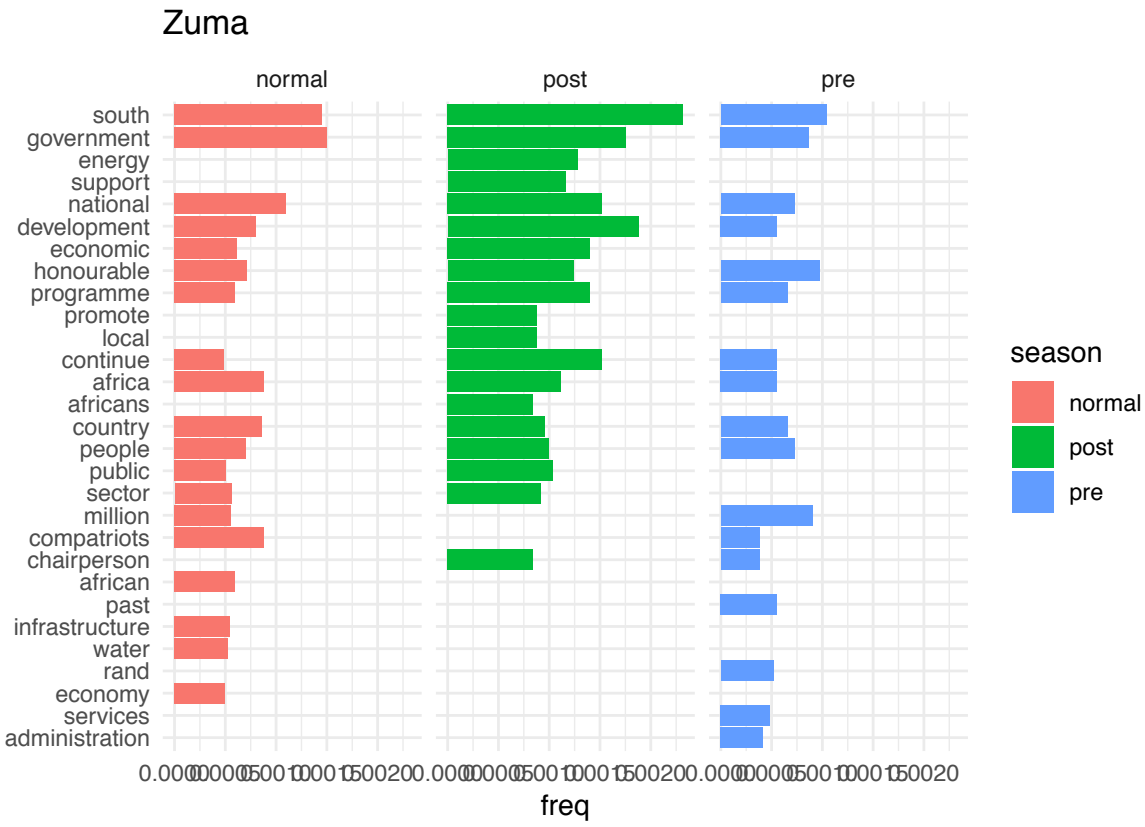

Figure 8: Top Words used by Zuma

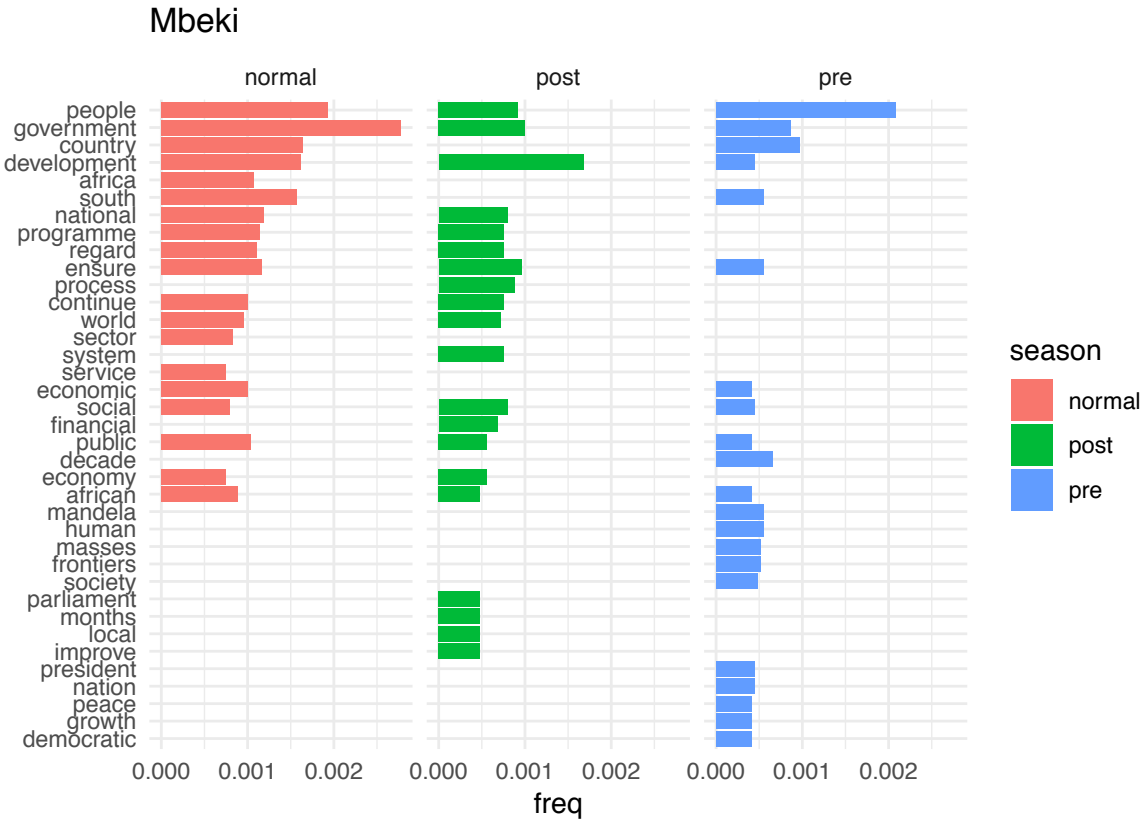

Figure 9: Top Words used by Mbeki

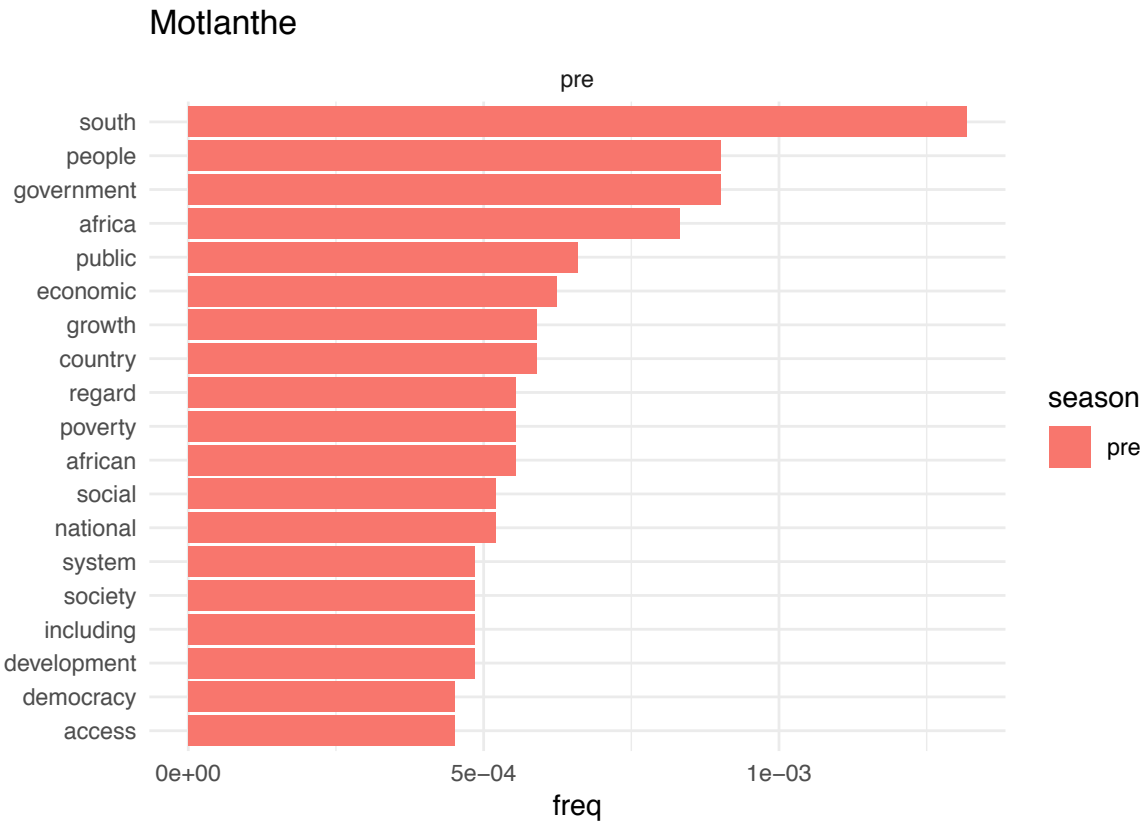

Figure 10: Top Words used by Motlanthe

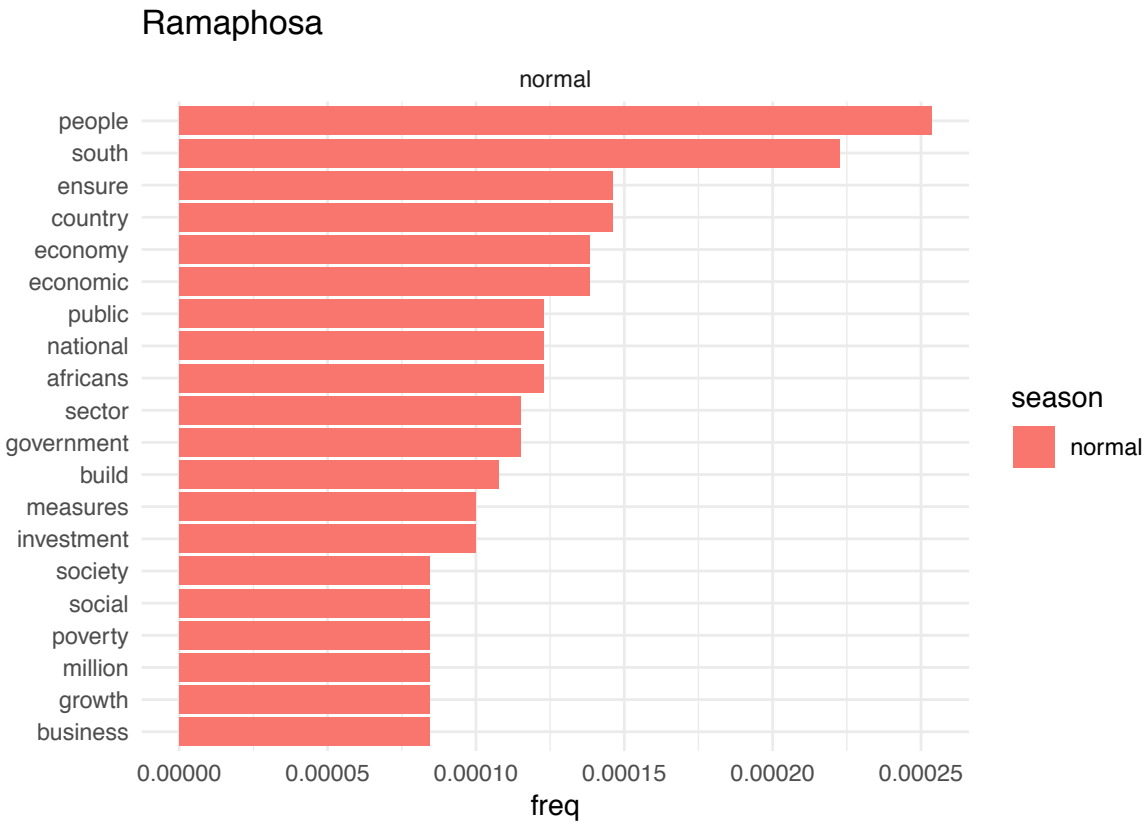

Figure 11: Top Words used by Ramaphosa