COVENTRY UNIVERSITY

Faculty of Engineering, Environment and Computing School of Computing, Electronics and Mathematics

MSc. Data Science and Computational Intelligence

7151CEM

Computing Individual Research Project

Analysing Topic-Sentiment Evolution for the US 2020 Elections using a Dynamic Bayesian Network

Author: Jack Berry

SID: 7785614

Supervisor: Dr. Alireza Daneshkhah

Submitted in partial fulfilment of the requirements for the Degree of Master of Science in Data Science & Computational Intelligence

Academic Year: 2022/2023

Abstract

In recent times more and more researchers have gravitated towards sentiment analysis due to the complexity of human language. It is this complexity that creates a challenge for researchers to come up with novel methods of enabling computers to accurately interpret human language. Recently, most works have aimed to analyse how sentiment evolves over time yet analysing how the sentiment of topics influence one another has only been done once (Liang et al. 2020). Despite this previous attempt, the approaches used are outdated, paired with the use of suboptimal datasets. In this paper I propose a more modern alternative referred to as TimeLM-21, produced by Loureiro et al., to perform the sentiment analysis with the aims of producing a more suitable DBN. Both the previously used CNN approach and the TimeLM-21 approach were used to perform sentiment analysis on a dataset containing 1748235 tweets referring to the US 2020 elections, this was then used to produce two DBNs. These two DBNs were then compared using AIC and BIC values where it was discovered that the DBN produced via the TimeLM-21 approach was more suitable. Despite this, the computational resources required for TimeLM-21 heavily outweighed the requirements necessary for the CNN.

Code available at: https://github.com/B3rry99/ProjectCode

Table of Contents

Declaration of Originality

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: Jack Berry Jack Berry Date: 08/12/2022

Please complete all fields.

First Name:	Jack
Last Name:	Berry
Student ID number	7785614
Ethics Application Number	P142441
Supervisor Name	Dr. Alireza Daneshkhah

This form must be completed, scanned, and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

1 Introduction

In the past, opinions of other people were primarily only accessible via reading topic-specific books/articles or by actively listening to others speak. In the modern era with the creation of the internet people have been given the ability to express their opinions to huge crowds of people in the space of minutes, in January 2022 the active social network penetration was estimated to be 58.4% globally (Tankovska, 2019). The most discernible examples of these social media websites are Reddit, Twitter, and Facebook. When looking at Twitter specifically, users are limited to publishing only short form text content with a limit of 280 characters, in addition to this, users can use hashtags to specify the topic they're talking about. These two features of Twitter, along with over 500 million tweets (Sayce, 2010) being published per day makes Twitter a very popular candidate for researchers to perform sentiment analysis-based research.

Sentiment analysis (SA) is a heavily researched area due to the numerous applications it applies to. Some such examples include customer relationship management, market research, voice of the customer (VoC), and product analysis. When looking at product analysis specifically, one could collect many tweets which make use of the "iphone14" hashtag once the product is announced to get a general feel for the public sentiment towards the product ahead of release. An additional example is collecting tweets containing the hashtags of people running for presidency such as Trump and Biden in 2020, this analysis could be a potential predictor of the winner.

Due to the potentially lucrative results SA can produce it is very popular area for researchers to continue to innovate and improve upon previous models. Some common methods for performing SA are convolutional neural networks (CNNs), long short-term memory (LSTM), Naïve bayes, and transformers. We can take SA a step further by performing the analysis over a given time period to gain an understanding of how public sentiment changes over time; this is referred to as Sentiment Evolution. However, researchers are also curious about how the sentiment of one topic can be affected by sentiment on a related topic over time. This problem can be solved using Dynamic Bayesian Networks (DBNs).

DBNs have been popularly used for modelling the dynamic interaction of multiple entities in time series data (Liang et al., 2020). In recent years, DBN models have seen a lot of use in industrial settings due to their characteristics as time series forecasting models and their interpretability, which has changed DBNs into more general use models (Quesada et al., 2022). Some examples of their uses are modelling sentiment analysis evolution (Liang et al. 2020), ecosystem changes based on climate variations (Trifonova et al., 2019), stock market prediction (Duan, 2016), and controlling aircraft wing cracks over time (Li et al., 2017).

Despite a previous attempt to model SA evolution using DBNs by Liang et al. there were weaknesses in the dataset used to train their CNN. These weaknesses include out of date text, small sample size, and a small overlap of vocabulary between the training dataset and case study dataset. In this paper I will be using a more modern and effective model for SA known as TimeLM-21 (proposed by Loureiro et al. 2021) which will result in a more accurate DBN, this will be performed on tweets relating to the US 2020 elections.

1.1 Problem Motivation

In the modern age public opinion appears in almost every corner of the internet. Places such as Twitter, Reddit, Facebook, and generic review sections allow users to express their sentiment on a multitude of topics. Researchers and companies alike can utilise the vast number of user-published comments to achieve a wide variety of objectives such as product marketing, company reputation management, and predicting stock pricing.

Dynamic Bayesian networks (DBNs) have been used to extend standard Bayesian networks with the concept of time which allows the user to model time series of sequences. DBNs have seen use in modelling the dynamics and interactions of the sentiment of topics over time on social media such as Twitter, this was performed in Liang et al.'s paper focussing on Brexit. This work presented some weaknesses by using an outdated method for performing sentiment analysis paired with using a sub-optimal dataset. Therefore, in this paper I propose a more modern and effective method for performing sentiment analysis which, in turn, should provide a more suitable DBN, this will be done on a dataset containing tweets relating to the US 2020 elections.

1.2 Project Statement/Objectives

The main goal of this project is to use a modern sentiment analysis approach to produce accurate results which will be used to produce a DBN. This goal provides us with two research questions:

1) Can a modern sentiment analysis approach be used to produce a DBN analysing US 2020 election sub-topics?
2) Does this modern approach provide a more suitable DBN than previous works?

1.3 Report Structure

This report will be divided into 7 sections, each of which is described below:

Section 1) Covers the introduction, problem motivation, and problem statement/objectives to provide the reader with background information relevant to this project.

Section 2) Covers a literature review aiming to highlight various author's works on similar topics.

Section 3) Covers the methodology of this project including the datasets used and model architectures.

Section 4) Covers the experimental setup, explaining how the data pre-processing, sentiment analysis, and DBN were executed.

Section 5) Covers all results produced. This includes data exploratory analysis, sentiment analysis, the DBNs produced, and a comparison of suitability between DBNs produced via CNN/TimeLM-21 approaches.

Section 6) Covers a brief conclusion of the project along with potential future works to improve on various weaknesses of this project.

Section 7) Covers the achievements of this project and student reflections.

Section 8) Covers references and bibliography

Section 9) Covers the appendix

2 Literature Review

The sudden upsurge in interest in new systems that deal with opinions as a first-class object has caused the sudden eruption of activity in the field of opinion mining and SA, which deals with the computational treatment of opinion, sentiment, and subjectivity in text (Pang & Lee, 2008). There have been numerous attempts at innovating new methods of performing SA along with reimagining existing methods to improve accuracy. As mentioned in the previous section the most prevalent methods for carrying out SA include CNNs, LSTM, Naïve Bayes, and more recently, Transformers. Typical approaches to classifying the sentiment of text include the 5- or 10-point scale, or simply labelling the text as "negative", "neutral", or "positive".

In a 2019 paper by Wongkar et al. they aimed to compare the effectiveness between the Naïve Bayes, K-Nearest Neighbours (KNN), and support vector machine (SVM) methods. Using a web crawler, tweets were collected from twitter relating to the 2019 Indonesian presidential elections and each method was employed and compared against one another. The tweets were initially cleaned by removing URLs, emojis, and punctuation and then the remaining text was tokenized. Following this the Naïve Bayes process was performed on the tokenized words, followed by calculating the value of the class probability by dividing the number of class data with the total or number of existing documents. It was discovered that the Naïve Bayes method performed the best with an accuracy of 80.1% followed by KNN (75.58%) and then SVM (63.99%). A problem with this particular method is the exclusion of emojis and punctuation during the data pre-processing. Emojis and punctuation (when used to create smiley/sad faces etc) can be a major indicator of sentiment for the given tweet, this missing information can potentially limit the accuracy of the SA.

A 2016 paper by Li et al. aimed to build a SA model using a recurrent neural network (RNN) based upon LSTM. The purpose of this paper was to attempt to improve upon the conventional RNN language model. Their experiment began by dividing the training data into several categories, according to their emotional labels. The LSTM models are then trained for each category of data, resulting in several LSTM models, each for the associated emotional reviews. The LSTM models developed during the training phase are assessed using the new input review (providing error values) in order to predict the emotion of a new input review. The emotional category for the new input review is assigned to the model with the lowest error value.

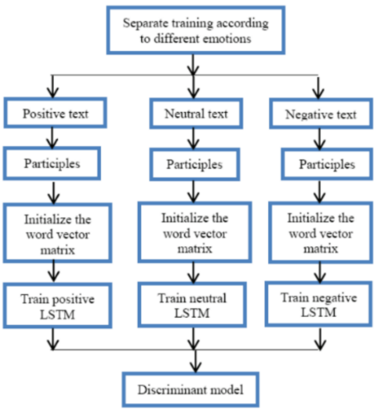

Figure 1 – Example of model architecture used in Li et al. 2016

In the results of their experiments, they found that using RNN based LSTM, their accuracy rate increased by 0.8%, 2%, and 0.85% for positive, negative, and neutral text, respectively. One strength that this model has over the Naïve bayes model is that it takes into account the order of words in a sentence rather than only looking at each word independently. Despite this improvement, this added complexity comes at a cost of additional computational resources commonly found with all deep learning models.

Instead of using RNNs a paper by Gao et al. instead aimed to perform SA using a CNN built on LSTM, the purpose of this was to compare accuracy to models using only CNNs or LSTM models separately. This was done by first pre-processing the text data in order to segment the text string into corresponding word sequences. Following that, Word2Vector maps each word in the tokenized text data into a d-dimensional word embedding. The text features of the text were selected by the LSTM model, CNN model, or the CNN model build on LSTM. Finally, the sentiment analysis of the comment was accomplished using the neural network-based sentiment classifier. In the following diagram Jieba refers to the library used for segmenting the text.

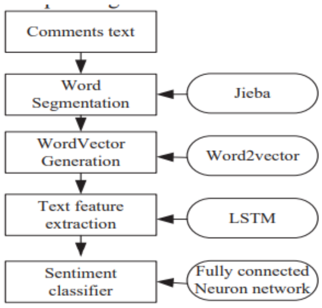

Figure 2 – Example of LSTM architecture for SA (Gao et al. 2019)

The results of this experiment demonstrated an increase in accuracy of 3.13% and 1.71% respectively, when compared to the CNN model and LSTM model. This provides evidence that hybrid models are more effective than CNN or LSTM standalone models.

A more recently developed model was proposed by Devlin et al. in 2018 which used a method named Bidirectional Encoder Representations from Transformers (BERT) based on a Transformer developed by Vaswani et al. (2017). BERT was designed to pretrain deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers. As a result, without making significant task-specific architecture alterations, the pre-trained BERT model may be improved with just one additional output layer to produce cutting-edge models for a variety of tasks, including question answering and language inference. After performing numerous experiments, the BERT model managed to obtain an accuracy of 80.5% for the GLUE score which was an improvement of 7% over the previous state of the art. The GLUE score stands for General Language Understand Evaluation which is a benchmark proposed by Wang et al. which includes a collection of diverse natural language understanding tasks, one of which is SA.

Despite this large improvement over the previous state of the art, it was discovered by Lie et al. (2019) that BERT was both undertrained and not given enough training passes, therefore in their paper they produced an improved version of BERT referred to as RoBERTa. The key differences they made to BERT was increasing the size of the training set and altering hyperparameters. More specifically, they trained the model longer, with bigger batches, and over more data. Furthermore, they removed the next sentence prediction objective, trained on longer sequences, and dynamically changed the masking pattern applied to the training data. When performing the analysis on the GLUE benchmarks the RoBERTa model outperformed the previous state of the art BERT model on all 9 of the task development sets.

	MNLI	**QNLI**	**QQP**	**RTE**	**SST**	**MRPC**	**CoLA**	**STS**	**WNLI**	**Avg**
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

Figure 3 – Comparison of accuracies across task development sets (Lie et al. 2019)

In 2021, Loureiro et al. proposed a further improved version of RoBERTa referred to as TimeLM. They state that currently both from model development and evaluation standpoints, this paradigm is essentially static, which affects both the ability to generalize to future data and the reliability of experimental results, since it is not uncommon that evaluation benchmarks overlap with pre-training corpora (Lazaridou et al. 2021). Specifically, both the BERT and RoBERTa model are trained on outdated corpora, a key example being COVID-19 related discussion. Due to this, when using these models on modern text data they may well struggle to determine what people are talking about when words such as “covid” or “lockdown” are used, amongst other examples. To remedy this flaw, Loureiro et al. collected large amounts of tweets collected over an evenly distributed amount of time. Furthermore, in order to collect a corpus that represents general conversation they collected tweets based on the most frequent stop-words. In total they trained the TimeLM model on over 123M tweets.

Models	**Additional**	**Total**
2019-90M	-	90.26M
2020-Q1	4.20M	94.46M
2020-Q2	4.20M	98.66M
2020-Q3	4.20M	102.86M
2020-Q4	4.20M	107.06M
2021-Q1	4.20M	111.26M
2021-Q2	4.20M	115.46M
2021-Q3	4.20M	119.66M
2021-Q4	4.20M	123.86M
2021-124M	33.60M	123.86M

Figure 4 – Number of tweets trained for TimeLM-21 (Loureiro et al. 2022)

	Emoji	**Emotion**	**Hate**	**Irony**	**Offensive**	**Sentiment**	**Stance**	**ALL**
SVM	29.3	64.7	36.7	61.7	52.3	62.9	67.3	53.5
FastText	25.8	65.2	50.6	63.1	73.4	62.9	65.4	58.1
BLSTM	24.7	66.0	52.6	62.8	71.7	58.3	59.4	56.5
RoBERTa-Base	30.8	76.6	44.9	55.2	78.7	72.0	70.9	61.3
TweetEval	31.6	79.8	55.5	62.5	81.6	72.9	72.6	65.2
BERTweet	33.4	79.3	56.4	**82.1**	79.5	73.4	71.2	67.9
TimeLM-19	33.4	**81.0**	**58.1**	48.0	**82.4**	73.2	70.7	63.8
TimeLM-21	**34.0**	80.2	55.1	64.5	82.2	**73.7**	**72.9**	66.2
Metric	M-F1	M-F1	M-F1	$F^{(i)}$	M-F1	M-Rec	AVG (F1)	TE

Figure 5 – Comparison of accuracies using “TweetEval” (Loureiro et al. 2022)

When evaluating the final iteration of the model (TimeLM-21) they found that it outperformed the original RoBERTa model on all seven of the downstream tweet classification tests collectively referred to as TweetEval (Barbieri et al. 2020).

Since there are many different models proposed for performing SA it is important to research which models perform the best before selecting one. A paper by Sinha et al. in 2022 aimed to compare multiple models when performing SA, these models include CNN, LSTM, CNN-LSTM, LSTM-CNN, GRU, BERT, BERT-CNN, and BERT-LSTM. By performing SA on movie reviews from IMDb the experiments concluded that the BERT-LSTM and BERT-CNN models were the most effective with the standalone BERT model following very closely behind.

Model	Precision	Recall	F1
CNN	80.38	84.98	82.61
LSTM	83.25	82.33	82.78
CNN-LSTM	81.12	81.45	81.29
LSTM-CNN	82.39	78.21	80.25
GRU	84.89	84.02	84.45
BERT	87.52	88.39	87.95
BERT-CNN	89.61	88.39	88.98
BERT-LSTM	89.13	89.32	89.22

Figure 6 – Comparison of models performing SA (Sinha et al. 2022)

It is worth noting that the increase in improvement from the hybrid BERT models in comparison to the standalone BERT model was only marginal, there were two reasons stipulated for this outcome. The first reason is that the reduced length of the comment allows for the elimination of potential confusions during the pre-processing making the comment classification process simpler. The second reason is simply due to the very high computational resources required for the hybrid BERT models which could not be fully met when performing the experiments.

DBNs have become increasingly popular in the last decade with applications ranging from stock price prediction to aircraft wing health monitoring. DBNs models the probabilistic conditional independence relationships of the variables in data over time, representing the effect that past instants have on the present. This model can perform probabilistic reasoning over any chosen collection of objective variables given that others have been observed, and it also facilitates a better understanding of a system through its graphical depiction (Quesada et al., 2022). Furthermore, DBNs allow us to forecast future conditions based on the evolution from previous fixed conditions.

A closely related piece of work was produced by Liang et al. (2020) which aimed to perform topic-sentiment evolution analysis on Brexit related tweets using a CNN and DBN. In this paper they opted to use a popular CNN approach proposed by Kim (2014) to detect the sentiment orientation of the data which they then aggregated into time bins based on what day the tweet was posted. They then constructed the DBN based on the aggregated input sentiment analysis to provide a graph demonstrating the interactions between various topics of tweets. The resulting DBN graphical representation displayed multiple clear to interpret links between various topics. One particular flaw with this approach was how their CNN was trained. They opted to train their CNN on a dataset containing 14640 tweets from 2015 that contained a mention of various US airline companies. This is problematic due to the dataset being outdated, containing a small sample size, and including a too specific corpus. Due to the nature of all the training tweets containing a mention of airline companies it is likely that most of these tweets are complaints/appraisals of user's flights or airport experiences. Because of this, the vocabulary used in these tweets likely has little overlap with tweets mentioning Brexit and therefore will not only affect the accuracy of the SA but also the DBN produced using the SA results.

In this paper I will be opting to use the TimeLM-21 model to improve the accuracy of the DBN that was produced by Liang et al. (2020) by improving the initial SA accuracy. The justification for this decision is based upon the comparison results produced by Sinha et al. and Loureiro et al. displayed above. We can see that not only does the BERT model outperform traditional CNN/LSTM approaches but also that the TimeLM-21 model outperforms the BERT model. Furthermore, the TimeLM-21 model is trained on a larger, more modern, and more linguistically diverse dataset which should result in a more accurate DBN being produced.

3 Methodology

3.1 Primary Dataset

The dataset used in this project was taken from Kaggle.com, an online repository of datasets open to the general public. The dataset contains two files, one containing tweets that make use of hashtags referring to Trump (971157 tweets) , and the other containing hashtags referring to Biden (777078 tweets). The producer of the dataset used the Twitter API "statuses_lookup" and "snsscrape" to collect tweets with the aforementioned hashtags over the course of 7 days from 15/10/2020 to 22/10/2020. The producer then collected approximately 32000 tweets each day following 22/10/2020 up until 08/11/2020 which resulted in a total of 1748235 tweets. The reason for collecting only ~32000 tweets per day is due to the Twitter API's limit on how many tweets users can scrape per day which makes finding large datasets more difficult.

US Election 2020 Tweets		
Feature name	Description	Data type
Created_at	Date and time tweet was created	Date
Tweet	Tweet content	String

Table 1 – US 2020 elections dataset

The original dataset contained 21 features but since the research only requires tweet content and date posted most features are omitted, this also upholds privacy issues. Exploratory data analysis will be shown in the results section.

3.2 Validation Dataset

The dataset used to validate the SA model's accuracy before using it on the primary dataset is also taken from Kaggle.com. It is worth noting that this is the same dataset used to train the CNN used in Liang et al.'s paper. This dataset contains 14640 tweets that all contain a mention of various US airline companies. Table 2 shows the original dataset.

Twitter US Airline Sentiment		
Feature	Description	Data Type
Airline_sentiment	Sentiment of tweet (negative,neutral,positive)	String
Airline_sentiment_confidence	Confidence level of sentiment (0-1)	Float
Text	Tweet content	String

Table 2– US Airlines dataset

Much like the previous dataset most of the original features are irrelevant for the purpose of this research therefore the only features kept are "text", "airline_sentiment", and "airline_sentiment_confidence". It is important to note that only 70.5% of the sentiment values in this dataset have a confidence level of 100%, therefore any rows containing a sentiment confidence score less than 100% were removed before performing the SA.

3.3 BERT model

In this paper I will be using an improved RoBERTa model referred to as TimeLM-21 which uses the original BERT model (Devlin et al. 2018) but with a more substantial amount of training along with some tweaks to the hyperparameters. To train the model the authors used the Twitter Academic API to gather a large sample of tweets that are evenly dispersed across time. By querying the API using the most common stop-words they acquired a sample that is typical of general conversation on Twitter, in total 123.86M tweets were collected and trained upon.

The BERT model's architecture is based upon the original implementation of a transformer described by Vaswani et al. in 2017 and so this will be the architecture explained in this section.

The transformer in BERT uses stacked self-attention, point-wise and fully connected layers for the encoder and decoder alike. Figure 7 describes the architecture.

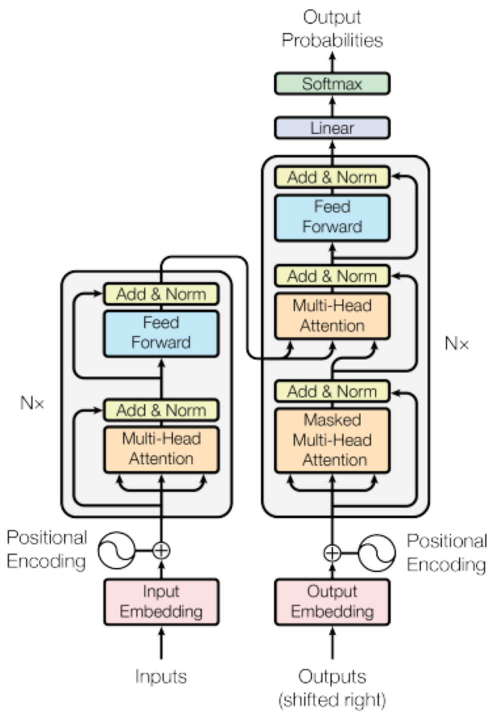

Figure 7 – BERT model architecture (Vaswani et al. 2017)

Learned dimension d_{model} embeddings are created from the source tokens and then modified by an additive positional encoding. The network lacks the ability to use the sequence's order naturally since it lacks recurrence or convolution, hence the positional encoding is required. The additive encoding used by the authors is defined in equations 1 and

2 where *pos* is the position of a word in the sentence and *i* is the dimension of the vector (Ahmed et al., 2017).

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}}) \tag{1}$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}}) \tag{2}$$

3.3.1 Encoder/Decoder

Each of the N = 6 identical layers that make up the encoder contains two sublayers. The first layer, as shown in the image, has a multi-head self-attention process, while the second layer is a position-wise fully connected feed-forward network. Around both two sub-layers, we use a residual connection (He et al., 2015), proceeded by layer normalisation. Each sublayer's output is LayerNorm(x + Sublayer(x)), where Sublayer(x) is the function that is executed by the sub-layer. All model sub-layers in addition to the embedding layers generate outputs with dimension d_{model} = 512 to allow these residual connections.

Similar to the encoder, the decoder also consists of N = 6 identical layers. The primary difference when compared with the encoder layer is that the decoder uses a third sub-layer that conducts multi-head attention over the encoder stack's output. Residual connections are then utilised around all the sub-layers which is then proceeded by layer normalisation.

In order to prevent positions from attending to following positions, the self-attention sub-layer within the decoder stack is adjusted. The predictions for position i can only depend on the known outputs at positions <i due to masking and the fact that the output embeddings are offset by one position.

3.3.2 Attention

The mapping of a query and a collection of key-value pairs to an output, where the query, keys, values, and output are all vectors, is the primary purpose of the attention function. The output is calculated as a weighted sum of the values, with each value's weight being determined by the query's compatibility function with its associated key. Figure 8 illustrates both the scaled dot-product attention (left) and the multi-head attention (right).

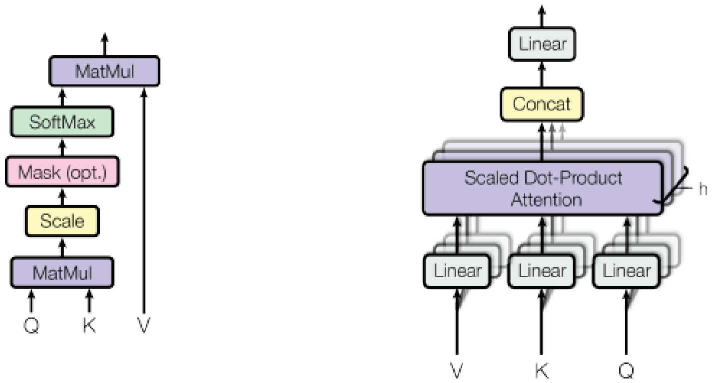

Figure 8 – Scaled dot-product attention (left), Multi-head attention (right)

Scaled dot-product attention, which works with a query Q, key K, and value V, is the foundation for a multi-head attention system. d_k represents the dimension of the key.

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

The inputs are concatenated in the first layer so that each of (Q, K, and V) equals the word vector matrix, with the exception of the scaling factor d_k, which enhances numerical stability, this is the same as dot-product attention. There are h distinct representations of (Q, K, V) which are obtained by means of multi-head attention mechanisms, which then calculate scaled dot-product attention for each representation, concatenate the results, and then project the concatenation with a feed-forward layer (Ahmed et al., 2017).

$$\begin{aligned} \mathrm{MultiHead}(Q, K, V) &= \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O \\ \text{where } \mathrm{head_i} &= \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{4}$$

Where the projections are parameter matrices:

$$W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$$

$$W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}.$$

In this model there is h = 8 parallel attention layers, for all of these we use $d_k = d_v = d_{model} / h = 64$.

A position-wise feed-forward network makes up the second part of each layer of the Transformer network. The authors of this model suggest utilising a two-layered network with a ReLU activation. Given trainable weights W_1, W_2, b_1, b_2, the sub-layer is defined as:

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{5}$$

In the authors experiments, the inner layer's dimension, d_{ff}, is set to 2048. For further discussion of the embeddings, softmax, and positional encoding part of the model I refer readers to the original paper “Attention Is All You Need” produced by Vaswani et al. in 2017.

3.4 Dynamic Bayesian Networks

Depending on the type of variables, in the majority of cases these nodes define either multinomial distributions or Gaussian distributions. This results in the three widely used models of discrete BNs, Gaussian BNs, and conditional linear Gaussian BNs, the latter of which is utilised for mixed types of variables (Quesada et al., 2022). In this paper I will be utilising Gaussian BNs.

A Bayesian Network (BN) is defined by a handful of features. The first being a network structure and directed acyclic graph (DAG), where each node $X_i \in \mathbf{X}$ corresponds to a random variable X_i . Secondly, a global probability distribution of X = (X_i , … , X_p) with corresponding parameters θ that in accordance with the graph's arcs $a_{ij} \in A$, can be factored into smaller local probability distributions. The primary function of the network structure is to describe the factorisation of the global distribution by graphically expressing the conditional independence relationships among the model's variables. The factorisation of the global distribution is defined in equation 6.

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_i|\mathbf{Pa}_i), \tag{6}$$

Assuming a Gaussian scenario where p(**X**) is a multivariate Gaussian distribution the local probability density of each node in the above equation is defined by a conditional probability distribution (CPD) given its parents, shown in equation 7.

$$p(x_i|\mathbf{Pa}_i) = \mathcal{N}\left(\beta_{0i} + \beta_{1i}x_{1(i)} + \dots + \beta_{ki}x_{k(i)};\ \sigma_i^2\right) = \mathcal{N}\left(\mu_i^{\mathbf{Pa}_i};\ \sigma_i^2\right). \tag{7}$$

β_{i0} , … , β_{k0} refers to the regression parameters corresponding to each parent node in $\mathbf{Pa}_i$, whereas σ_i^2 refers to the unconditional variance of X_i that is independent of its parents.

The joint probability distribution of this particular network can be defined as:

$$p(\mathbf{X}) = \prod_{i=1}^{n} \mathcal{N}\left(\mu_i^{\mathbf{Pa}_i};\ \sigma_i^2\right). \tag{8}$$

One major drawback of BNs is that they are only representative of variables at a given time point, they are unable to describe time series data where past variables may influence future variables. To overcome this problem, we can extend BNs to DBNs which are capable of taking time into account. This is done by creating multiple time slices, each containing the same variables. These time slices can have two types of arcs, one is referred to as intra-slice which only remain within the given time slice, and inter-slice arcs which can be connected to past and future time slices. The inter-slice arcs demonstrate the effects of the given variable on variables in the future version of the system.

When calculating the joint probability distribution, we take all the preceding time slices up to the point in time, T, we want to forecast. In the below equation, $\mathbf{X}^t = \{X_1^t, X_2^t, \dots, X_n^t\}$ refers to the set of nodes at time slice t.

$$p(\mathbf{X}^0, \dots, \mathbf{X}^T) \equiv p(\mathbf{X}^{0:T}) = p(\mathbf{X}^0) \prod_{t=0}^{T-1} p(\mathbf{X}^{t+1} | \mathbf{X}^{0:t}), \tag{9}$$

Working under the assumption that the past states of the system are only able to affect the present up to a certain order we can represent the joint probability distribution as the following, where m is the Markovian order of the given network (Quesada et al., 2022).

$$p(\mathbf{X}^{0:T}) = p(\mathbf{X}^0) \prod_{t=0}^{m-1} p(\mathbf{X}^{t+1} | \mathbf{X}^{0:t}) \prod_{t=m}^{T-1} p(\mathbf{X}^{t+1} | \mathbf{X}^{(t-m+1):t}), \tag{10}$$

In this study I will be working with the Markovian set to 1. This means that time slices are only able to affect the time slice immediately ahead. Using a Markovian order of 1 simplifies the equation to the following:

$$p(\mathbf{X}^{0:T}) = p(\mathbf{X}^0) \prod_{t=0}^{T-1} p(\mathbf{X}^{t+1} | \mathbf{X}^t). \tag{11}$$

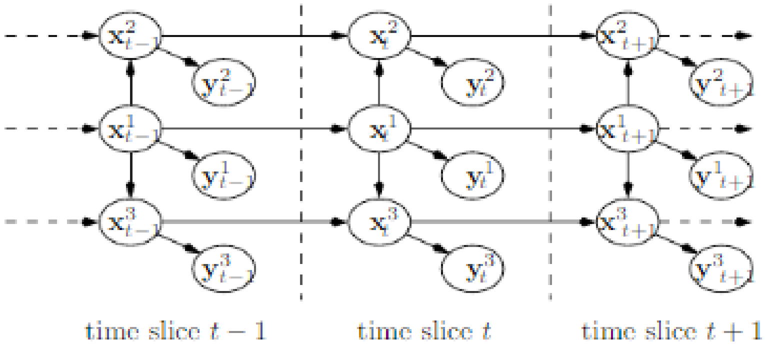

Figure 9 – Visual representation of a Dynamic Bayesian Network (Gonzales et al. 2011)

4 Experimental Setup

4.1 Data pre-processing

Before performing data pre-processing it is important to research how the authors of the TimeLM-21 model pre-processed their data. In Loureiro et al.'s paper the only data pre-processing that was performed was to simply replace any links or mentions with the placeholders "http" and "@user", respectively. Table 3 shows some examples of pre-processing.

Original Tweet	Pre-processed Tweet
One of the single most effective remedies to eradicate another round of #Trump Plague in our #WhiteHouse. https://t.co/QGB9ODIVS8	One of the single most effective remedies to eradicate another round of #Trump Plague in our #WhiteHouse. http
@cnnbrk #Trump owes #RicardoAguirre $730,000 to pay for the mass murder of his family. #TrumpLiedPeopleDied @FauciFan	@user #Trump owes #RicardoAguirre $730,000 to pay for the mass murder of his family. #TrumpLiedPeopleDied @user

Table 3 – Examples of data pre-processing

In order to optimise the sentiment analysis' performance, various methods of data pre-processing were tested . Table 4 displays the accuracies of the various data pre-processing techniques when performed on the validation dataset (US airlines dataset), it is important to note that all the techniques used involve the use of the "http" and "@user" placeholders. It was discovered that using only placeholders was the optimal method of pre-processing, this makes sense since it was the only form of pre-processing used in TimeLM-21's original implementation.

Pre-processing method (TimeLM-21)	Accuracy
Remove nothing	**84.73%**
Remove digits	84.55%
Remove hashtags	83.52%
Remove stop-words	83.29%
Remove punctuation	82.81%
Remove everything	77.80%

Table 4 – Data pre-processing test results for TimeLM-21

This process was then repeated to determine which data pre-processing method would be the most accurate for the CNN approach. It is important to note that for the CNN, both links and mentions were completely removed for each pre-processing method.

Pre-processing method (CNN)	Accuracy
Remove nothing	93.93%
Remove digits	93.20%
Remove hashtags	93.45%
Remove stop-words	91.40%
Remove punctuation	**94.94%**
Remove everything	91.01%

Table 5 – Data pre-processing test results for CNN

As expected, the accuracy rate is much better than the TimeLM-21 method due to the CNN being trained and tested on tweets containing a very specific vocabulary. Although the TimeLM-21 method is less accurate for this particular dataset, the fact that it is trained on a significantly wider range of vocabulary means that it should be able to generalise better when faced with differing topics.

4.2 Sentiment Analysis

Since the data I will be using has no true values for the sentiment of each tweet, it is necessary to validate the accuracy of the SA model before performing it on the primary dataset. Therefore, both the CNN and TimeLM-21 models will be tested using a popular sentiment analysis benchmark referred to as “TweetEval”, proposed by Barbieri et al. (2020). In this test the CNN attained an accuracy of 74.00% whereas TimeLM-21 attained an accuracy of 76.76%, this supports the hypothesis that the TimeLM-21 model is more capable at generalising to broader vocabularies.

One beneficial feature twitter uses is the hashtag system, this allows us to determine the topic of a particular tweet by reading the hashtags used in the tweet. Using this to our advantage, we can sum together how many times each hashtag occurs within the entire dataset and display the most commonly occurring. When selecting which topics to analyse I prioritised the most commonly occurring hashtags which also tend to create semi-controversial discussion, some examples include #Trump or #COVID19. It is important to note that many hashtags have different spellings but refer to the exact same topic, for example: #Trump and #DonaldTrump, or #COVID19 and #coronavirus. These hashtags will be grouped together and treated as the same topic. Table 6 shows the topics selected to be analysed with their respective hashtags and sum of tweets associated with the given topic.

Topic	Associated Hashtags	Total Tweets
Donald Trump	#Trump, #trump, #DonaldTrump, #donaldtrump	1208433
Joe Biden	#Biden, #biden, #JoeBiden, #joebiden	1022419
2020 Election	#Elections2020, #USAElections2020, #USElections2020	149592
COVID-19	#COVID19, #covid19, #coronavirus, #Coronavirus, #COVID	68100
MAGA (Make America Great Again)	#MAGA, #maga, #MAGA2020	62534
Kamala Harris	#KamalaHarris, #Harris, #harris	54579
Trump Meltdown	#TrumpMeltdown, #trumpmeltdown	23858
Democrats	#Democrats, #democrats	23221
GOP (synonym for Republican party)	#GOP, #GOP2020, #gop	20762
Pennsylvania	#Pennsylvania, #pennsylvania	17472
CNN (US news channel)	#CNN, #cnn	16173
Barrack Obama	#Obama, #obama	15525
Trump Is Losing	#TrumpIsLosing, #trumpislosing	14729
Republicans	#Republicans, #republicans	13814
Michigan	#Michigan, #michigan	13596
China	#China, #china	13515
Vote Him Out	#VoteHimOut, #votehimout	13018
Florida	#Florida, #florida	12673
Georgia	#Georgia, #georgia	12567
Fox News (US news channel)	#FoxNews, #foxnews	12364
Mike Pence	#Pence, #pence	5610

Table 6 – Selected topics for sentiment analysis

Once the topics have been selected the tweets belonging to each topic and placed in one of twenty-five time bins based on the date the tweet was created. Each time bin represents a 24h period starting from 15/10/2020 and ending at 08/11/2020. The purpose of this is to allow us to analyse the sentiment for each topic on each given day which in turn enables us to assess sentiment evolution over the given time period. Due to the large number of tweets and computationally intensive model used for calculating the sentiment we limit each topic to approximately 10000 tweets with exception for the Mike Pence topic.

The next step is to calculate a normalised single value that represents the net sentiment for each given day for each topic; this is referred to as the sentiment score. This is calculated using equation 12 where S_{score} , S_{pos} , and S_{neg} represent sentiment score, total positive tweets, and total negative tweets, respectively. A sentiment score close to 1 represents a net majority of positive tweets, a sentiment score close to -1 represents a net majority of negative tweets, and a sentiment score close to 0 represents a somewhat balanced number of positive and negative tweets.

$$S_{score} = \frac{S_{pos}-S_{neg}}{S_{pos}+S_{neut}} \tag{12}$$

Once the sentiment scores have been calculated for each day and topic, they are then aggregated into a single data frame; this will be included in the appendix. The aggregated sentiment score data for each topic is then saved to a CSV file where it is later opened in R to create a DBN.

This entire process is then repeated using the CNN model trained on the US airlines dataset to acquire a second data frame containing the sentiment score data which will then also be used to create a second DBN, these will then be compared. Figure 10 shows a summary of the CNN used.

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_5 (Embedding)     (None, 35, 100)           823600

 conv1d_5 (Conv1D)           (None, 28, 32)            25632

 max_pooling1d_5 (MaxPooling  (None, 14, 32)           0
 1D)

 flatten_5 (Flatten)         (None, 448)               0

 dense_10 (Dense)            (None, 10)                4490

 dense_11 (Dense)            (None, 1)                 11

=================================================================
Total params: 853,733
Trainable params: 853,733
Non-trainable params: 0
_________________________________________________________________
..
```

Figure 10 – Summary of CNN architecture

4.3 Dynamic Bayesian Network

The DBN is created using the R package “dbnR” developed by David Quesada who published two papers demonstrating the package’s capabilities. The first step is to decide our Markovian order, in this paper the Markovian order will be set to 1. The next step is to learn the structure, this was done using the dynamic max-min hill climbing (DMMHC) algorithm developed by Trabelsi (2013). DMMHC is an extension of the max-min hill climbing algorithm (Tsamardinos et al. 2006) which is a hybrid method that searches the space of possible structures with a local search and then directs the arcs and scores the resulting networks with an evaluation criterion (Quesada et al. 2021). In the dbnR package there are two alternative algorithms for learning the network structure, these work by leveraging particle swarm optimisation. In Quesada et al. (2021) they found that the particle swarm approaches were more specialised for high-order networks, and both suffered slightly at small networks. Based on this information along with the DMMHC’s faster execution time I decided this would be the optimal algorithm.

The next step is to fold the data table. The dataset is widened by including the t previous time slices in each row so that the second stage of the DMMHC can utilise it to learn temporal arcs. Once this is done, the DBN parameters are fitted using maximum likelihood estimation. The DBN can now be visually plotted. After both DBNs have been produced using the CNN sentiment score data and the TimeLM-21 sentiment score data we then calculate and compare the Bayesian information criterion (BIC) value to assess which model is more suitable.

5 Results

5.1 Exploratory Data Analysis

As previously mentioned, the dataset consists of 1748235 tweets with 971157 of those including a Trump related hashtag and 777078 containing a Biden related hashtag. The average tweet contained 23 words, 153 characters, and 4.2 hashtags. In total there were 285380 unique hashtags used, most of these are likely irrelevant hashtags with very little use or misspelt hashtags. Figure 11 shows what percentage of the total number of hashtags used are accounted for by hashtags with less than X usages. For example, hashtags with only 3 or less uses account for 78.3% of the total unique hashtags.

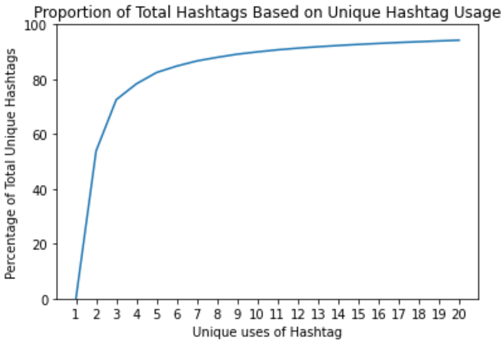

Figure 11 – Graph showing frequency of low use hashtags

When deciding which hashtags, or topics, to analyse we must first aggregate them all to find the most commonly used. Figure 12 displays a word cloud containing the top 50 most used hashtags, this graph clearly demonstrates the frequency of similar hashtags.

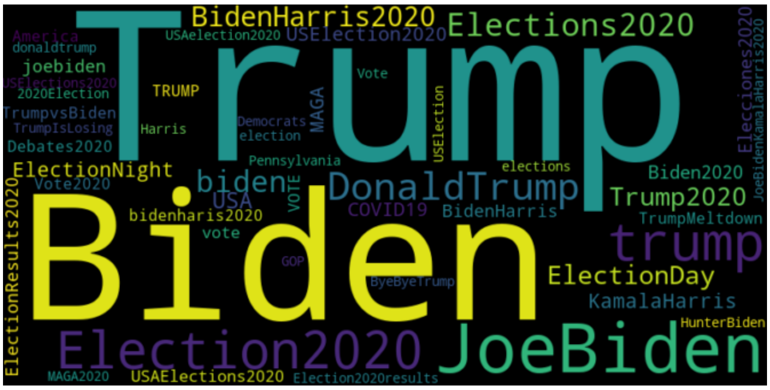

Figure 12 – Word cloud of top 50 most used hashtags

When looking at the number of tweets posted per day it clear to see in figure 13 that there are large spikes on the 4th, 23rd, and 7th of November. This is likely due to the final presidential debate (22nd Oct), election day (3rd Nov), and Trump publicly claiming voter fraud via Twitter (6th Nov).

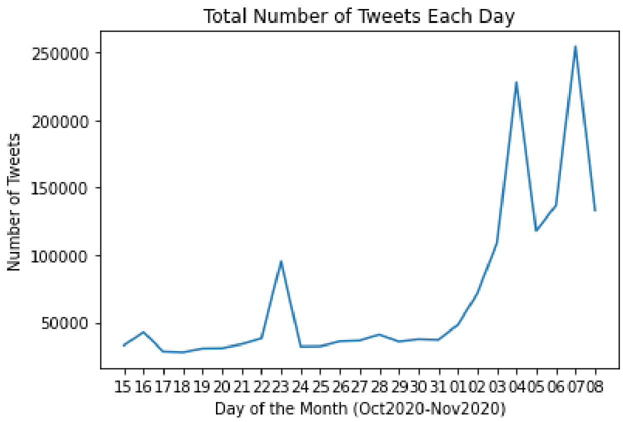

Figure 13 – Number of tweets per day

5.2 Sentiment Analysis

Figure 14 shows the data acquired from performing SA on Trump and Biden related tweets where 0, 1, 2 represent number of negative, neutral, and positive tweets respectively.

	15	16	17	18	19	20	21	22	23	24	...	30	31	01	02	03	04	05	06	07	08
0	229	193	228	202	216	204	211	188	160	177	...	168	196	167	138	89	79	129	150	133	149
1	149	161	141	163	159	155	155	174	191	180	...	190	158	188	203	247	280	243	230	225	218
2	22	46	31	35	25	41	34	38	49	43	...	42	46	45	59	64	41	28	20	42	33

	15	16	17	18	19	20	21	22	23	24	...	30	31	01	02	03	04	05	06	07	08
0	175	147	170	168	146	149	148	138	163	148	...	150	144	130	103	63	53	84	76	61	63
1	183	172	186	181	207	204	193	211	192	203	...	187	198	208	206	261	296	267	257	201	235
2	42	81	44	51	47	47	59	51	45	49	...	63	58	62	91	76	51	49	67	138	102

Figure 14 – Number of negative, neutral, and positive tweets sorted by day for #Trump (top) and #Biden (bottom)

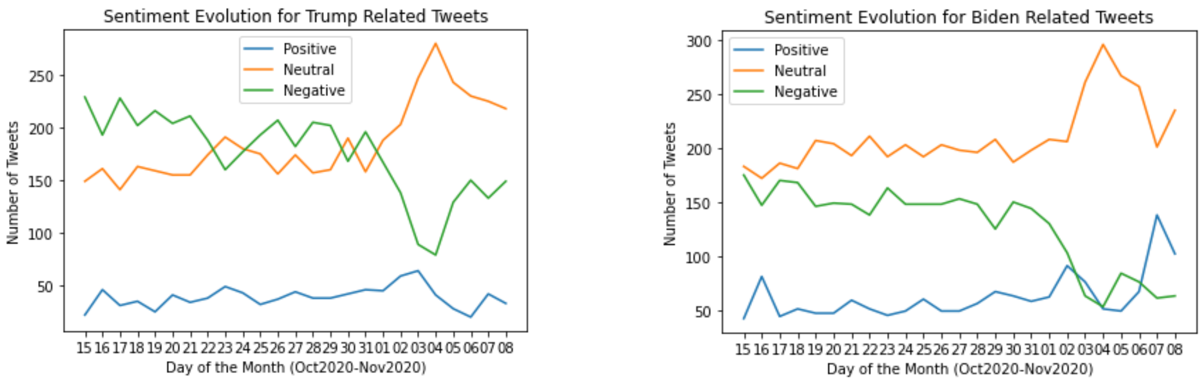

Figure 15 – Number of negative, neutral, and positive tweets per day

When looking at the plots, it is clear to see the immediate effect of election day on the 3rd of November. The large increase of neutral tweets is perhaps explained by high numbers of people simply "spreading the word" of who is winning the election rather than making overtly positive or negative tweets. We then notice an increase in positive tweets for Biden following the election day, presumably due to people celebrating his projected victory.

Table 7 shows some examples of aggregated sentiment score data which has been plotted on a graph of Trump sentiment over time. The full table of sentiment scores will be included in the appendix.

Date	Hashtags		
	Trump	Elections2020	TrumpIsLosing
15/10	-0.5175	-0.193966	-0.921569
16/10	-0.3675	-0.202500	-0.953488
17/10	-0.4925	-0.184066	-0.923077
18/10	-0.4175	-0.166667	-0.911765
19/10	-0.4775	-0.187500	-0.888889
20/10	-0.4075	-0.057500	-1.000000
21/10	-0.4425	-0.107500	-0.906250
22/10	-0.3750	-0.167500	-0.885714
23/10	-0.2775	-0.112500	-0.875000
24/10	-0.3350	-0.172500	-1.000000
25/10	-0.4025	-0.175000	-0.811644
26/10	-0.4250	-0.175000	-0.847500
27/10	-0.3450	-0.122500	-0.797500
28/10	-0.4175	-0.187500	-0.780000
29/10	-0.4100	-0.132500	-0.752500
30/10	-0.3150	-0.112500	-0.776471
31/10	-0.3750	-0.087500	-0.764398
01/11	-0.3050	-0.105000	-0.905830
02/11	-0.1975	-0.017500	-0.813880
03/11	-0.0625	-0.010000	-0.652500
04/11	-0.0950	0.025000	-0.668539
05/11	-0.2525	-0.075000	-0.707500
06/11	-0.3250	-0.077500	-0.700000
07/11	-0.2275	0.045000	-0.627500
08/11	-0.2900	-0.022500	-0.620000

Table 7 – Aggregated sentiment data for #Trump, #Elections2020, and #TrumpIsLosing

Most sentiment scores amongst the topics are negative, this could be due to various subsets of people using each hashtag. As expected, for topics that are less controversial such as #Elections2020 we notice a sentiment score much closer to 0, suggesting either a balanced number of positive/negative tweets or a large majority of neutral tweets. As for #TrumpIsLosing we find a large majority of sentiment scores close to -1 which suggests that this hashtag is used almost exclusively for negative tweets.

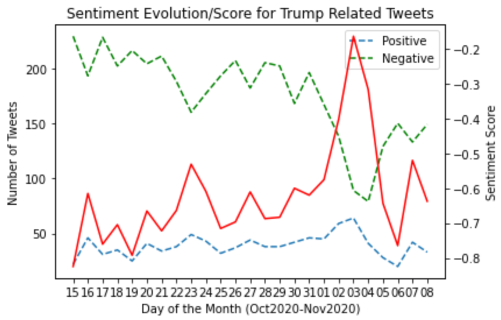

Figure 16 – Sentiment score in relation to positive/negative tweets

When plotted on a sentiment evolution graph the sentiment score curve simply represents the difference between positive and negative tweets. This is most noticeable on the 4th of November when the green and blue curve are closest together which causes the sentiment score to spike up, representing a more balanced sentiment.

5.3 Dynamic Bayesian Network

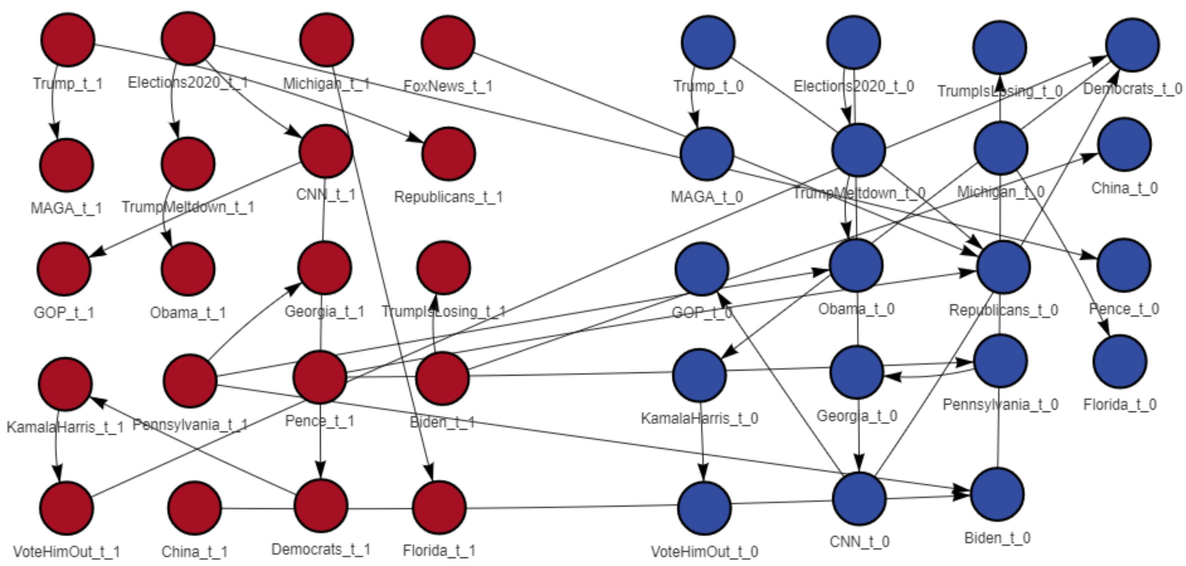

Figure 17 – Dynamic Bayesian Network produced via TimeLM-21 (two time slices)

Figure 17 shows the DBN produced via the TimeLM-21 sentiment data (of Markovian order 1) where each arrow (arc) tells us what topics are influenced by other topics. For example, we notice an intra-slice arc linking the node Pennsylvania to Georgia, this tells us that user sentiment for #Georgia is influenced by user sentiment for #Pennsylvania. Another example is the inter-slice arc linking #FoxNews in time slice 0 to #Republicans in time slice 1, this suggests that user sentiment for #FoxNews influences user sentiment for #Republicans the following day. It is worth noting that nodes with no connections were removed, these include COVID_t_1, FoxNews_t_0, and COVID_t_0. The reason for this is likely due to there not being enough data for the DBN to learn from, collecting more data could potentially provide more connections between nodes.

When looking more closely we notice many clear interpretations represented in the DBN. One example is sentiment for #Trump influencing the sentiment for both #MAGA and #Republicans. This makes sense since #MAGA (Make America Great Again) is used almost exclusively to support Trump, therefore one would expect a rise in positive sentiment for #Trump to also cause a rise in positive sentiment for #MAGA. Due to Trump being the leader of the Republican party one could hypothesise the same reason for these two topics being linked although due to #Republicans taking the plural form it may be used to negatively comment on people belonging to the republican party.

A further clear interpretation is the arc between #Biden and #TrumpIsLosing. Since #TrumpIsLosing is used almost exclusively amongst Biden supporters it makes sense that the sentiment for #Biden influences the sentiment for #TrumpIsLosing. A very similar example can also be seen between #KamalaHarris (Biden's running mate) and #VoteHimOut (negatively referring to Trump).

5.4 Comparison to CNN-produced DBN

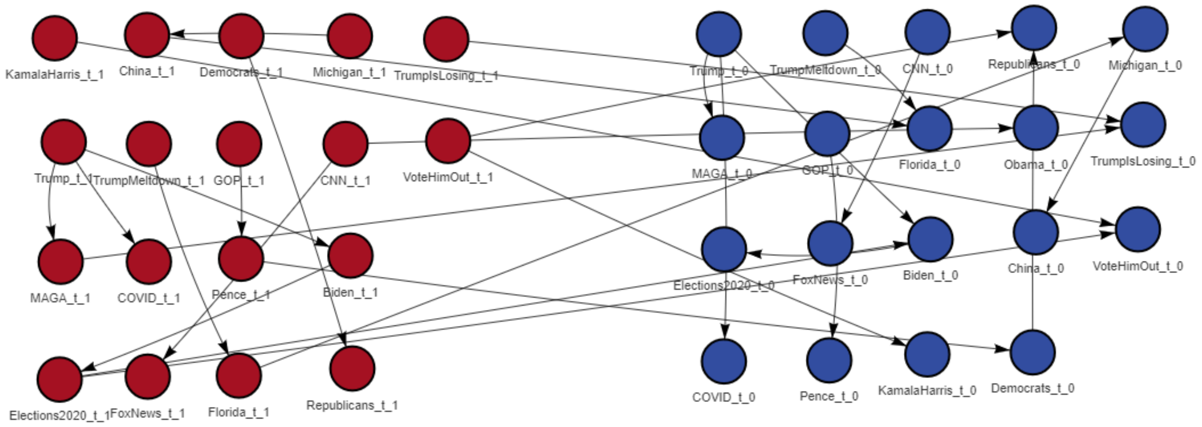

Figure 18 – Dynamic Bayesian Network produced via CNN (two time slices)

Figure 18 shows the DBN produced via the sentiment data acquired from the CNN, the omitted nodes with no connections are Pennsylvania_t_1, Obama_t_1, Georgia_t_1, Pennsylvania_t_0, and Georgia_t_0. Despite there being many clearly interpretable links between topics the AIC and BIC values calculated both suggest that the DBN produced via TimeLM-21 is the more suitable model. Although the TimeLM-21 approach produced a more suitable DBN it does come at a cost of computational resources, when analysing 1000 tweets, on average, TimeLM-21 executed in 116s whereas the CNN executed in only 0.12s. Table 8 displays the results.

Model	AIC	BIC	Runtime per 1000 tweets (for SA)
TimeLM-21	711.70	642.78	116s
CNN	989.19	922.63	0.12s

Table 8 – AIC, BIC, and time to execute metrics

6 Conclusion

In this paper I have developed an improved method of analysing how sentiments towards various topics can influence each other over a period of time. This was done by utilising a previously proposed model called TimeLM-21 to acquire a time series of sentiment data for various sub-topics related to the US 2020 elections which was then used to learn a DBN. This process was then repeated using a CNN instead of the TimeLM-21 model to produce different time series sentiment data. These two sets of data were then used to produce DBNs. These DBNs then allows us to visualise how the sentiment of sub-topics can influence the sentiment of other sub-topics over time. After comparison using both AIC and BIC values it was discovered that the DBN produced via TimeLM-21 was a more suitable model albeit a much computationally intensive method.

6.1 Future Work

Despite this work producing the expected results, there are some limitations that could be improved. Due to the scarcity of large datasets of tweets containing observed values for sentiment, it was impossible to evaluate the accuracy of the sentiment analysis for the US 2020 election dataset and instead I had to settle for evaluating the method using a much smaller dataset. Future work could instead collect their own, more suitable, dataset containing observed values or instead find a similar dataset produced by a different author to evaluate the sentiment analysis performed before creating a DBN. Furthermore, due to limited computational resources I was only able to analyse around 10000 tweets per topic which limits the data necessary for learning a more reliable DBN.

6.2 Social, Legal, Ethical and Professional Considerations

This project made use for two datasets containing tweets published by thousands of users around the world. The primary issues here are caused by tweets having their creator's username and location attached to it, to solve this they were simply removed before performing any pre-processing or analysis on the data.

Throughout this project any works from other authors are clearly cited and referenced.

7 Achievements

When regarding the primary objective, “Can a modern sentiment analysis approach be used to produce a DBN analysing US 2020 election sub-topics?”, it is clear to see that by using TimeLM-21 I was able to produce a DBN capable of displayed numerous clear to interpret results. Despite this method providing weaker results when analysing the US airlines dataset (which was expected), it exceeded the CNN model at analysing the US 2020 elections dataset.

When regarding the second objective, “Does this modern approach provide a more suitable DBN than previous works?”, we can see that the DBN produced via the TimeLM-21 approach was more suitable as suggested by the calculated AIC and BIC values. Admittedly this did come at a cost of increased computational resources.

7.1 Student Reflections

Completing a MSc research project is a lengthy process that requires time-management skills and organisation. Being the longest piece of work I have produced it has allowed me to significantly develop these skills. Furthermore, producing a piece of work to this level requires research into many related works which has further helped me to develop my skills in research.

When reflecting on my time producing this work, one thing I would alter is how much research I do into different coding languages. Since Python is my strongest coding language, I tended to tunnel vision into only coding using this language, this is problematic since often times superior packages/libraries are only available in alternate coding languages.

Overall, I am satisfied with the sentiment analysis and DBNs produced, even improving upon a previously developed method. Despite this, there were weaknesses in my work, for example the number of tweets analysed, due to my limited computational resources.

8 Bibliography and References

Tankovska, H. (2019). Social networks: penetration in selected countries 2019 | Statistic. Statista; Statista. https://www.statista.com/statistics/282846/regular-social-networking-usage-penetration-worldwide-by-country/

Quesada, D., Bielza, C., Fontán, P., & Larrañaga, P. (2022). Piecewise forecasting of nonlinear time series with model tree dynamic Bayesian networks. International Journal of Intelligent Systems, 37(11), 9108–9137. https://doi.org/10.1002/int.22982

Sayce, D. (2010, March 3). Number of tweets per day? | David Sayce. David Sayce. https://www.dsayce.com/social-media/tweets-day/

Liang, H., Ganeshbabu, U., & Thorne, T. (2020). A Dynamic Bayesian Network Approach for Analysing Topic-Sentiment Evolution. IEEE Access, 8, 54164–54174. https://doi.org/10.1109/access.2020.2979012

Lazaridou, A., Kuncoro, A., Gribovskaya, E., Agrawal, D., Liska, A., Terzi, T., Gimenez, M., d'Autume, C. de M., Ruder, S., Yogatama, D., Cao, K., Kocisky, T., Young, S., & Blunsom, P. (2021). Pitfalls of Static Language Modelling. Www.arxiv-Vanity.com. https://www.arxiv-vanity.com/papers/2102.01951/

Barbieri, F., Camacho-Collados, J., Espinosa Anke, L., & Neves, L. (2020, November 1). TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification. ACLWeb; Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.148

Trifonova, N., Karnauskas, M., & Kelble, C. (2019). Predicting ecosystem components in the Gulf of Mexico and their responses to climate variability with a dynamic Bayesian network model. PLOS ONE, 14(1), e0209257. https://doi.org/10.1371/journal.pone.0209257

Duan, T. (2016). Auto Regressive Dynamic Bayesian Network and Its Application in Stock Market Inference. (n.d.). Springerprofessional.de. Retrieved November 16, 2022, from https://www.springerprofessional.de/en/auto-regressive-dynamic-bayesian-network-and-its-application-in-/10651456

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1–135. https://www.cs.cornell.edu/home/llee/omsa/omsa.pdf

Wongkar, M., & Angdresey, A. (2019, October 1). Sentiment Analysis Using Naive Bayes Algorithm Of The Data Crawler: Twitter. IEEE Xplore. https://doi.org/10.1109/ICIC47613.2019.8985884

Gao, J., Yao, R., Lai, H., & Chang, T.-C. (2019, May 1). Sentiment Analysis with CNNs Built on LSTM on Tourists Comments. IEEE Xplore. https://doi.org/10.1109/ECBIOS.2019.8807844

Li, D., & Qian, J. (2016, October 1). Text sentiment analysis based on long short-term memory. IEEE Xplore. https://doi.org/10.1109/CCI.2016.7778967

Wang, W., Yan, M., & Wu, C. (2018, July 1). Multi-Granularity Hierarchical Attention Fusion Networks for Reading Comprehension and Question Answering. ACLWeb; Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-1158

Sinha, S., Jayan, A., & Kumar, R. (2022, May 1). An Analysis and Comparison Of Deep-Learning Techniques and Hybrid Model for Sentiment Analysis for Movie Review. IEEE Xplore. https://doi.org/10.1109/INCET54531.2022.9824630

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. ArXiv.org. https://arxiv.org/abs/1408.5882

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Brain, G., Research, G., Jones, L., Gomez, A., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. https://arxiv.org/pdf/1706.03762.pdf

He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). Deep Residual Learning for Image Recognition. ArXiv.org. https://arxiv.org/abs/1512.03385

Ahmed, K., Keskar, N., & Socher, R. (2017). Weighted Transformer Network for Machine Translation. Undefined. https://www.semanticscholar.org/paper/Weighted-Transformer-Network-for-Machine-Ahmed-Keskar/3861ae2a6bdd2a759c2d901a6583e63a216bc2fc

Gonzales, C., Dubuisson, S., & N'Guyen, X. S. (2011). Simultaneous Partitioned Sampling for Articulated Object Tracking. Advanced Concepts for Intelligent Vision Systems, 150–161. https://doi.org/10.1007/978-3-642-23687-7_14

Loureiro, D., Barbieri, F., Neves, L., Espinosa, L., & Camacho-Collados, J. (n.d.). TimeLMs: Diachronic Language Models from Twitter. https://arxiv.org/pdf/2202.03829.pdf

Ghada Trabelsi. (2013) New structure learning algorithms and evaluation methods for large dynamic Bayesian networks. Machine Learning [cs.LG]. Université de Nantes; Ecole Nationale d'Ingénieurs de Sfax, 2013.English. ffNNT : ff. fftel-00996061f

Quesada, D., Valverde, G., Larrañaga, P., & Bielza, C. (2021). Long-term forecasting of multivariate time series in industrial furnaces with dynamic Gaussian Bayesian networks. Engineering Applications of Artificial Intelligence, 103, 104301. https://doi.org/10.1016/j.engappai.2021.104301

Tsamardinos, I., Brown, L. E., & Aliferis, C. F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, 65(1), 31–78. https://doi.org/10.1007/s10994-006-6889-7

CNN. (2020). 2020 presidential election results. Www.cnn.com. https://edition.cnn.com/election/2020/results/president

10 Appendix

Sentiment data from TimeLM-21

	Trump	Biden	Elections2020	KamalaHarris	MAGA	COVID	TrumpMeltdown	Democrats	GOP	Pennsylvania	...	Michigan	VoteHimOut	CNN
0	-0.5175	-0.3325	-0.193966	-0.2150	-0.3200	-0.4125	-0.809524	-0.4650	-0.6125	-0.458824	...	-0.625000	-0.636735	-0.442500
1	-0.3675	-0.1650	-0.202500	-0.1850	-0.2625	-0.4675	-0.650794	-0.4150	-0.6100	-0.365385	...	-0.197802	-0.692029	-0.401786
2	-0.4925	-0.3150	-0.184066	-0.1275	-0.2075	-0.4175	-0.840000	-0.4725	-0.6250	-0.377551	...	-0.256684	-0.666667	-0.417614
3	-0.4175	-0.2925	-0.166667	-0.1950	-0.1425	-0.4575	-0.852941	-0.4675	-0.5175	-0.438596	...	-0.272989	-0.622047	-0.381381
4	-0.4775	-0.2475	-0.187500	-0.1150	-0.2275	-0.5175	-0.854839	-0.4525	-0.6550	-0.300813	...	-0.381295	-0.587500	-0.459302
5	-0.4075	-0.2550	-0.057500	-0.0375	-0.2375	-0.4850	-0.671875	-0.4650	-0.5675	-0.203390	...	-0.342593	-0.635000	-0.380000
6	-0.4425	-0.2225	-0.107500	-0.1000	-0.1675	-0.4425	-0.671233	-0.3425	-0.6600	-0.280130	...	-0.310559	-0.642500	-0.382500
7	-0.3750	-0.2175	-0.167500	-0.2325	-0.1550	-0.4575	-0.740000	-0.4175	-0.6275	-0.383117	...	-0.357143	-0.672500	-0.357500
8	-0.2775	-0.2950	-0.112500	-0.1425	-0.1700	-0.4375	-0.642500	-0.3550	-0.6025	-0.322500	...	-0.345029	-0.635000	-0.360000
9	-0.3350	-0.2475	-0.172500	-0.0800	-0.1850	-0.5450	-0.711599	-0.3875	-0.3700	-0.236760	...	-0.247788	-0.642254	-0.305000
10	-0.4025	-0.2200	-0.175000	-0.1625	-0.1925	-0.5650	-0.702381	-0.3625	-0.4775	-0.352727	...	-0.250000	-0.731629	-0.446746
11	-0.4250	-0.2475	-0.175000	-0.3725	-0.1850	-0.6075	-0.756757	-0.3925	-0.5700	-0.185000	...	-0.197605	-0.726471	-0.400000
12	-0.3450	-0.2600	-0.122500	-0.2175	-0.0375	-0.5600	-0.500000	-0.3975	-0.5325	-0.294737	...	-0.281022	-0.749196	-0.325269
13	-0.4175	-0.2300	-0.187500	-0.1950	-0.2500	-0.6150	-0.756684	-0.4325	-0.6725	-0.322034	...	-0.255452	-0.647500	-0.490000
14	-0.4100	-0.1450	-0.132500	-0.2425	-0.1425	-0.5725	-0.577889	-0.4250	-0.5750	-0.272340	...	-0.227848	-0.635000	-0.395000
15	-0.3150	-0.2175	-0.112500	-0.2800	-0.0925	-0.5350	-0.711111	-0.4300	-0.6475	-0.342629	...	-0.178832	-0.617500	-0.400000
16	-0.3750	-0.2150	-0.087500	-0.1700	-0.1775	-0.5525	-0.719231	-0.4000	-0.4925	-0.220000	...	-0.216901	-0.537500	-0.407500
17	-0.3050	-0.1700	-0.105000	-0.0700	-0.0050	-0.5650	-0.583916	-0.4250	-0.6150	-0.205000	...	-0.190476	-0.635000	-0.367500
18	-0.1975	-0.0300	-0.017500	-0.1450	-0.0200	-0.4400	-0.587500	-0.3525	-0.5850	-0.215000	...	-0.245000	-0.627500	-0.317500
19	-0.0625	0.0325	-0.010000	0.1550	0.2475	-0.3275	-0.507082	-0.2500	-0.4975	-0.082500	...	0.085000	-0.322500	-0.210000
20	-0.0950	-0.0050	0.025000	0.0425	0.0650	-0.3450	-0.442500	-0.2500	-0.4600	-0.022500	...	0.065000	-0.327500	-0.075000
21	-0.2525	-0.0875	-0.075000	-0.1475	-0.1575	-0.3825	-0.527500	-0.3900	-0.5700	-0.080000	...	-0.212500	-0.563981	-0.312500
22	-0.3250	-0.0225	-0.077500	0.0100	-0.2550	-0.3250	-0.477500	-0.4125	-0.5975	0.032500	...	-0.315000	-0.367424	-0.345000
23	-0.2275	0.1925	0.045000	0.3875	-0.1675	-0.3000	-0.445000	-0.2125	-0.4825	0.105000	...	-0.237500	-0.355670	-0.105000
24	-0.2900	0.0975	-0.022500	0.2675	-0.1725	-0.2725	-0.530000	-0.1500	-0.5375	-0.268156	...	-0.380117	-0.572816	-0.230000

Republicans	Florida	Georgia	China	FoxNews	Pence	TrumpIsLosing
-0.536585	-0.337278	-0.319149	-0.427500	-0.514563	-0.341176	-0.921569
-0.500000	-0.254658	-0.195122	-0.395000	-0.550820	-0.339713	-0.953488
-0.510204	-0.325490	-0.516129	-0.428934	-0.305221	-0.260563	-0.923077
-0.397222	-0.284946	-0.448718	-0.455041	-0.474510	-0.321839	-0.911765
-0.507500	-0.177536	-0.388889	-0.382500	-0.481203	-0.228723	-0.888889
-0.517500	-0.194175	-0.509804	-0.384840	-0.520147	-0.193548	-1.000000
-0.392500	-0.254167	-0.116279	-0.412500	-0.394737	-0.175000	-0.906250
-0.462725	-0.377358	-0.236842	-0.405000	-0.506250	-0.227053	-0.885714
-0.412500	-0.253602	-0.280000	-0.375000	-0.377500	-0.240664	-0.875000
-0.423780	-0.177945	0.018868	-0.357500	-0.352273	-0.240260	-1.000000
-0.390052	-0.255319	-0.341463	-0.422500	-0.404959	-0.522255	-0.811644
-0.480000	-0.131868	-0.084211	-0.477500	-0.423676	-0.540761	-0.847500
-0.525000	-0.350820	-0.197674	-0.407311	-0.404332	-0.376812	-0.797500
-0.424165	-0.180108	-0.105590	-0.457500	-0.467500	-0.428571	-0.780000
-0.380000	-0.155000	-0.510989	-0.317500	-0.532500	-0.401130	-0.752500
-0.462500	-0.197500	-0.239316	-0.366234	-0.453757	-0.300000	-0.776471
-0.502500	-0.245000	-0.257143	-0.429412	-0.419142	-0.356808	-0.764398
-0.532500	-0.252500	-0.258621	-0.307692	-0.435000	-0.252874	-0.905830
-0.367500	-0.312500	-0.337121	-0.347500	-0.402500	-0.237705	-0.813880
-0.207500	0.010000	-0.179775	-0.235000	-0.220000	-0.026163	-0.652500
-0.202500	-0.037500	0.020000	-0.295000	-0.225000	-0.131915	-0.668539
-0.225000	-0.212418	-0.027500	-0.270000	-0.340000	-0.093960	-0.707500
-0.500000	-0.212121	-0.002500	-0.325000	-0.390000	-0.224806	-0.700000
-0.345000	-0.228395	-0.065000	-0.305000	-0.272500	-0.252280	-0.627500
-0.327500	-0.191304	-0.248042	-0.075000	-0.307500	-0.264957	-0.620000

Sentiment data from CNN

	Trump	Biden	Elections2020	KamalaHarris	MAGA	COVID	TrumpMeltdown	Democrats	GOP	Pennsylvania	...	Michigan	VoteHimOut	CNN
0	-0.560	-0.500	-0.482759	-0.575	-0.535	-0.565	-0.904762	-0.760	-0.580	-0.858824	...	-0.525000	-0.526531	-0.645000
1	-0.630	-0.525	-0.625000	-0.510	-0.470	-0.675	-0.460317	-0.640	-0.625	-0.692308	...	-0.604396	-0.507246	-0.535714
2	-0.580	-0.600	-0.587912	-0.505	-0.505	-0.710	-0.280000	-0.615	-0.685	-0.714286	...	-0.689840	-0.575758	-0.488636
3	-0.545	-0.540	-0.447917	-0.525	-0.400	-0.710	-0.352941	-0.650	-0.475	-0.614035	...	-0.201149	-0.590551	-0.507508
4	-0.645	-0.520	-0.545000	-0.545	-0.480	-0.635	-0.225806	-0.710	-0.705	-0.707317	...	-0.482014	-0.480000	-0.668605
5	-0.585	-0.570	-0.615000	-0.435	-0.575	-0.645	-0.468750	-0.725	-0.715	-0.581921	...	-0.555556	-0.495000	-0.485000
6	-0.530	-0.540	-0.540000	-0.485	-0.405	-0.520	-0.506849	-0.730	-0.570	-0.615635	...	-0.751553	-0.570000	-0.345000
7	-0.600	-0.565	-0.505000	-0.670	-0.425	-0.625	-0.505000	-0.690	-0.730	-0.753247	...	-0.857143	-0.455000	-0.545000
8	-0.435	-0.490	-0.545000	-0.495	-0.410	-0.595	-0.315000	-0.555	-0.625	-0.525000	...	-0.672515	-0.570000	-0.530000
9	-0.560	-0.465	-0.500000	-0.470	-0.450	-0.625	-0.479624	-0.575	-0.815	-0.482866	...	-0.663717	-0.538028	-0.225000
10	-0.545	-0.420	-0.545000	-0.515	-0.425	-0.600	-0.476190	-0.530	-0.650	-0.643636	...	-0.562500	-0.654952	-0.639053
11	-0.470	-0.530	-0.490000	-0.595	-0.515	-0.590	-0.281853	-0.640	-0.765	-0.645000	...	-0.604790	-0.588235	-0.697143
12	-0.550	-0.430	-0.510000	-0.555	-0.460	-0.705	-0.500000	-0.655	-0.665	-0.715789	...	-0.722628	-0.511254	-0.618280
13	-0.570	-0.440	-0.525000	-0.605	-0.460	-0.730	-0.481283	-0.665	-0.805	-0.508475	...	-0.582555	-0.465000	-0.590000
14	-0.565	-0.460	-0.430000	-0.530	-0.465	-0.680	-0.527638	-0.560	-0.625	-0.523404	...	-0.544304	-0.535000	-0.520000
15	-0.565	-0.590	-0.480000	-0.580	-0.440	-0.655	-0.140741	-0.650	-0.760	-0.697211	...	-0.569343	-0.530000	-0.545000
16	-0.595	-0.445	-0.475000	-0.475	-0.460	-0.750	-0.415385	-0.745	-0.815	-0.530000	...	-0.566197	-0.530000	-0.460000
17	-0.400	-0.420	-0.505000	-0.400	-0.325	-0.580	-0.391608	-0.715	-0.745	-0.645000	...	-0.528822	-0.505000	-0.605000
18	-0.440	-0.430	-0.355000	-0.495	-0.315	-0.525	-0.330000	-0.735	-0.720	-0.620000	...	-0.655000	-0.315000	-0.490000
19	-0.405	-0.380	-0.145000	-0.280	-0.230	-0.550	-0.365439	-0.545	-0.580	-0.640000	...	-0.625000	-0.485000	-0.520000
20	-0.275	-0.200	-0.110000	-0.405	-0.290	-0.460	-0.345000	-0.680	-0.680	-0.455000	...	-0.445000	-0.290000	-0.425000
21	-0.330	-0.325	-0.335000	-0.480	-0.405	-0.470	-0.465000	-0.635	-0.675	-0.525000	...	-0.435000	-0.327014	-0.445000
22	-0.460	-0.375	-0.225000	-0.405	-0.450	-0.455	-0.310000	-0.675	-0.675	-0.540000	...	-0.480000	-0.454545	-0.545000
23	-0.360	-0.235	-0.105000	-0.170	-0.370	-0.490	-0.275000	-0.600	-0.680	-0.525000	...	-0.465000	-0.247423	-0.540000
24	-0.465	-0.345	-0.305000	-0.285	-0.360	-0.595	-0.260000	-0.490	-0.665	-0.541899	...	-0.567251	-0.087379	-0.555000

Republicans	Florida	Georgia	China	FoxNews	Pence	TrumpIsLosing
-0.766938	-0.869822	-0.829787	-0.610000	-0.637540	-0.658824	0.019608
-0.718750	-0.770186	-0.560976	-0.595000	-0.547541	-0.598086	-0.116279
-0.609329	-0.474510	-0.419355	-0.578680	-0.550201	-0.647887	-0.076923
-0.583333	-0.688172	-0.256410	-0.564033	-0.654902	-0.494253	-0.117647
-0.765000	-0.695652	-0.370370	-0.470000	-0.736842	-0.574468	-0.111111
-0.755000	-0.543689	-0.450980	-0.673469	-0.714286	-0.591398	-0.243243
-0.700000	-0.808333	-0.581395	-0.725000	-0.245614	-0.560000	0.187500
-0.758355	-0.811321	-0.684211	-0.660000	-0.643750	-0.555556	-0.314286
-0.575000	-0.682997	-0.400000	-0.655000	-0.420000	-0.568465	-0.416667
-0.591463	-0.694236	-0.660377	-0.735000	-0.238636	-0.662338	-0.444444
-0.534031	-0.824468	-0.463415	-0.710000	-0.528926	-0.655786	-0.363014
-0.700000	-0.728938	-0.494737	-0.760000	-0.657321	-0.652174	-0.430000
-0.715000	-0.777049	-0.546512	-0.665796	-0.638989	-0.702899	-0.390000
-0.681234	-0.634409	-0.155280	-0.645000	-0.595000	-0.783550	-0.413333
-0.575000	-0.640000	-0.648352	-0.675000	-0.550000	-0.604520	-0.480000
-0.655000	-0.660000	-0.675214	-0.761039	-0.433526	-0.623529	-0.317647
-0.725000	-0.675000	-0.161905	-0.676471	-0.458746	-0.643192	-0.434555
-0.720000	-0.735000	-0.517241	-0.669516	-0.590000	-0.655172	-0.336323
-0.670000	-0.705000	-0.462121	-0.650000	-0.490000	-0.737705	-0.425868
-0.435000	-0.685000	-0.430712	-0.660000	-0.565000	-0.459302	-0.365000
-0.585000	-0.680000	-0.445000	-0.620000	-0.540000	-0.685106	-0.528090
-0.445000	-0.705882	-0.440000	-0.580000	-0.525000	-0.651007	-0.470000
-0.705000	-0.565657	-0.265000	-0.485000	-0.510000	-0.651163	-0.335000
-0.590000	-0.604938	-0.645000	-0.445000	-0.470000	-0.702128	-0.300000
-0.585000	-0.617391	-0.775457	-0.750000	-0.440000	-0.700855	-0.225000