# Introduction

1. **Introduction**

In December 2019, an outbreak of pneumonia caused by a novel coronavirus occurred in Wuhan, and has spread rapidly throughout the globe(World Health Organization, 2020). The COVID-19 outbreak has forced people to cancel plans and practice social distancing, which increases people’s level of stress. Fear of being infected by the virus and the panic of losing jobs or education opportunities will add to peoples’ burden. The difficulties that the COVID-19 outbreak places on people in different geographic regions can determine their cause and degree of stress, which corresponds with risk of getting serious depression caused by COVID-19(CDC, 2020). According to KFF poll (KFF, 2020), nearly half (45%) of adults in the United States reported that their mental health has been negatively impacted due to worry and stress over the virus. As the pandemic continues, it is likely the mental health burden will increase as several actions have been taken to slow the spread of the virus, such as social distancing, business and school closures, and shelter-in-place orders, leading to greater isolation and potential financial distress.

Traditional mental health studies rely on information primarily collected through personal contact with a healthcare professional or through survey-based methods (e.g., via phone or online questionnaire). For instance, the Patient Health Questionnaire (PHQ) is a self-administered version of the PRIME-MD diagnostic instrument for common mental disorders(Kroenke and Spitzer 2002). However, these survey based methods are time consuming and suffer from cognitive and sampling biases, and therefore cannot be used to build large datasets for a real-time depression analysis (Haselton et al., 2005). Furthermore, understanding of spatial epidemic trends and geographic distribution patterns of COVID-19 provides timely information on people’s risk perception of epidemics, however, these important spatial and environmental leading factors are hard to be included in the survey based method to model COVID-19 related mental stress.

Geographic Information System (GIS) and social media data mining have become an essential tool to examine the spatial distribution of infections diseases (), which can aid in the process of investigating the spatiotemporal pattern of mental stress caused by the pandemic. For instance, social media data (e.g., Twitter data) provides a unique opportunity for learning about users’ moods, feelings, and behaviors that reflect their mental health as they experience daily struggles in real-time (De Choudhury et al., 2013; Coppersmith et al., 2014; Larsen et al., 2015). Many articles focused on using feature-based approaches to perform sentiment and emotional analysis using Twitter data. For instance, Go et al. [3] investigated the usage of unigrams, bigrams, and their combination in training the classifiers for sentiment analysis of tweets. Various supervised classifiers were trained, which included Maximum Entropy, Naive Bayes, and Support Vector Machines classifiers and their performance on the n-grams were compared. On the other hand, Barbosa and Feng [4] showed that n-grams are not useful in classifying tweets as the tweet contains a lot of words that are rarely used and this can cause problems in training the classifier. The authors proposed the usage of microblogging features like hashtags, emoticons, re-tweets and comments to train an SVM classifier and showed that it resulted in higher accuracy than training on n-grams.Several articles address the effect of using POS tag features in text classifiers. Agarwal et al. [5] investigated the POS, lexicon and microblogging features. The results showed that the most relevant features are ones that combine prior polarity with the POS tags of the words. There have however been mixed results reported on the usage of POS tags. Go et al. [3] showed that the POS tags resulted in a drop of performance, whereas Pak et al. [6] have showed that POS tags can be strong indicators of emotions in text and serve as a helpful feature in opinion or sentiment analysis. Moreover, Bootstrapping approaches rely on a seed list of opinion or emotion words to find other such words in a large corpus. Mihalcea et al. [7] describe two types of methods for bootstrapping the subjectivity lexicons into dictionary-based and corpus-based. However, all the above mentioned methods can only detect the emotion of Tweets in general and lack the ability to model depression levels in detail. Latent Dirichlet Allocation (LDA) is one of the most commonly used unsupervised topical methods, where a topic is a distribution of co-occurring words(Blei, Ng, and Jordan 2003). However, the topics learned by LDA are not specific enough to correspond to depressive symptoms and human judgments (Ramage et al., 2011). Later, Yazdavar et al. (2017) extended the LDA method by using terms that are strongly related to PHQ-9 depression symptoms as seeds of the topical clusters and guide the model to aggregate semantically-related terms into the same cluster. However, their approach uses Python TextBlob, a standard library to determine positive/neutral/negative polarity of the words that have multiple meanings. This step is not necessary in our approach as the context is determined using the dependency relations formed during bootstrapping. Also, their approach only detects the presence, duration, and frequency of stress symptoms, and ignores the spatial context which is important in modeling the COVID-19 spread pattern and its corresponding cause of mental stress.

In this article, we propose a CorExQ9 algorithm to detect COVID-19 related stress symptoms at a spatiotemporal scale. We aim to tackle research questions such as what are the stress symptoms of people in different geographic regions following the development of COVID-19 spread? What are the leading factors that cause mental stress in different regions at different times? What is the spatial uncertainty related to the level of stress symptoms caused by COVOD-19 pandemic?

1. **Analytical Approach**

**2.2.1 Bootstrapping the initial Keywords**

In this paper, we use a bootstrapping algorithm called Basilisk to find the semantic lexicons that can be later used to divide the Tweets into the two categories namely stressed and non-stressed. This approach can be extended to divide the Tweets into multiple categories across different areas. The Bootstrapping Approach to Semantic Lexicon Induction using Semantic Knowledge, also known as the Basilisk algorithm, was developed by Thelen and Riloff in 2002. It employs a bootstrapping method to determine high-quality semantic lexicons of nouns. The algorithm takes a huge unannotated corpus from where it finds new related words and assigns them to the different semantic categories, which is stressed and non-stressed in our case. It is a form of categorization that is based on the seed words that we manually provide to the algorithm. These seed words are bootstrapped to identify new words that fall within the two categories.

In the first step, Basilisk must be seeded with carefully selected terms for it to be effective. The two categories of seeds used for the task have 20 words each ( see Table 1 and Table 2). The first category (Table 1) contains words that describe stress and are used to bootstrap other words semantically related to stress or carrying similar context. Similarly, the second category (Table 1) contains words that describe non-stress or a relaxing behavior. These two categories contain words that fall on the opposite sides along the axis of stress.

Table 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Initial seed words related to being stressed: | | | | | | |
| addiction | boredom | dissatisfaction | grief | insecure | overwork | stress |
| alcoholism | burnout | distress | guilt | irritable | panic | tense |
| anger | conflict | embarrassment | headache | irritated | pressure | tension |
| anxiety | criticism | fear | impatience | loneliness | problem | tired |
| backaches | deadline | frustration | impatient | nervous | sadness | worry |
| Initial seed words related to being non-stressed: | | | | | | |
| chill | satisfaction | self-confidence | cure | relax | heart | prevention |
| enjoy | happiness | self-improvement | meditation | communication | right | self-talk |
| productivity | commitment | empower | wedding | perfectionism | change | tension |
| relax | perfection | joy | marriage | self-help | family | tired |
| relaxation | well-being | love | perfection | control | self-image | empowerment |

Before the bootstrapping process, the patterns must be extracted on the unannotated corpus. This is used to extract all the noun phrases that are either subject, direct object or prepositional phrase. The noun phrases are extracted from the corpus using the Stanford Dependency Parser. It is a natural language parsing program used to find grammatical structure in sentences and can be used to find relationship or dependencies between nouns and the actions or words that form a group and go together. The dependency parser is run on all the sentences in the corpus and dependency relations are extracted for each word in the text. These extracted dependency relations are then used to extract patterns that will be used by the Basilisk algorithm to generate seeds. These are called the extraction patterns and are created for each dependency relation obtained in the previous step. After the input has been generated, the next step is generating the seeds using Basilisk.

**2.2.2 Identify Stressed or Non-Stressed Tweets using Basilisk Algorithm**

The process of Basilisk as proposed by Thelen and Riloff can be described using the following algorithm (Table 2):

Table 2. Illustration of Basilisk algorithm.

|  |
| --- |
| Input: Extraction patterns in the unannotated corpus and their extractions, seed lists  Output: Updated list of seeds |
| lexicon = {seed words} for i := 0  1. Score all extraction patterns  2. pattern pool = top ranked 20+i patterns  3. candidate word pool = extractions of patterns in pattern pool  4. Score candidate words in candidate word pool  5. Add top 5 candidate words to lexicon  6. i := i + 1  7. Go to Step 1. |

Using Basilisk algorithm, we count the total number of occurrences of any of the keywords in both the categories. After we obtain the total count of stress and non-stress words in each tweet, we determine whether the tweet falls in the category of stressed or not stressed or neutral. This is done by finding the maximum of the stress and non-stress word counts. If the number of stress words are more than the non-stress words, we annotate the tweet to express stress. If the number of non-stress words are more than the stress words, we annotate the tweet to express relaxed behavior. If the count is zero for both the stress and non-stress words, we don't annotate the data at this stage. The tweets and their corresponding labels generated using this process are the initial training set, which are used in the next stages to train a classifier to classify the other unannotated tweets.

**2.2.3 Generate Word Embeddings and Train the Classifier**

The Universal Sentence Encoder was used to generate word embeddings. These text embeddings convert the tweets into a numerical vector. It encodes the tweet text into high dimensional vectors which can be used for finding semantic similarity and performing the classification task. It takes a variable length English text as input and the outputs a 512-dimensional vector embedding. The encoder model is trained with a deep averaging network (DAN) encoder. After the word embeddings were obtained for each stressed and non-stressed category tweets, a technique was used to make the two classes equalized. To do this, we select the category with a lesser number of samples, and make the other category of similar size by removing extra samples. This ensures that the training process is not biased towards a particular class.

Before training the classifier, the data was splitted into training and validation sets for training the classifier. The data is randomly shuffled and put into the two dataset splits with the test to train split as 0.8. To obtain the best performance, multiple classifiers were used and performance were compared using accuracy metrics. The classifiers used in the training process are Support Vector Machine (SVM)[8], logistic regression, Naïve Bayes Classifier(), and Simple Neural Network ().

SVM handles nonlinear input spaces and separates data points using a hyperplane using the largest amount of margin. SVM being a discriminative classifier finds an optimal hyperplane for our data which helps in classifying new unannotated data points. We use different kernels for training the SVM and find out that the linear kernel performs the best for our dataset. The hyperparameters are tuned and the optimal value of regularization and gamma are also recorded. The logistic regression classification algorithm can be used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (stressed) or 0 (not stressed). The logistic regression model predicts P(Y=1) as a function of X. Naive Bayes is another probabilistic classifier which makes classifications using the Bayes rule. This classifier is simple and effective for text classification. A Simple Neural Network consisting of three dense layers is used to train our datasets. The loss function and optimizer used in the training is binary crossentropy and RMSProp. The training is done for 40 epochs with a batch size of 512. Table 3 illustrates the performance evaluation of these classifiers.

Table 3. Performance evaluation of classifiers.

|  |  |
| --- | --- |
| **Model** | **Validation Accuracy** |
| **SVM (rbf kernel)** | 0.8218 |
| **SVM (Linear kernel)** | 0.8668 |
| **Logistic Regression** | 0.8620 |
| **Naïve Bayes** | 0.8076 |
| **Simple Neural Network** | 0.8690 |

**2.2.4 Generate labels using the trained classifier**

After the model has been trained, the model is run on the unannotated tweets to label them. To get the sentence embeddings for the tweets to be labeled, the same procedure is used as for the training set. The Universal Sentence Encoder extracts 512 features and creates vectors that can be used to classify the tweets based on the model. The SVM classifier was used to predict the probabilities of the tweets Since the SVM classifier with linear kernel performed the best among all the trained models(see Table 3). Here, a threshold of 0.75 was set to determine if the tweet belongs to a particular category or not. If the probability of the tweet is above 0.7 for that category, the tweet will be classified with the corresponding label.

The tweets and labels generated using the above process are then again used to train another classifier to generate the final model for classification of the entire unannotated corpus. Here, a Logistic Regression model was used to train tweets and their corresponding labels generated using the above process to ensure that the model is more robust and is not overfitted on the initial set of tweets that were filtered out using the Basilisk generated keywords. The trained model has an accuracy of 90.2% on the validation data.

# 

---------------------------------------------------------------------------------------------------------------

# Methods

## Bootstrapping the Initial Keywords

In this paper, we use a bootstrapping algorithm called Basilisk to find the semantic lexicons that can be later used to divide the Tweets into the two categories namely stressed and non-stressed. This approach can be extended to divide the Tweets into multiple categories across different areas. The Bootstrapping Approach to Semantic Lexicon Induction using Semantic Knowledge, also known as the Basilisk algorithm, was developed by Thelen and Riloff in 2002[1]. It employs a bootstrapping method to determine high-quality semantic lexicons of nouns. The algorithm takes a huge unannotated corpus from where it finds new related words and assigns them to the different semantic categories, which is stressed and non-stressed in our case. It is a form of categorization that is based on the seed words that we manually provide to the algorithm. These seed words are bootstrapped to identify new words that fall within the two categories.

In the first step, Basilisk must be seeded with carefully selected terms for it to be effective. The two categories of seeds used for the task have 20 words each ( see Table 1 and Table 2). The first category (Table 1) contains words that describe stress and are used to bootstrap other words semantically related to stress or carrying similar context. Similarly, the second category (Table 2) contains words that describe non-stress or a relaxing behavior. These two categories contain words that fall on the opposite sides along the axis of stress.

Before the bootstrapping process, the patterns must be extracted on the unannotated corpus. This is used to extract all the noun phrases that are either subject, direct object or prepositional phrase. The noun phrases are extracted from the corpus using the Stanford Dependency Parser[2]. It is a natural language parsing program used to find grammatical structure in sentences and can be used to find relationship or dependencies between nouns and the actions or words that form a group and go together. The dependency parser is run on all the sentences in the corpus and dependency relations are extracted in the CoNLL-U format for each word in the text.

For the sentence, “watch this newly released video”, the extracted dependency relation looks like this:

1 watch watch VB VB Mood=Imp|VerbForm=Fin 0 root \_ \_

2 this this DT DT Number=Sing|PronType=Dem 5 det \_ \_

3 newly newly RB RB \_ 4 advmod \_ \_

4 released release VBN VBN Tense=Past|VerbForm=Part 5 amod \_ \_

5 video video NN NN Number=Sing 1 obj \_ \_

The format of the pattern extracted is:

index text lemma xpos xpos feats governor dependency\_relation

index - Index of the word in the sentence

text - Text of the word at the particular index

lemma - Lemmatized value of the word

xpos - Treebank-specific part-of-speech of the word. Example: 'NNP'

feats - Morphological features of the word. Example: 'Gender=Fem'

governor - The index of governor of the word, which is 0 for root

dependency\_relation - dependency relation of the word with the governor word which is root if governor = 0 . Example: 'nmod'

Here, if we consider a pattern that looks like:

1 watch watch VB VB Mood=Imp|VerbForm=Fin 0 root \_ \_

Here, the index of the word into consideration is 1, watch is the text, the lemmatized version of the word watch is watch itself, the features are Mood which is imperative and verb form that is finite. The word has no governing word, so the value of governor is 0 and the dependency with it is root.

For this pattern:

2 this this DT DT Number=Sing|PronType=Dem 5 det \_ \_

The index of the word into consideration is 2, ‘the’ is the text, the lemmatized version of the word watch is ‘the’ itself, the features are Number which is singular and pronoun type which is demonstrative pronoun. The word has a governing word, which is ‘video’. Since the word video has index ‘5’, the value of governor is ‘5’ and the dependency with it is ‘det’, which means it is a determiner for the word at index 5.

These extracted dependency relations are then used to extract patterns that will be used by the Basilisk algorithm to generate seeds. These are called the extraction patterns and are created for each dependency relation obtained in the previous step. The extraction patterns are created by finding the dependency of each noun with other words in the sentence. For example, the word ‘video’ is the noun. Therefore, the extraction pattern of the word would contain dependencies of the following words: ‘this’ as the determiner, and ‘released’ as the adjectival modifier.

After the input has been generated, the next step is generating the seeds using Basilisk.

The process of Basilisk as proposed by Thelen and Riloff[1] can be described using the following algorithm:

Input: Extraction patterns in the unannotated corpus and their extractions, seed lists

Output: Updated list of seeds

lexicon = {seed words} for i := 0

1. Score all extraction patterns

2. pattern pool = top ranked 20+i patterns

3. candidate word pool = extractions of patterns in pattern pool

4. Score candidate words in candidate word pool

5. Add top 5 candidate words to lexicon

6. i := i + 1

7. Go to Step 1.

During the process of similar words generation, the stopwords were removed because they are irrelevant and do not carry any information. Also, they can negatively affect the algorithm as they appear a lot of times in the data.

Initial seed words related to being stressed:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| addiction | boredom | dissatisfaction | grief | insecure | overwork | stress |
| alcoholism | burnout | distress | guilt | irritable | panic | tense |
| anger | conflict | embarrassment | headache | irritated | pressure | tension |
| anxiety | criticism | fear | impatience | loneliness | problem | tired |
| backaches | deadline | frustration | impatient | nervous | sadness | worry |

Initial seed words related to being not stressed:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| chill | satisfaction | self-confidence | cure | relax | heart | prevention |
| enjoy | happiness | self-improvement | meditation | communication | right | self-talk |
| productivity | commitment | empower | wedding | perfectionism | change | tension |
| relax | perfection | joy | marriage | self-help | family | tired |
| relaxation | well-being | love | perfection | control | self-image | empowerment |

After generating stress and non-stress related words using the above algorithm, the next step is to use these words to identify an initial set of tweets that contain these words. This is done by simple keyword filtering in the Tweets. However, before this can be used to filter the tweets, data cleaning needs to be done.

## Data Preprocessing

The data was obtained from Twitter and consists of tweets with the following hashtags: #coronavirus, #coronavirusoutbreak, #coronavirusPandemic, #covid19, #covid\_19. The data was composed of around 6.8 million Tweets in English language. This data needs to be prepared for the subsequents steps as it is very noisy. The objective of the data preprocessing step is to remove the irrelevant information (e.g., remove non-English language Tweets, remove punctuation, lower case, URL, user name, hashtags, stop words, numbers, and query terms ) from the tweets and leave behind the useful one. This is one of the most practiced steps in information storage and retrieval systems, where data is preprocessed in order to make retrieval better by reducing noise. For the text classification tasks, it helps in improving the quality of features and in turn make the models perform better. The data is passed through multiple preprocessing steps before applying the classification model. The next step involves tokenizing the tweets using NLTK’s TweetTokenizer. We also replace the repeated character sequences that are of length 3 or greater with the sequences of length 3. This is done because Twitter users often tend to be very informal in the language that they use. Most users extend the words or add redundant characters to express strong feelings. For example, the term “yaaaayyyyyyyyy” coveys a stronger expression than “yay”. In our project, we remove the occurrence of more than 3 subsequent occurrences of the same repeated letter in the words and replace it with a sequence of three characters. For example, we convert “yaaaayyyyyyyyy” to “yaaayyy”.

## Filtering the tweets based on keywords generated using the basilisk algorithm

After tokenizing the tweets, we use the keywords that were generated for the two categories of stressed and non-stressed using the Basilisk algorithm. To do this, we count the total number of occurrences of any of the keywords in both the categories. After we obtain the total count of stress and non-stress words in each tweet, we determine whether the tweet falls in the category of stressed or not stressed or neutral. This is done by finding the maximum of the stress and non-stress word counts. If the number of stress words are more than the non-stress words, we annotate the tweet to express stress. If the number of non-stress words are more than the stress words, we annotate the tweet to express relaxed behavior. If the count is zero for both the stress and non-stress words, we don't annotate the data at this stage. The tweets and their corresponding labels generated using this process are the initial training set, which are used in the next stages to train a classifier to classify the other unannotated tweets.

## Generating word embeddings

In this step, we generate embedding vectors for the cleaned tweets obtained using the above steps. The first step is to generate the text embeddings for the tweets that have been annotated as stressed and the tweets that have been annotated as non-stressed. These text embeddings convert the tweets into a numerical vector. The Universal Sentence Encoder is used for this purpose. It encodes the tweet text into high dimensional vectors which can be used for finding semantic similarity and performing the classification task. The pre-trained Universal Sentence Encoder is available publicly in Tensorflow-hub. It has been trained on a variety of tasks and a variety of data sources with the objective of making various natural language processing tasks easier. It takes a variable length English text as input and the outputs a 512-dimensional vector embedding. The encoder model is trained with a deep averaging network (DAN) encoder.

After we obtain the word embeddings for each stressed and non-stressed category tweets, we have to equalize the two classes to make the dataset more balanced. To do this, we select the category with a lesser number of samples, and make the other category of similar size by removing extra samples. This ensures that the training process is not biased towards a particular class.

## Training the Classifier

Before training the classifier, we split the test data into training and validation sets. The data is randomly shuffled and put into the two dataset splits with the test to train split as 0.8. To obtain the best performance, we train multiple classifiers and compare the performance of them using the accuracy metric.

The various classifiers that were used to compare the performance are listed below:

Support Vector Machine

Support Vector Machine(SVM) offers very high accuracy for many classification tasks and is known for its kernel trick. It handles nonlinear input spaces and separates data points using a hyperplane using the largest amount of margin. SVM being a discriminative classifier finds an optimal hyperplane for our data which helps in classifying new unannotated data points. We use different kernels for training the SVM and find out that the linear kernel performs the best for our dataset. The hyperparameters are tuned and the optimal value of regularization and gamma are also recorded.

Logistic Regression

The logistic regression classification algorithm is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (stressed) or 0 (not stressed). The logistic regression model predicts P(Y=1) as a function of X. We train a simple logistic regression model on the data split and record the performance.

Naïve Bayes Classifier

Naive Bayes is a probabilistic classifier which makes classifications using the Bayes rule. This classifier is simple and effective for text classification and is a good choice for our task.

Simple Neural Network with three dense layers

A neural network consisting of three dense layers is created using Keras and trained on the word embeddings. For the training, the loss function used is binary crossentropy, and the optimizer used is RMSProp. The performance is evaluated on the accuracy metric. The training is done for 40 epochs with a batch size of 512.

The performance of various models is listed below:

|  |  |
| --- | --- |
| **Model** | **Validation Accuracy** |
| **SVM (rbf kernel)** | 0.8218 |
| **SVM (Linear kernel)** | 0.8668 |
| **Logistic Regression** | 0.8620 |
| **Naïve Bayes** | 0.8076 |
| **Simple Neural Network** | 0.8690 |

## Generating labels using the classifier & Training Further

After the model has been trained, the model is run on the unannotated tweets to label them. To get the sentence embeddings for the tweets to be labeled, the same procedure is used as for the training set. The Universal Sentence Encoder extracts 512 features and creates vectors that can be used to classify the tweets based on the model. Since the SVM classifier with linear kernel performed the best among all the trained models, we use the SVM classifier to predict the probabilities of the tweets.

After predicting the labels, we set a threshold of 0.75 to determine if the tweet belong to a particular category or not. This means that if the probability of the tweet is above 0.7 for that category, we classify it with the corresponding label. This is done to ensure that we consider tweets that have high confidence levels.

The tweets and labels generated using the above process are then again used to train another classifier that would be the final model for classification of the entire unannotated corpus. For this, we use a Logistic Regression model and train it on the tweets and their corresponding labels generated using the above process. This is done to ensure that the model is more robust and is not overfitted on the initial set of tweets that were filtered out using the Basilisk generated keywords. The trained model has an accuracy of 90.2% on the validation data. This is the model that is saved and used for annotating other tweets.

The twitter data we used in this research are tweets data collected between January and April. Part of tweets are hydrated from The First public Coronavirus Twitter Dataset (Chen, Lerman, and Ferrara 2020). Another part of tweets are collected by Twitter's official streaming API. All those tweets contain coronavirus related entities (hashtag, trends and news). Around eight million tweets are collected as raw data in our research.

The independent variables like median household income, hospital count by county, ICU bed count by county, unemployment rate, etc. are collected from the United States and Census Bureau and self-developed web spiders.

## **1.1** **Data Preprocessing**

### **1.1.1** **Extract Location**

This research is expected to find out the spatial relationship between the coronavirus and stress level of individuals. Thus, not all tweets are applicable to use in our model. We separated tweets data into three different categories, each needs specific preprocessing method:

1. For those tweets with geotag, we use reverse geocoding method to locate the county of the tweet. Reverse geocoding is the process of back coding a point location (latitude, longitude) to a readable address or area. If those tweets are outside of the U.S. or the county name is missing, then those twitter will not be counted in our following research.

2. For those tweets without geotag but its user profile contains user-defined location, we use fuzzy search and regular geocoding method to locate those tweets. This substitution method is heavily used by twitter official API. Defined by twitter official API description, user defined location is used on its advanced tweets search operation (Twitter 2020). Some of the user-defined location profiles may not be precisely described as its name on census data. For example, most users use ‘NYC’ to represent ‘New York City’. We use fuzzy set search (Ji et al. 2009) and city alias dataset (grammakov 2020) to match each tweet to the county name. For those tweets with unfounded location and unformatted user location profile, we will abandon those tweets.

3. If tweets don’t have any location mark on user profile and geo tag, we will not use those tweets in our following research.

### **1.1.2** **Clean Text**

As we probably know, natural language processing (NLP) algorithms process the text data in a way of computer processing numbers. The raw twitter data might easily be understood by humans but it is hard to be understood by computers. Some oral expressions and emoji are considered as noise data in NLP models. In this research, we use Natural Language Toolkit (NLTK) and scikit-learn toolkit to tokenize and vectorize the filtered tweets data. Besides, spatial symbols and characters are also required to be cleaned in our following method (e.g. punctuation and URL).

### **1.1.3** **Resample Time Windows**

The causes of stress may change over time. To find out the temporal pattern among the whole bunch of tweets, we resample tweets by two week frequency. Next, we re-organize those tweets into different time windows. However, due to the previous preprocessing step, the number of tweets varies from different time windows.

### **1.1.4** **Standardization and Normalization**

Standardization of datasets is a common requirement for many algorithms. Some input datasets might behave badly if the individual features do not more or less look like standard normally distributed data. The normalization is the common process of scaling individual samples to have unit norm. This process can be useful if we are using a global regression model to fit our data with independent variables. In our case, two standardization methods are used to scale the features of our dataset. For the chaotic distribution in resampled time windows inside, we transform the tweets to center by removing the mean value of each feature, then scale it by dividing non-constant features by their standard deviation. For the generated stress index and cleaned independent variable generated in later steps, we transform features by scaling each feature to a given range. The transformation is given by:

Where are feature ranges.

# **2** **Method**

The Patient Health Questionnaire (PHQ) is a self-administered version of the PRIME-MD diagnostic instrument for common mental disorders. At 9 items, the PHQ depression scale is a widely accepted depression measure (Kroenke and Spitzer 2002). In this case, we use the PHQ-9 depression lexicon (1704 words) as anchored knowledge to find out the possible causes of different stress levels. Common topic modeling method use probabilistic generative model to find out how spares knowledge in a document is organized by different topics, e.g. Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan 2003). The general mechanism of LDA does not allow topic inference to be tractable if we expect to find out the relationship between PHQ-9 and coronavirus related tweets. The later research proposed correlation explanation (CorEx) in a way of learning latent topics by anchored topics (Ver Steeg and Galstyan 2014; Gallagher et al. 2017). However, most words in PHQ-9 lexicon are psychology or pathology terminologies. Therefore, we need to first build up a simplified PHQ-9. Then we take advantage of the previous research and utilizing correlation explanation to fit our stressed tweets, which is expected to conduct the PHQ-9 stress topic spatial-temporal analysis.

## **2.1** **PHQ-9 Lexicon**

Depression is one of the most prevalent and treatable mental disorders in general medical as well as specialty settings. Relevant research shows that the cause of stress is highly related with depression (Kubo 2007). In our research, we adopted PHQ-9 lexicon (examples show on *Table 1*) based on description from previous questionnaire result (Lewis 2005) as an anchored word to calculated the correlation between tweets and stress level. However, The problem of calculating the semantic similarity between two concepts, words or phrase is a long dealt problem in the area of NLP. In general, semantic similarity is a measure of conceptual distance between two object, based on the correspondence of their meanings. Even though the PHQ-9 is verified to precisely describe individuals depression level in an academic expression, tweets from twitter user are more likely to use oral expression. Therefore, lexicon mismatch and the vague semantic meaning are two big challenges when we analysis the stressing tweets with lexicon.

***Table 1*** *PHQ-9 Lexicon Examples*

|  |  |
| --- | --- |
| **Description** | **Lexicon Examples** |
| Little interest or pleasure in doing things | Acedia, anhedonia, bored, boring, can’t be bothered… |
| Feeling down, depressed | Abject, affliction, agony, all torn up, bad day… |
| Trouble Falling or staying asleep | Active at night, all nightery, awake, bad sleep… |
| Feeling tired or having little energy | Bushed, debilitate, did nothing, dog tired… |
| Poor appetite or overeating | Abdominals, anorectic, anorexia, as big as a mountain… |
| Feeling bad about yourself | I am a burden, abhorrence, forgotten, give up… |
| Trouble concentrating on things | Absent minded, absorbed, abstracted, addled… |
| Moving or speaking so slowly that other people could have noticed | Adagio, agitated, angry, annoyed, disconcert, furious… |
| Thoughts that you would be better off dead | Belt down, benumb, better be dead, blade, bleed… |

To reduce the influence of vocabulary mismatch and ambiguity of PHQ-9 lexicon, first, we need to create a vocabulary set and a phrase set. The tweets vocabulary set are created by using the set of all words used in the cleaned tweet corpus. Next, the shallow parsing step, which is the process of extracting phrase from unstructured text, is performed by a general industrial NLP toolkit spaCy (Honnibal, Matthew and Montani 2017). The input corpus in this step is half cleaned tweets (the stop words are kept temporary before shallow parsing). All words in phrase set is included in vocabulary set.

Second, we reduce the complexity of PHQ-9 by matching sophisticated medical term with simple phrase in tweets(e.g. match *anhedonia* with *reduced motivation* or *unable to feel pleasure*). A common method to pair phrases or sentences is cosine distance (Handler 2014), which is an effective method for measuring the linguistic or semantic similarity of the corresponding words. Many different types of methods were proposed for words as continuous vectors including latent semantic analysis (LSA) and LDA. Those method using the assumption that similar words will occur in similar documents – LSA creates as matrix, whereby the rows represent unique words and the columns represent each paragraph. Then LSA applies singular value decomposition (SVD), which is used to reduce the number of rows while preserving the similarity structure among columns. The problem is that those modes become computationally very expensive in our large research case (Background review). In our research, we utilize global vectors for word representation (GloVe) to vectorize cleaned tweet and PHQ-9 lexicon. GloVe provides a quantitative way to distinguish nuance difference of two word (e.g. happy or unhappy), which is necessary for PHQ-9 lexicon to associate more than a single dimension to word pair. Besides, to avoid data sparsity issues, we obtain our lexicon vectors by *Wikipedia and Gigaword 5* pre-trained vectors, which contains 300 dimensional word vectors with four million unique tokens. Those pre-trained vectors are loaded to Gensim (Sojka 2010) to perform the following calculation. Now that we have two sets of vectorized phrases, we calculate the average feature vector for each phrase before we use cosine distance to match tweets phrase. Given two average vectors of phrase, and , the cosine similarity, , is represented by:

We keep all phrase with high similarity rate (above 0.8) for each level of PHQ-9 lexicon.

## **2.2** **PHQ-9 Stress Topic Detection**

To detect each level simplified PHQ-9 topic from COVID-19 tweets set, we utilize the anchored CorEx model, which can be flexibly worked with correlation explanation through anchor words (Gallagher et al. 2017). In particular, a controllable human intervention can conduct to extracting topic in COVID-19 tweets set. First, to calculated the correlation of COVID-19 corpus and simplified PHQ-9 lexicon, we transform those tokenized and vectorized tweets into a sparse matrix, which is considered as vocabulary dense matrix in our case. We use term frequency inverse document frequency (TFIDF) to get terms per document with nonzero entries. The math formula for this measure is:

Where denotes the terms; denotes each documents; denotes the collection of documents. The first part of formula is to calculate the number of times each word in COVID-19 corpus appeared in each document. The second part of is made up with a numerator and a denominator . Numerator is inferring to document space, which are all documents in our COVID-19 corpus. Denominator implies the total number of times in which term t appeared in all of our document . The can be represented by:

After that, we calculate total correlation of each topic by anchored CorEx model with the result of TFIDF. The total correlation in our PHQ-9 lexicon detection can be expressed using Kullback-Leibler Divergence as below (Ver Steeg and Galstyan 2015).

Where represents the probability distribution and is non-negative or zero factorizes of . In our context of PHQ-9 detection, represents the group of word types among the COVID-19 corpus. When introducing an random variable , the can explains the correlations reduce in , which is a measure of the redundant information that the word types carry about topic (Schneidman, Bialek, and Berry 2003). The final total correlation can be represented by:

Where .

Thus, the implement of correlation explanation starts with a randomly initialized variables and, where is indicator variables of which are assigned to 1 if topic detected and the represents the approximate empirical distribution. Then, the correlation explanation updates both variables iteratively until the result achieve convergence. In each iteration, first, the estimate marginals and mutual information are calculated. Note that is the Kronecker delta of random variables . Next, the update for in each t step is calculated by:

Where conduct a smooth optimization of the soft-max function (Steeg and Galstyan 2014) . Finally, the soft labeling of any can be computed by:

After the soft-max function converges to the true solution at a particular step in the limit , the mutual information terms can be rank by the informative order in each factor. To perform semi-supervised anchoring strategies, a bottleneck function is proposed by Ryan Gallagher and Kyle Reing in 2017 (Gallagher et al. 2017). Note that Z is typically labels in a supervised learning task (Tishby, Pereira, and Bialek 2000; Friedman 2001) and is a constant parameter to constrain supervising strength so that can imply a word type correlated with topic . In particular, in our context of case Z represented by each variable generated by simplified PHQ-9 lexicon.

# References

1. Thelen, M., & Riloff, E. (2002). A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts. EMNLP.
2. Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, David. 2014. [The Stanford CoreNLP Natural Language Processing Toolkit](http://nlp.stanford.edu/pubs/StanfordCoreNlp2014.pdf) In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60.[[pdf](http://nlp.stanford.edu/pubs/StanfordCoreNlp2014.pdf)][[bib](http://nlp.stanford.edu/pubs/StanfordCoreNlp2014.bib)]
3. Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Processing, pages 1–6.
4. Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on Twitter from biased and noisy data. In Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING ’10). Association for Computational Linguistics, USA, 36–44.
5. Agarwal, Apoorv & Xie, Boyi & Vovsha, Ilia & Rambow, Owen & Passonneau, Rebecca. (2011). Sentiment Analysis of Twitter Data. Proceedings of the Workshop on Languages in Social Media.
6. Pak, Alexander & Paroubek, Patrick. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. Proceedings of LREC. 10.
7. R. Mihalcea, C. Banea, and J. Wiebe. 2007. Learning multilingual subjective language via cross-lingual projections. In Proceedings of the Association for Computational Linguistics, Prague, Czech Republic
8. Bernhard E. Boser, [Isabelle M. Guyon](http://www.gabormelli.com/RKB/Isabelle_M._Guyon), and [Vladimir N. Vapnik](http://www.gabormelli.com/RKB/Vladimir_N._Vapnik). A training algorithm for optimal margin classifiers. In [D. Haussler](http://www.gabormelli.com/RKB/D._Haussler), editor, Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, pages 144–152, Pittsburgh, PA, July ([1992](http://www.gabormelli.com/RKB/1992)). ACM Press.