



Predicting the pandemic: sentiment evaluation and predictive analysis from large-scale tweets on Covid-19 by deep convolutional neural network

Sourav Das¹ · Anup Kumar Kolya²

Received: 11 June 2020 / Revised: 7 January 2021 / Accepted: 21 March 2021 / Published online: 30 March 2021
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract

Engaging deep neural networks for textual sentiment analysis is an extensively practiced domain of research. Textual sentiment classification harnesses the full computational potential of deep learning models. Typically, these research works are carried either with a popular open-source data corpus, or self-extracted short phrase texts from Twitter, Reddit, or web-scraped text data from other resources. Rarely do we see a large amount of data on a current ongoing event is being collected and cultured further. Also, an even more complex task would be to model the data from a currently ongoing event, not only for scaling the sentiment accuracy but also for making a predictive analysis for the same. In this paper, we propose a novel approach for achieving sentiment evaluation accuracy by using a deep neural network on live-streamed tweets on Coronavirus and future case growth prediction. We develop a large tweet corpus exclusively based on the Coronavirus tweets. We split the data into train and test sets, alongside we perform polarity classification and trend analysis. The refined outcome from the trend analysis helps to train the data to provide an incremental learning curvature for our neural network, and we obtain an accuracy of 90.67%. Finally, we provide a statistical-based future prediction for Coronavirus cases growth. Not only our model outperforms several previous state-of-art experiments in overall sentiment accuracy comparison for similar tasks, but it also maintains a throughout performance stability among all the test cases when tested with several popular open-source text corpora.

Keywords Coronavirus · Covid-19 · Sentiment analysis · Predictive analysis · Deep convolutional network · Twitter

1 Introduction

Twitter is not a newbie term anymore in today's natural language research perspective. In the last decade, this micro-blogging platform has emerged as one the first choice for expressing opinions, conducting market reviews, advertisement campaigns, and informative links, all within a very concise space. Not only that, but the 280-character space of

tweets also¹ lets the users post any statement on any topic with a limited choice of words and directly targeted opinion polarity.

This is probably why Twitter is one of the most famous options for textual data extraction for researchers and organizations [1, 2]. The tweet collections lead to many practical real-world applications, naming sentiment analysis, rumor prediction, spam filtering, market survey, etc. With 330 million² active Twitter users currently in the world, it is quite obvious that this platform generates large clusters of opinions based on regions, issues, events, and topics. The concept of basic emotion categorization and cognitive sentiment identification originated long before the emergence of social networks [3]. Since then, it has been a focal point of research. Sentiment analysis of such large-scale events has been under the focus for a decade now [4–6]. Most of

✉ Sourav Das
sourav.das.research@gmail.com

Anup Kumar Kolya
anup.kolya@gmail.com

¹ Department of Computer Applications, University of Engineering & Management Kolkata, New Town, Kolkata, India

² Department of Computer Science and Engineering, RCC Institute of Information Technology, Belegkata, Kolkata, India

¹ <https://www.bbc.com/news/technology-41900880>.

² <https://www.omnicoreagency.com/twitter-statistics/>.

such works, and overall sentiment classification has also experimented on other similar social media users' opinions. The three basic perspectives of sentiment analysis, namely document level [7], sentence-level [8], and aspect level sentiment analysis [9] have been diversely carried on large tweet corpora. These works meticulously explain whether a whole opinion document expresses a positive or negative sentiment [10]. This process of filtration further brings out the level of analysis closely related to subjectivity classification [11]. Additionally, aspect or entity-level sentiment analysis directly looks at the opinion itself. It is based on the idea that an opinion consists of sentiment (positive or negative) and a target (of opinion). An opinion without its target being identified is of limited use [12]. Based on this level of analysis, a structured summary of opinions about entities and their aspects can be produced, which turns unstructured text into structured data and can be used for all kinds of qualitative and quantitative analysis.

A similar topic of sentiment research prospect recently originated with the Coronavirus pandemic. It is being stated as the greatest challenge for humankind since the 2nd world war,³ with millions of already affected combining China, India, entire Europe, and the North American region, as well as the other continents of the world. While originally incepted from the local meat-market of Wuhan city of Hubei province in China, this virus is also known as several other aliases such as bat coronavirus [3, 13], novel Coronavirus, and Covid-19.⁴ It is already mentioned by several distinguished research findings and medical bulletins that the novel Coronavirus or Covid-19 has a lower fatality but higher contamination and spreading rate than that of the SARS and MERS coronavirus that has spread before.⁵ Recently we have also witnessed the shifting of the coronavirus epicenter from China to Europe, resulting in a major number of deaths in Italy and Spain.⁶

It is evident without saying that an incident of such a large statue has already attracted a plethora of research works relating the Twitter-based sentiment analysis from a healthcare standpoint. Real-world applications of such sentiment evaluation outcomes can benefit us in multiple ways and domains. Kabir and Madria have developed an application CoronaVis based on USA's Covid-19 tweet dataset that is named as CoronaVis Twitter dataset [14]. They have approached for a data analytics model made from scratch by streaming live tweets on Covid-19. Further, they

demonstrated the changes of topics, human emotions, and subjectivity over the time using feature attributes as frequent word, top frequent Bigrams, topic trends, sentiment (polarity), user movement and topic modeling. Yang et al. developed SenWave fine-grained sentiment analysis model that showed increased positive sentiment over time [15]. They collected a cumulation of 105 million multilingual tweets from different continents, and manually annotated the tweets on several fine-grained emotion categories. Overall, they presented a polarity refined dataset in major languages for additional analysis and future trend direction. Finally, the researchers used a deep learning classifier to evaluate the manual annotations with actual emotion detection from the tweets. Samuel et al. addressed the issues of the people's fear or panic sentiment progression based on Coronavirus tweets [16]. They reported that Naïve Bayes performs better than logistic regression for short tweets whereas both have shown better performance for long tweets. They filtered out public sentiment reflection on Covid-19 tweets, as well as visualizing them for a better understanding of the subject matter. Lamsal reported Covid-19 tweet dataset design [17]. In detail design, the authors have shown people's understanding during the crisis of the present corona pandemic by temporal and spatial dimension of the dataset. The author has also explored the network visualizations for churning out the impact of sentiment related to Covid-19 in different regions. From the network clusters, the author has tagged the IDs of the tweeting people for distinctly identifying them and the sentiment associated with their tweets. Furthermore, the load balancing and CPU optimization of the system used has also been demonstrated while performing the experimental phases. Chakraborty et al. analyzed that at the initial phase of the Coronavirus pandemic negative and neutral tweets were more in numbers and comparatively in the later part of the pandemic positive and neutral tweets were more [18]. They used the Gaussian fuzzy classifier to evaluate the social media sentiments of people in the early months of the Covid-19 virus spread-out. Their primary motivation was to monitor the negative psychology correlated with the pandemic itself. Often tweets do not express singularly clear sentiment polarities. Hence, they implemented a fuzzy logic-based approach to determine the majority of sentiment flavor from the tweets to validate their claim. During the current pandemic scenario, we can utilize such frameworks to model end-to-end healthcare, mental support, and panic resistant systems. Often in this time a large section of the community urgently needs assistance with a bulk of medical information. Their emotions are usually associated with the group of fear, grief, helplessness, and pensiveness, which corresponds to overall negative polarity after all. An exact sentiment evaluation system would be able to cultivate and treat such human sentiment polarity for a better understanding of the pandemic scenario.

³ <https://www.bbc.com/news/world-52114829>.

⁴ <https://www.who.int>.

⁵ <https://www.medicalnewstoday.com/articles/how-do-sars-and-mers-compare-with-covid-19>.

⁶ <https://www.nytimes.com/2020/03/21/world/europe/italy-coronavirus-center-lessons.html>.

Recently, a variety of deep learning models are heavily being deployed for standard NLP tasks [19, 20]. One such model which is certainly under the focus is the convolutional neural network (CNN). The classic convolutional network has proven to be successful for experiments like text mining, polarity generation, information retrieval, and so on. Often for the discussed tasks, the convolutional network outperforms its other deep learning counterparts [21, 22]. For filtering sentence classification within a short scope, unidirectional convolution layers can be stated as translation invariant, as the output is directly transformed from single channeled input, i.e. the similar compiled set of words within a sentence is selected and positioned on every patch. Accordingly, a pattern is learned from the combination of that particular matrix patch position which later is shifted and stored at a different position. Both 1D and 2D convolutional networks emphasize input and output either the maximum or an average value of the total input parameters. This mechanism is technically referred to as max pooling. In correspondence with that, it is also used for trimming the length of the 1D input channel, i.e., subsampling.

1.1 Problem definition

In this work, we propose a novel approach for phase-by-phase data modeling of large-scale Coronavirus tweets and thereafter evaluating the sentiments with accuracy by the deep convolutional network. Not only restricted to that, we further propose to deploy the sentiment evaluated data for predictive analysis of Coronavirus growth plotting within a limited timeline. Upon collecting tweets on the given topic of interest, we apply the logistic regression algorithm with the training partitioned data for polarity labeling. This data helps us to make the trend analysis using the gradient of parameters taking into account. Next in line, we feed the sentiment labeled tweets in our neural network for training. In this phase, we also feed our CNN model with one layer of convolution on top of word vectors which were loaded from an unsupervised neural language model. These vectors are the non-zero elemental correlation produced by global vectors or *GloVe* [23]. We feed the word vectors explicitly to make the model learn only the short text evaluation. In the testing or validation phase, the test tweets are blended in five epochs each. We achieve an accuracy result peak of 90.67% among the testing epochs. We also fair our model with a reserved number of open-source public datasets for observing the performance compatibility. Finally, based on the analyzed data, we further apply the Bayesian regression technique to predict the range of cases, deaths, and recoveries for a fixed period within a common plane.

1.2 Motivation

We are keen to leverage the large amount of data on Coronavirus generated from Twitter to find a meaningful data insight for analysis, visualization, and evaluating the sentiments. Although Twitter data analysis in the premise of sentiment analysis is nothing new and has been under the spotlight for some years now [24–26], tackling a large-scale event as it unfolds is quite new and challenging. This is primarily the exact novelty that we propose to contribute to the current work.

With that being said, there are further selectively explored areas that we try to inspect through our work. For instance, it is not fruitful to use a shallow convolutional network for fairly large data processing, rather deep architecture involvements can be studied more. Keeping it in mind, we build a deep CNN architecture capable of taking sentiment labeled word vectors as channelized input. Also, a traditional problem with intermediate deep learning networks is data overfitting, which eventually leads to dropout. We aim to address this issue from the perspective of the learning rate, and the reduction of input data vectors. Another common concern is the selective but best feature selection within the CNN max-pooling layers, for which the kernel size is kept smaller; to provide a limited matrix position availability for only the best features from a text set to come through. But for dense feature-rich data, this approach could fall short, eradicating even quality features. These are some frequent topic of limitations that we would like to overcome in our present work.

1.3 Contribution

The novel contributions of this paper are:

- We show a detailed feature study of a newly developed Twitter corpus consisting of only Covid-19 tweets.
- We present a gradient scale comparison contrasting between the major affected countries.
- We demonstrate a detailed convolutional architecture deployed for a successful train and validation case of sentiment accuracy evaluation.
- Finally, we simulate a model based on Bayesian regression from the refined polarity tweets, which helps us to make a predictive analysis on future Covid-19 case reports and recovery rates on divergence with the current scenario.

The organization of the paper is as follows: In Sect. 2, we discuss some relevant works combining the applications of deep neural networks and recent trends in sentiment analysis. Next, we discuss our proposed methodology in Sect. 3 with an entire outline covering our work. Following, we elaborate on developing the Twitter corpus and extracting the features

from it in Sect. 4. In Sect. 5, we labelize our tweets with polarity labeling and proceed with the current trend analysis from the data gathered. Next, we demonstrate our convolutional network architecture with layer-wise detailing in Sect. 6. In Sect. 7, we discuss about the experimental setup, hyperparameter tuning and analytical details of the model. In Sect. 8, we represent the evaluation of training and test case validation. With that, we show the comparative performance comparisons between our obtained results with some of the previous gold standard works on the same public corpora. Here we also compare our performance with the previous state-of-the-art works on large Twitter corpus and customer review data. In Sect. 10, we perform the predictive analysis from the sentiment evaluated data. Section 11 concludes the paper by discussing the scopes for future applications and improvements.

2 Related works

With the advent of processing capabilities and computing resources, neural networks have been ever-so-popular, especially in the likes of image classification and processing, handwriting recognition, text, and sentiment analysis. From the short text summarization and analysis perspective, it is no doubt in saying that a particular number of neural network varieties are experimented with, and proven over time. To name a few, recurrent neural networks with long-short term memory (RNN-LSTM) [27], bi-directional long-short term memory (Bi-LSTM) [28], and the convolutional neural networks (CNN) [29]. While the problem with classic rule-based machine learning algorithms is, though they are faster to learn [30], they are shallow, hence the learning curve never really takes off with the time flow. Hence, harnessing the modern GPU processing powers, more and more deep neural networks are being deployed for the aforesaid tasks [31].

2.1 Application of convolutional networks

Traditionally, it was thought that convolutional networks are superior in image classification only [32]. The convolutional layers can be used singularly or stacked upon each other to form a dense layer (which is by the way stated as the “deep” layer) to filter out the cluster data points (or information) as matrix forms from the original image [33]. The more convolutional layers are there, the more fine-grained filtering is possible [34]. Similarly, the max-pooling technique is used to extract out the information to build the custom classifier of the model [35]. While this “information retrieval and stacking” of images could lead to successful training and validation phase upon learning for a fairly large set of

image sets, this approach was not trialed in the context of sentiment analysis.

Meanwhile, though the RNN-LSTM models were churning out quite fair and accurate results for the textual experiments [36], one major problem with this model is their backpropagation feedback-per cycle system [37]. Though the learning errors and any of the learning incapability are made feedback with each cycle of the vectorization and classified text, it takes a significantly larger amount of time [38]. On the other hand, convolutional networks stack the learning errors with the successful learning accuracy in each convolutional layer, ultimately these erroneous sets are discarded to develop the noise-free or error-free prediction [39]. There is a setback that the CNN model will keep an eye for the local optimum instead of global accuracy [40], but it can be fixed with a simple classifier adjoined with every convolutional layer. This technique not only reduces the time but also brings out the improvement in the overall accuracy-based testing and prediction approach [34].

Since late 2017, the Deepfake videos are making quite a stir on the internet, as well as in the deep learning research community. The word “seeing is believing” really falls short here since the celebrity face swapped videos are really hard to detect for human eyes to verify the originality of the video. The concept behind this approach is to collect a large number of pictures for both persons who are intended to be swapped. Sequentially an encoder based on existing Tensorflow⁷ or Keras⁸ library is developed to encode all these pictures using a deep convolutional network. Then a decoder is used to reconstruct the image.⁹ Then it uses another convolution layer to upsample the image. The decoder continues the upsampling with further convolution layers until it reconstructs the 64×64 images back. To upsample the spatial dimension, say from 16×16 to 32×32 , we use a convolution filter ($3 \times 3 \times 256 \times 512$ filters) to map the (16, 16, 256) layer into (16, 16, 512). Then we reshape it to (32, 32, 128).

While the mechanism sounds impressive for the modeling and real-time “face-mask” creating the ability of the CNN model, this experiment has some serious downgrades. The videos generated by the FakeApp or Deepfake also makes them a target tool for engaging in hate speech, hoax, rumor spreading, false information channeling, and so on. The problem is now even more serious than ever, as we are facing a global pandemic situation caused by Covid-19, which is already triggering massive social panic and fear. A group of researchers has addressed this by contributing to detecting fake videos [41]. They proposed a two-phase analytical composition of a CNN to extract features frame by frame

⁷ <https://www.tensorflow.org/resources/tools>.

⁸ <https://keras.io/api/>.

⁹ https://medium.com/@jonathan_hui.

followed by a temporally responsible RNN network to capture temporal inconsistencies between frames introduced by the face-swapping process, which is, in other words, the flicker generated in an ongoing video. Then they deployed a collection of 600 videos to evaluate the proposed method. They also used half number of videos being Deepfakes collected from multiple video sources. Next, they represented experimentally the effectiveness of their experiment. It allowed them to detect if a suspect video is a Deepfake manipulation with 94% more accuracy than a random detector baseline in a balanced setting.

Already Collobert and Weston showcased that even a simple deep learning framework outperforms most state-of-the-art experiments in various NLP tasks [41], named-entity recognition (NER), semantic role labeling (SRL), and POS tagging, to name a few. Since then, numerous complex deep learning-based algorithms have been proposed to solve difficult NLP tasks. One essential thing the deep learning models need for training is the word vectorization. Following that working mechanism, popularization of word vectors, and the sheer capability to represent words in an evenly distributed space across the layering of a deep neural network, the requirement occurred for an effective feature selection tool that extracts higher-level contextual or aspect-based features from constituting words or n-grams. These sets of features could then be used for specialized tasks such as sentiment analysis, text summarization, automatic encoding or decoding, machine translation, and question answering (QA) via digital chatbots [42].

Another use of CNNs for sentence modeling is elaborately demonstrated by Er et al. [43]. They demonstrated multi-task learning characteristics to output multiple prediction spaces for POS tagging, chunking, named entity tagging, semantically-similar words finding, which leads to the automatic modeling of natural English like language. For transforming each word into a vector of user-defined dimensions, a tabular form was represented.

For NLP related tasks through the CNNs, the models primarily undergo two mechanisms; sentence modeling and window approach [44]. For a given sentence, a word embedding agent is represented for the particular word carrying weight in the sentence. Simultaneously, the dimension of the word embedding is also measured with another coefficient. If a sentence has n number of words, the sentence can be represented as an embedding matrix that has a layer of vector weights of the words depicting a similar sentiment or semantic categorical sentence as an input to the discussed CNN framework. In a typical CNN model used for the same, the number of convolution filters, each containing different widths slide over the entire word embedding matrix that we have discussed before. Each of these filters fetches a specific pattern of n-gram without stop words from a given sentence. In the same context, a window approach mechanism is used,

which assumes that the tag prediction of a word primarily depending on its immediate next neighboring word. In this approach, for each word, a fixed-size window composed for a sentence itself is stacked, and a particular portion of the sentence within the window is taken for consideration. A CNN filter is applied to this focused portion and the prediction that can be extracted for the word.

2.2 Recent trends in sentiment analysis

Sentiment analysis is an extremely important tool to analyze and predict the key events and trends with ongoing time. In this context, Alamoodi et al. have shown a great application in fighting against Covid-19 [45]. They aimed to develop a coherent database of all literature on similar medical and pandemic events from the past ten years that has been published with major online research paper databases. They sorted and organized the repository with taxonomy or keyword-based approach, cumulating all the major applications of dedicated sentiment analysis works regarding the topic. When several speakers express their emotion on various topics by social media, even then they shift their mood from the initial emotional state in the time of the conversation. In this context, it is a challenging task to identify the exact emotion of a particular person. Recent work addressed the issues by their ensemble deep learning model [46]. They composed an ensemble learning approach from the sub deep learning models, and train the respective systems with some sentiment weightage vectors as well as emoji with respective polarity ratings. They observed that the ensemble model performs better than their standalone counterparts in tri-emotional expression (angry, happy, sad) detection. Another work from 2018 addresses the aspect based on targeted sentiments [47]. While the targeted sentiment can vary depending on the context, even within a single sentence; the authors proposed a hierarchical attention network for detecting the specified targets. They further incorporated common-sense knowledge for aspect-based sentiment with the proposed LSTM model. For their findings, the common-sense knowledge infusion within the LSTM model efficiently helps to detect the sentiment spaces of a sentence, with overall better sentiment classification. Alongside traditional sentiment analysis from texts, other different multimedia sources can also play a vital role in successful sentiment detection. These sources are mainly audio, video, and speech in the form of text. A group of researchers used a contextual recurrent network trained with utterance vectors of the same dimensionality [48]. They infused the modalities within the fully connected layers, for retaining the contextualities of the sentiment vector. Here, the different inputs for training originate from a different modal source, i.e., audio, video, and text. Their trimodal benchmark analysis reveals the highest accuracy obtained of 80%.

Considering the recent pre-training vectors and attributes of audio and video analysis, this concept can likely grow for betterment in near future. As we have already mentioned, several distinct deep learning models have shown significant potency in NLP tasks in recent years, specifically for sentiment analysis. A work from 2020 organizes a detailed survey on today's widely accepted deep learning models deployed for sentiment analysis [49]. They have discussed in detail the broad application areas of sentiment analysis with practical case scenarios. They also enlisted a plethora of journals from which they have selected papers for the review study process. They also demonstrated the yearly vicissitude of the research publications on this area, as well as which models gained popularity for the said experimentations, and which approaches lost popularity gradually with time. Alongside that, the authors also presented a detailed elaboration on which deep learning approaches are most suited and used for which subtopic of sentiment analysis research.

It is already demonstrated by Howard and Ruder that developing an accurate deep convolutional network needs the fine-tuned parameters [50]. The group of researchers proposed a three-step to train the deep learning model for sentiment classification. It was concisely contributed by:

- (1) word embeddings, which are initialized using a neural language model to train on a large unsupervised collection of tweets,
- (2) the convolutional neural network to even further refine the embeddings on a large distant supervised corpus,
- (3) the previously done fine-tuned parameters of the network obtained at the previous phases are then used to initialize the network that is followingly trained on a supervised corpus.

One of the frontiers of the word embedding model introduced was the CBOW model [51]. This model is capable of fitting a target word on a prediction basis into any given sentence based on the concept of conditional probability. The sophisticated mechanism behind the word level embedding is to mapping n number of words in an arbitrary dimensional space, while the words conveying similar contextual semantic meaning are the neighbors of those words [52]. Though the closeness of the words implies different contextual meanings in cross-lingual applications, and for that several vector spaces can be applied concurrently with each other for forming a semantic combination. Also, for cross-lingual vocabulary, two similar identical words can lead up to way different meanings. For that, distinguished vector weights should be applied to each of the words in a multilingual vocabulary corpus. Since the vectorized inputs of words are generated using word embeddings and fed into the neural networks [53], hence either the corpus can directly be processed to produce the word to its corresponding vector, or the corpus

can be made sentiment labeled first with any baseline algorithm, then the word annotations with appropriate semantic understanding leads to more accurate vectorization.

3 Proposed methodology

3.1 Data collection

Data (here tweets) collection is the foundation of our work. Our experiment starts with a large-scale tweets collection. We then aim for feature extraction for our tweets using a binary vector count. Feature expression fragments words from short phases or texts to represent distinct categorical properties. We specifically convert text to numeric feature through the aforesaid binary encoding. In this scheme, we create a vocabulary by looking at each distinct word in the complete data corpus. For each tweet, the output of this particular mechanism will be a vector of binary size n , where n will be the total number of words that will be produced as our cumulative vocabulary. Initially, all entries in the vector will be 0. If the word in the given document exists in the vocabulary then the vector element at that position is set to 1.

3.2 Polarity rating

Next, we split the whole corpus into a 70:30 ratio for training and testing. We intend to load the testing data and load it in a quantitative per epoch manner while the testing phase. For the training data labeling, we use the logistic regression approach as our baseline algorithm. The only logic behind this approach is logistic regression works best when it is used for high dimensional sparse data. Our data is no exception, considering the sheer volume and parameters maintained to stream the live tweets. Since the data is streamed uniformly with equivalent parameters, we use the linear mechanism to classify the polarity of the training tweets. The linear regression algorithm for collecting feature f_i from the previous vectorization for the prediction $f(x)$ holds:

$$\phi f_i = \beta_i(x_i - E[x_i]) \quad (1)$$

3.3 Gradient scaling of cases

Following the polarization of tweets, we opt for the trend comparison on the number of reported cases and the gradient number of these cases from the tweets to find the insight for future predictions on the analyzed data in our next phases. For finding the trend gradients from our tweet corpus parameters, we map them in an equivalent comparable space, which is $G \rightarrow R_{c \times d}$, where g is the number of gradients for

comparison, mapped with the combination of countries and date of the tweets. Now summing up the countries in conjunction for collective trend comparison:

$$\sum f : G = [R_1, R_2, \dots, R_n] \in R \tag{2}$$

Considering every appearance of g within the list is the cases of each country, which as a whole belongs to the R space, and mapped into the output function of R altogether.

3.4 Comparative and predictive analysis

These tweets rounded up for trend analysis with sentiment polarity are then feed to the batchwise $r \times c$ representation in the first layer of the convolution network. On the other hand, we load the *GloVe*¹⁰ vectors from the existing Keras module. Simultaneously we input our test data while the convolutional layer is processed. Then the max-pooling strides are used to combining the feature mapping. Finally, the fully connected output layers predict the batch-by-batch loss and accuracy for both the training and testing data.

After getting the accuracy predicted and evaluated data, we perform the comparative analysis with other state-of-art systems of previous experiments and address the noticeable performance efficiency of our system. Now, cooperatively, also from this data, we perform the Bayesian regression-based predictive analysis for the future case reports and increment in data volume for Coronavirus. The regression principle operates on the Bayesian theorem of a prior condition and likelihood scenario [54]. Figure 1 shows the hierarchical process diagram of our proposed methodology.

4 Dataset and experiments

Twitter is probably the most sophisticated and popular platform for live streaming textual data. Mostly because of its “micro-blogging” nature and short targeted texts, the data which are scrolled from Twitter is full of raw data which can further be processed into insightful information. While Twitter API only lets developers to stream 1–2% number of total tweets per hour, the scale of the Twitter user’s community is large enough to gather a large source of the corpus on any given topic. From 7th November 2017, Twitter expanded its character limits to 280 characters per tweet. Hence the target probability of tweet collection is now higher than ever before.

For March and April 2020, we collected 600 k or 6 lakh tweets exclusively on Coronavirus or Covid-19. The tweets were extracted w.r.t to the parameters like username, date, time, retweet, and country or location of the tweet. Also,

we collected parallel tweets after Indian Prime Minister Modi’s live speech on 24.03.2020 and US President Donald Trump’s press briefing on 02.04.2020, to gain valuable mass opinion insight on the given topic. We scroll our tweets based on certain parameters for scaling, naming tweet id, the original tweet body, retweet count, location or region, date, and time. In correspondence with the target topic of our objective tweets, we filter the tweets as for parameter cumulation:

$$\sum T_p = T_{[p1 + p2 + p3 + p4 + p5]}$$

While for topic channeling:

$$\sum T_t = [\text{topic_word}_1, \dots, \text{topic_word}_n]$$

The objective here is to find an optimal combination of both the formulations, i.e., the model acts as a funnel to stream down only the topic focused tweets in English irrespective of the date parameter.

Now, following the approach of a combined linear equation from both the formulations,

$$\begin{aligned} \sum T_c(T_p + T_t) = & [T_{p1, \text{topic_word}_1} + T_{p2, \text{topic_word}_1} + T_{p3, \text{topic_word}_1} \dots] \\ & + [T_{p1, \text{topic_word}_2} + T_{p2, \text{topic_word}_2} + T_{p3, \text{topic_word}_2} \dots] \\ & + [T_{p1, \text{topic_word}_3} + T_{p2, \text{topic_word}_3} + T_{p3, \text{topic_word}_3} \dots] \end{aligned} \tag{3}$$

We can depict that the system forms a linear additive matrix of tweet parameters and topic words combined for funneling any type of tweet on the Coronavirus topic. Here T_c is the total tweets collected, T_p is the tweet parameters, and T_t is the tweet topics strictly on coronavirus depending on the hashtags and keywords. Based on our tweet filters, in Fig. 2 we plot the cluster regions from where the highest number of tweets were collected.

While we couldn’t find any geolocations from China despite Wuhan being the first origin and epicenter of Coronavirus. It can lead to two probable aspects: One is the Chinese people usually use Mandarin typing even on social media platforms, while we only stream tweets that are in English. Also, the second reason being Twitter is officially blocked in China; people there majorly use Weibo instead. Also, since April 2020, the reported cases in China fell significantly¹¹ in contrast to the other parts of Southeast Asia, Europe, and North America. Hence, we do not take the total case report insights from China for further predictive analysis due to these limitations as well as for inspecting the scenario of more severely affected regions.

¹⁰ <https://nlp.stanford.edu/projects/glove/>.

¹¹ <https://www.coronatracker.com/country/cn>.

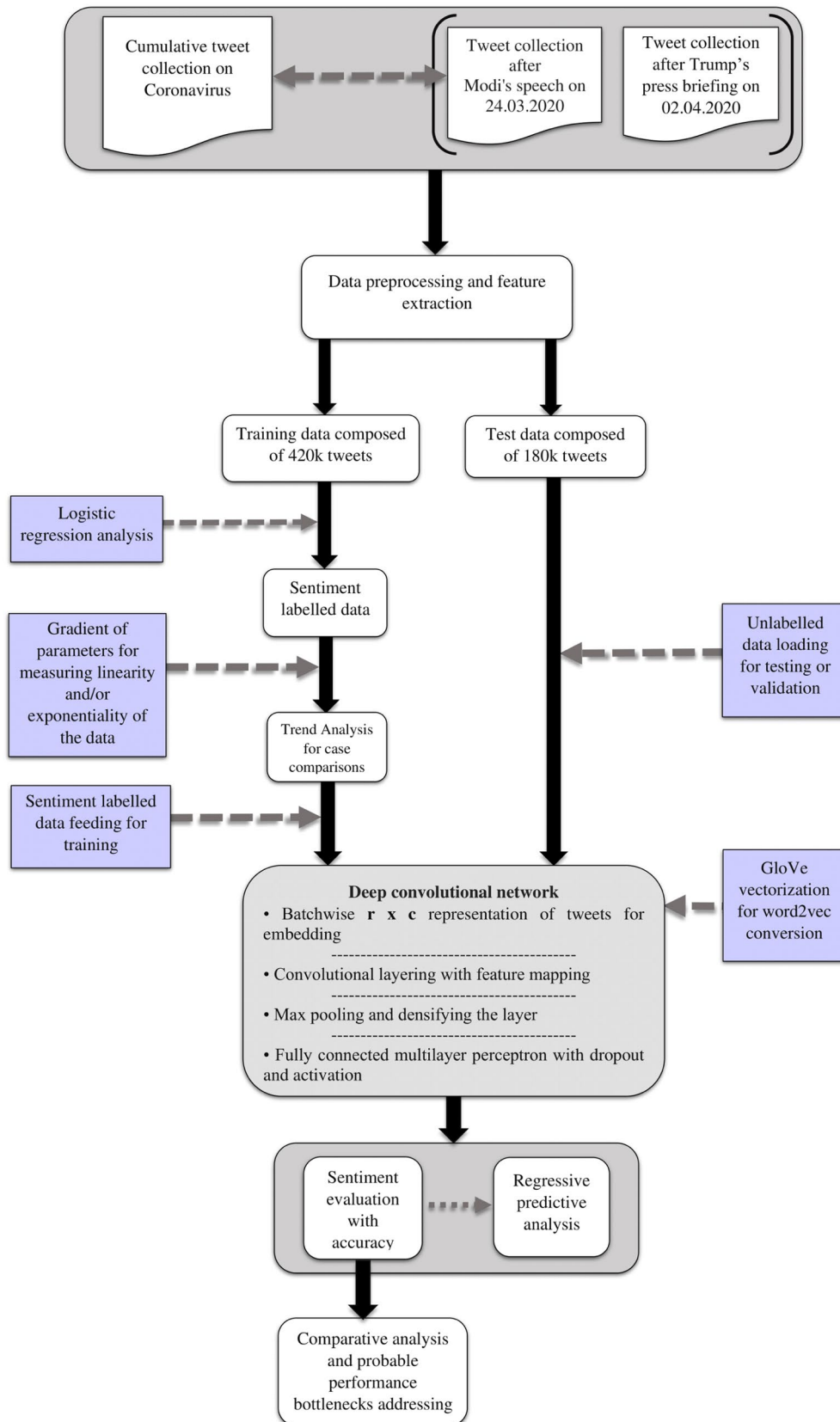
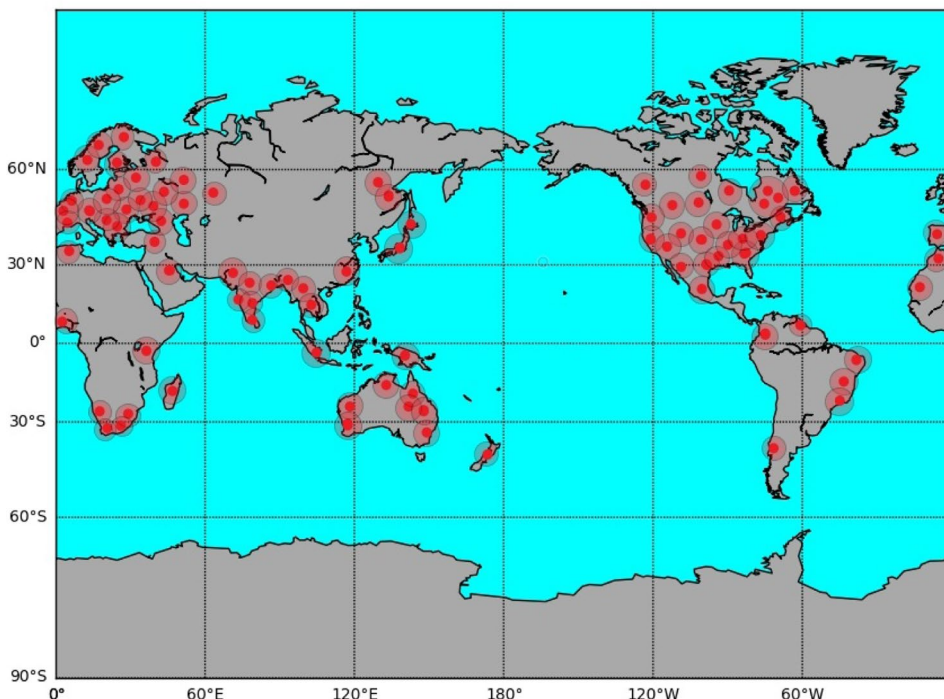


Fig. 1 Hierarchical phases of the proposed methodology

Fig. 2 Coronavirus tweet regions plotted globally



4.1 Feature extraction

Following the data collection, we focus on feature extraction from our corpus. Where we churn out the words of the text which represent discrete and categorical features. We make the list of unique words in the data corpus. Then we can represent each sentence of a tweet or the whole tweet as a vector with each word represented as 1 for being present and 0 for absent from the vocabulary. Simultaneously, we representation the count of appearances for the number of times each word appears in the collective tweet data.

For finding a word with a high appearance and with high impact on the analysis of the data, if V is the vocabulary contained from the data preprocessing and W is the number of the words, and D is the total data volume:

$$\{V \rightarrow W_{(i...n)}\} \approx DT(T_c(T_p + T_l)), \quad \forall W \in DT \tag{4}$$

The approximation here leads to the simulation of cases where a word may contain a frequent occurrence, but low impact and low unique perspective from the unique nature of the word itself. These words will be discarded. Others are formulated in the term frequency distribution for the.

remaining unique words appeared for t number of times with the total numbers of column entries as C with n number of columns:

$$TF = \frac{W(i \dots n) \times t}{C_n} \tag{5}$$

Figure 3 helps us to visualize such words with the number of occurrences in the closed spe of entire covered tweets from our corpus.

5 Sentiment labelling

For generating polarity labels for the 70% training data, we use a logistic regression algorithm for our approach. With the regular tweets on Coronavirus, we also analyzed the sentiments of the tweets which we streamed just after Modi’s live speech and Trump’s press briefing on Covid-19, to capture how the impact of the world leaders’ statements can affect the polarity of tweeting people towards positivity or negativity.

Since our data is dense with one too many parametric notions for each tweet, we apply logistic regression here. We use the linear mechanism to classify the polarity of the training tweets. The regression algorithm for collecting feature f_i from the previous vectorization for the prediction $f(x)$ holds, then Eq. (1) holds.

For our polarity notation, either the classes are 1 for positive and 0 for negative. Now we have a binary output variable S , which is the expression of both the polarity generated values. We want to model the conditional probability $Pr(S=1|0)$; where the parameters of our tweets help to be deterministic for estimating the maximum likelihood. Like, a person tweeting from a heavily ill-affected country is not likely to tweet about any good views on Covid19, and our

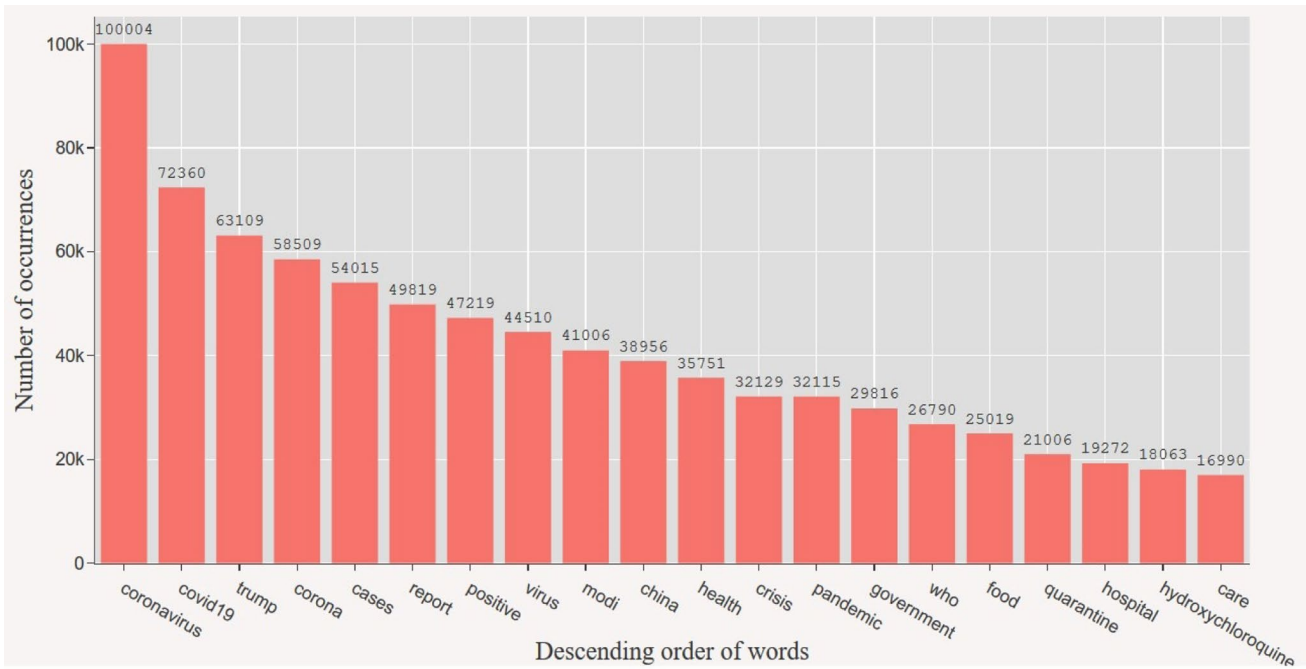


Fig. 3 Most used words within our cumulated tweet corpora

parameter and topic combination from the previous section will ensure that while tweet filtering.

Now, if $p(S)$ be a linear function of S , every keyword of a tweet add or subtract to the probability of being positive or negative. The output must be binary, and the higher probabilistic combination of positive or negative cumulation will define the polarity tag of a tweet. Since linear functions are unbounded, hence for the modification of the function of S , we convert it to \log_S scale for comprising it within a bounded region for the logistic transformation of the Sentiment value. Now the range can hop between $0 \leq \log_S \leq 1$. We propose this as the linear function of S with a closed ranging value.

Hence, we form the regression model for sentiment as:

$$\log \frac{p(S)}{0 \leq \log_S \leq 1} = \Pr(S_0 \pm S_1) \tag{6}$$

More formally,

$$p(S \rightarrow W(i \dots n) \approx DT) = \frac{e^{\Pr(S_0 \pm S_1)}}{1 + e^{\Pr(S_0 \pm S_1)}} \tag{7}$$

Here, the linearity of S is directly relevant to the word vocabulary directly generated from the tweets during feature extraction previously. Now solving this equation reveals that as the bounded region is kept within 0–1, hence for minimization of the misclassification sentiment rate, the logistic regression equation gives us a linear classifier. In our context, the pattern of the classifier can be easily identified from Eq. (7) as the Sigmoid classifier. The

inclusion of the Sigmoid classifier fits with our data since a Sigmoid can separate decision boundaries learning from the feature vectors. The decision boundary of the classifier separating the two predicted classes leads to the sentiment flavor of the tweet. Mathematically, it can be stated as:

To concise from Eq. (7),

$$S = \frac{1}{1 + e^{-\Pr(S_0 + S_1)}} \tag{8}$$

where the combination between both the positive and negative sentiment is compared based on conditional probability and then the decision boundary determines the linearly classified boundary value between both the regions.

Logistic regression clarifies where the boundary between the classes is, and also depicts the class probabilities depending on the separability from the boundary, and that the part towards either the terminal ends (0 and 1) of a sentiment more rapidly when logistic scale probability is higher. We churn out the polarity from the rule-based logistic regression with a Sigmoid classifier. This is why the discussed approach is a good suite for dense text with high parameters but short summarizations.

5.1 Case trends with gradient parameters

Statistically, the gradient of a number is used to find the partial derivative of a scalar unit. Similarly, several

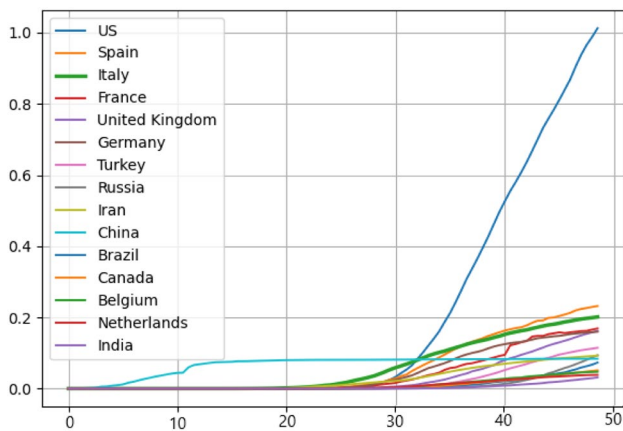


Fig. 4 The number of case vicissitudes among the most affected countries

gradually increasing or decreasing pivot values along a slope can determine the linearity path of the scalar unit.

Incorporating the country as the main parameter of a tweeting person, and differentiating his or her sentiments towards the cases of Coronavirus in that particular country, the countries can be summarised as the function for gradient for the number of reported cases. Now as we both have our tweet parameters and sentiment values for each tweet, the differentiation to find the gradient can be implied to the functions S of a scalar variable $G \rightarrow R_{c \times d}$ as previously discussed.

With that, as we have mentioned previously, we also consider other general parameters such as the date of the tweet where the function G combines with another parameter, and

this second parameter works as another deafferenting factor. It can be summed up as:

From Eq. (2),

$$\sum f: G = [R_{1c \times d}, R_{2c \times d}, \dots, R_{nc \times d}] \in R$$

$$\text{or, } \sum S: G = [R_1, R_2, \dots, R_n]_{c \times d} \tag{9}$$

Now, without looking for even higher order partial derivatives following the mechanism of gradients [55], where the function itself is dependent on more variables, we rather find the gradient of the function S concerning c and d by varying one variable at a time and keeping the others constant. It implies that when we are looking at tweets from various countries at first to retrieve information about new Coronavirus cases, we are keeping the date parameter as secondary. Simply because, if any country does not report new cases while our tweet collection, then adding it on the comparison list is nullified. Whenever we come across new cases, then the second parameter, i.e., date counter starts for case tracking. This is how the gradient numbers are then collected into a separate file for comparison. At first, Fig. 4 shows a simple timeline visualization about how the number of cases rose in the most affected fifteen countries. From Fig. 4, it is easily depictable the rapid growth of Coronavirus cases in the US from 15th April 2020 onwards, which is the real scenario. Since we have not considered the mortality rates as our parameters, hence compared to the US’s 1 Million reported cases, Italy and Spain are moderately closer to the x -axis, as the number of reported cases started to decline over there from the end of April.

Now, from the general comparison of the previous figure, we focus to compare the gradient number of case reports

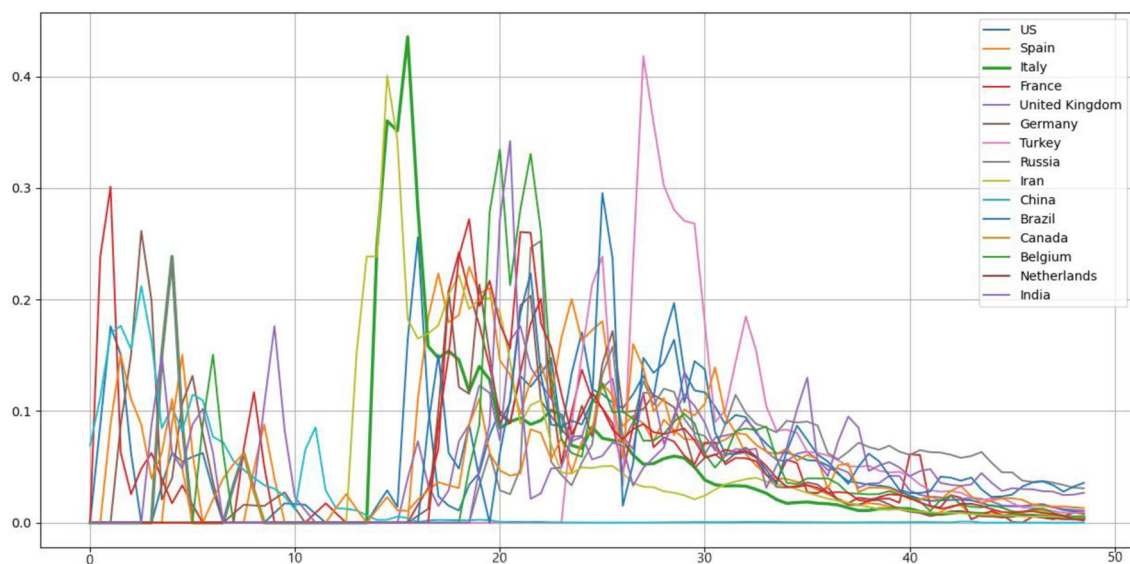


Fig. 5 The gradient of numbers cases comparison among the most affected countries

for the top fifteen most hit countries as before. This analysis certainly elaborates on the abnormal incline and decline of the cases reported of these countries when compared to each other.

From Fig. 5, two decisive issues can be drawn:

- The highest number of cases were reported in Italy on a single day, between 25th March and 15th April 2020. No other country witnessed such a high surge in the number of cases so rapidly. This was accurate as per the reports from several international media.¹²
- Some countries were early affected and rapidly climbed to a peak in the number of affected people, while the other countries are gradually getting affected now and slowly rising in reported case numbers.

6 Proposed architecture of CNN

Classically the convolutional network is based on the principle of multilayer perceptron, correlated and designed to recognize two or more dimensional shapes with a dedicated assignment to translation and vector representations. Convolution is an operation on two functions with a real-valued argument for identifying the sentiment from the training set, then evaluating the test case tweets, and thereafter successfully classifying them with a per epoch cyclic loss and accurate prediction.

Traditional neural network layers use a dense connection among the hidden neurons and with the bias in the final layer as the output unit. As a result, every incoming connection from a neuron carrying a particular weight interacts with every other neuron in the inner-layers of the networks. However, convolutional networks, typically maintain sparse interactions among the hidden layer neurons to avoid overfitting the data. This mechanism leads to dropout. While we are using a 6-layered convolutional network, we only maintain dropout in the $n-4$ and $n-3$ levels of the network. Also, we make the max-pooling strides as well as the kernel smaller than the input grid, to specifically compare the word vectors and semantics of the input words from the tweets with that with a standard external word vectorization algorithm. It helps us to detect only the meaningful features of the statements not only at a sentence or phrase level but from the aspect level. The network learns in a supervised manner meaning the feature mapping strides feature out the constraints from the sentiment classified tweets. The main features which are must be retrieved here are the singular polarity nature of the training tweets, and the gradient

parameters mentioned previously. We show the architecture of our convolutional neural network in Fig. 6.

Our network structure includes the following phases and analytical details of training in the forms of the following sequential mechanisms:

6.1 Step 1: initiation of neurons with feature extraction

We propose our primary neuron accepts a text of $W \times D$ size. Here W is the width of the tweet, while D is a dimension of putting the vectors into their respective positions in the first grid. There are $r \times c$ numbers of grid-like this in the first input layer. Now, for the hyperparameters of filtering, we maintain a K number of filters, which are anyway our parameters mentioned before. When we feed the linear queue of tweets for training purposes, then the input formatting can be represented as a queue of tweets $W \in K^{p1, p2, S}$, where $p1$, $p2$ are the parameter sets, and S is the sentiment.

While the dimension would be:

$$D \times W \in (K^{p1, p2, S} \times K^F)$$

where F represents spatial filtering. After adding the bias, as $b \in K$, the input layer computes:

$$IP = (D \times W)K^{(p1, p2, S)+F} + b$$

$$\text{or, } IP = \sum_{K=1 \dots n}^S D[((p1, p2 \cdot S) + F)] * W[((p1, p2 \cdot S) + F)] + b \quad (10)$$

This represents the convolution for each tweet, i.e., whenever a tweet is entered for vector as well as semantic decoding.

6.2 Step 2: convolutional layer with feature mapping

Each convolutional layer in the connection is composed of multiple feature mapping grids, as in our case an 8-grid of 64×64 size. Here, each grid develops a plane that contains independent nodes that share the same combination of synaptic weights. The benefits of this structural combination allow us to shift the convolution grids to flexible invariant positions, with a small kernel and connected with a Softmax and ReLU function [56]. We select them because of the faster training time and monotonic nature. This characteristic promptly affects the negative to positive curbing and maintains stability while training. Also, the limited grid restricts the parameter independence, making a mandate for the same parameters for extracting features from all the variety of tweets. With that, we load the GloVe vector representation for words. This allows us to map our previously sentiment embedded words by

¹² <https://news.sky.com/story/coronavirus-italy-reports-919-virus-deaths-in-one-day-highest-number-so-far-11964712>.

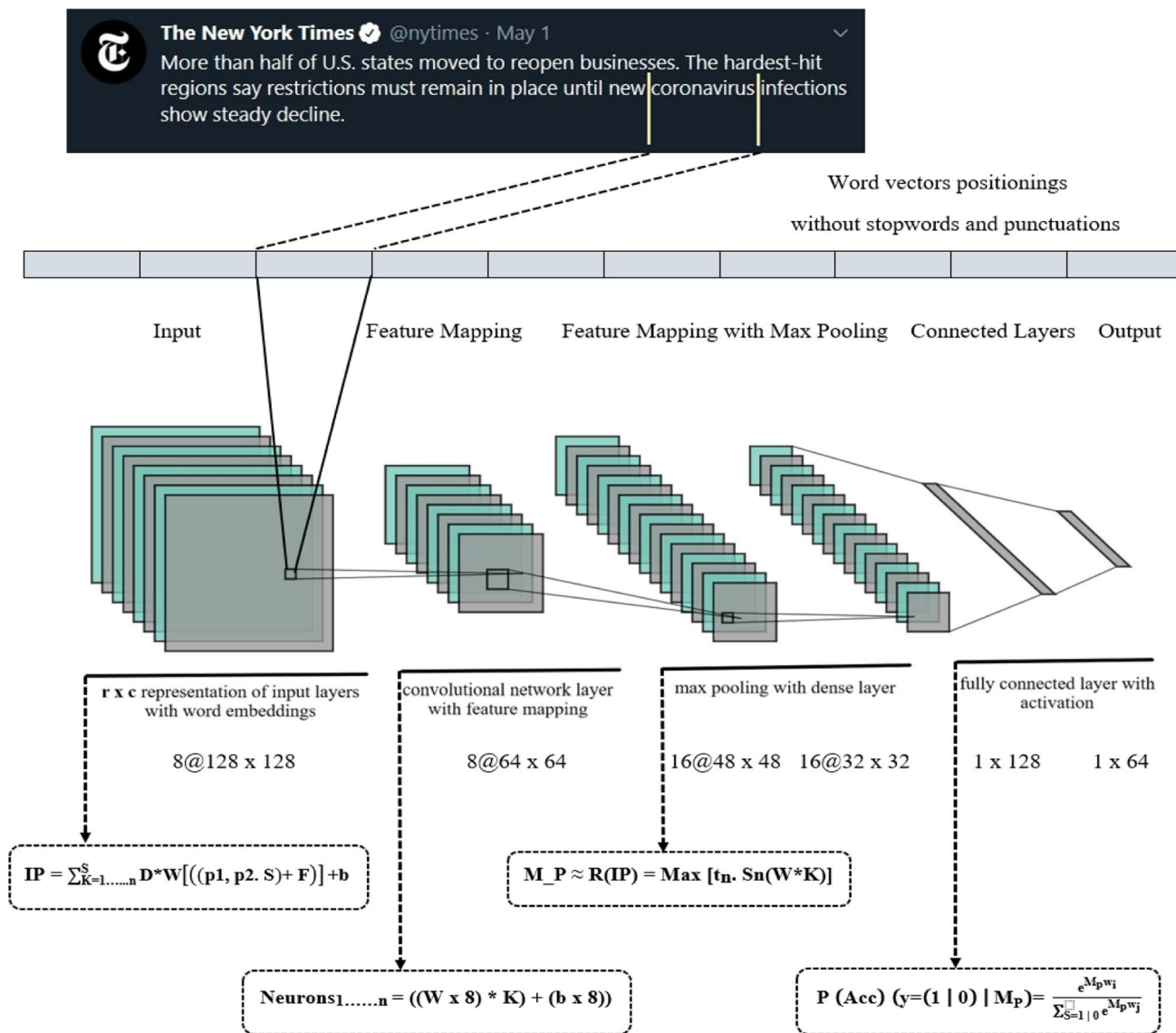


Fig. 6 Architectural representation of the proposed convolutional network

logistic regression to getting compared with the standard vector corpus, and maintain the vector-balance analogous feature. This layer can alternatively repeat after a limited cyclic number of layers.

Now, we can show that if a convolutional feature mapping layer contains N number of neurons or nodes, the interconnected width and bias among them are the subset of K filters. Hence,

$$(W, b) * Gr_{1...n} = \subseteq K$$

To be represented from neurons end connected with the previous layer:

$$Neurons_{1...n} = ((W \times Gr) * K) + (b \times Gr)$$

or, for our experimentation,

$$Neurons_{1...n} = ((W \times 8) * K) + (b \times 8) \tag{11}$$

where Gr is the number of grids, and b is the bias.

6.3 Step 3: max pooling with dense layer

The max-pooling with dense layer is responsible for achieving subsampling along with local averaging of features, and the grid sizes are further reduced. We are using a 16-grid set but with 48 x 48 size. The shrinking in size with reduced height and width helps to downsample the features from the previous layer. It restricts any distortion of the outputs from the previous layer.

At first, the subsampling operation takes place to scale down the features:

$$\begin{aligned} R(IP) \in \text{Neurons}_{1,1} &= (((W \times 8) * K) + (b \times 8)), \\ \text{Neurons}_{1,2} &= (((W \times 8) * K) + (b \times 8)), \\ \text{Neurons}_{1,3} &= (((W \times 8) * K) + (b \times 8)) \dots \end{aligned}$$

where values are selected as $\text{Neurons}_1 \geq \text{Neurons}_2$, $\text{Neurons}_2 \geq \text{Neurons}_3$, and so on.

Here the respective parameters contained in the previous layer neurons are reflected as outputs in this layer, and the downscaling of the features based on the richer feature set of values. Then the averaging function takes place to evenly distribute the features scaled:

$$R(IP) = \frac{t \cdot S_1 + t \cdot S_2 + \dots + t \cdot S_n}{S_1 \cdot (W * K) + S_2 \cdot (W * K) + \dots + S_n \cdot (W * K)} [\text{Neurons}(S \cdot (W * K))] \quad (12)$$

Finally, the max-pooling function computes the maximum linearly distributed feature set captured from the previous steps, and proceed towards the global pooling. The closely selected pooling window extracts the highest value within that window, allowing it to leverage the best possible feature collection for each tweet with its respective sentiment polarity.

Mathematically,

$$M_P \approx R(IP) = \text{Max} [t_1 \cdot S_1(W * K) + t_2 \cdot S_2(W * K) + t_3 \cdot S_3(W * K) + \dots + t_n \cdot S_n(W * K)] \quad (13)$$

These refined features then successfully shift towards the fully connected layer or layer with dropouts for final training evaluation and prediction accuracy.

6.4 Step 4: fully connected layers

At the end of our convolutional network, the output of our previous pooling layer enters as the input to the so fully connected layer. There can be multiple fully connected layers gradually, with a decrement in size. We are using two fully connected layers with sizes respectively 1×128 and 1×64 . These layers are termed as fully connected since every neuron in a fully connected layer is connected to every neuron of the second fully connected layer.

These connected layers perform classifications based on the features extracted by the previous layers. Now, the classic fully connected layer consists of a softmax function, along with the parameters like loss, optimizer, and metrics. The softmax function here acts like a normalizer exponential function, which produces a discrete probability. Also, the function is more efficient, when it is combined with the parameters that we have just discussed.

Based on the vector input M_P from the previous layer, and w_i is the weight vector, the predicted probability will either be 1 or 0.

Hence,

$$P(\text{Acc})(y = (1|0)|M_p) = \frac{e^{M_p \cdot w_i}}{\sum S = 1 | e^{M_{pw_i}}} \quad (14)$$

The combination churns out the output which outputs an identifying probability of non-trained test texts, along with predicting it with the identified polarity with a certain extent of accuracy. The more the extent is, the better the whole network is operating altogether. The process of identifying sentiment polarity can also be stated as the classification labels that the model tries to predict successfully.

7 Experimental settings

7.1 Setup

We use Google Colab¹³ as our platform for experimentation. It is a cloud-based Python notebook platform primarily aimed at Python versions over 3. Also, it introduces a native feature named a hardware accelerator for faster execution time, but with a limited resource threshold. Colab provides the options to choose from the dedicated graphical processor environment execution based on Tesla K80 GPUs or Google's tensor processing units, developed for parallel neural computations simultaneously. While users can select any of the previously mentioned options, we select none, because there is a significant delay in resource granting for adjusting usage limits and hardware availability if the GPU or TPU accelerator is selected. We also connect to a hosted runtime with sufficient but abstract RAM and disk availability, as Google does not share the exact figurative information with the users.

7.2 Hyperparameters tuning

Our model tuning contains a word embedding dimension of 400. We are using 500 hidden layers for each of the deep convolutional networks. The dense layering holds the activation classifiers. We use adam optimizer with categorical cross-entropy for binary categorical classification of sentiment within the training phase. We then initialize it with the loss function to evaluate the training loss. We keep the learning rate of the optimizer as 0.01. The dropout rate for avoiding overfitting is kept at 0.6 for the vectorization layer. We fit the model within the padded vector–matrix in the

¹³ <https://colab.research.google.com>.

Table 1 Iteration-per epoch training loss and accuracy

Epoch no.	Iterations	Time taken	Loss	Accuracy
1/5	12,000	76 min 9 s	0.5120	0.8916
2/5	12,000	78 min 5 s	0.4819	0.9047
3/5	12,000	86 min 2 s	0.4266	0.9186
4/5	12,000	75 min 8 s	0.3872	0.8961
5/5	12,000	80 min 3 s	0.3586	0.9225
Average accuracy				90.67%

x-axis and y-axis consecutively. The verbose information is kept as 1 for word (vector) to training logs. Finally, we print the collective outcome as the `model.summary()`.

We start by defining the parameters for the channels for our convolutional layers. We define filter size as 128, kernel size as 8, and ReLU activation. Then we merge the layers using concatenation. We also isolate the dense layer and the output layer at this stage, calling their respective activation functions. Then we load the global vector corpus in the feature mapping layer and input our training data. We also define the `document_length` as `max_length` and set the vocabulary size by `vocab_size` by incrementing the word tokenizer index. With that, we define the `embedding_vector_length`. Further, we compile our model using `binary_crossentropy` loss, adam optimizer, and accuracy metrics. We also allow to flatten the model and represent the shapes of the model. Now at this point, the model is ready to enter the training phase. Then the max-pooling layers are added for further refinement of the distribution of the feature set. These are two connected layers with dropouts of filter sizes 16 each and kernel sizes 48×48 and 32×32 . At this phase, we are using the `model.dropout` before the fully connected output layers is added. Now we define the training and testing epoch with activation function for output layer and `batch_sizes` respectively as 20 and 15. We use the Softmax activation here. Furthermore, we fit the training data into our model using the `split()` function. Next, we load the test set for evaluation, and we print the accuracy per epoch completion, with `model.evaluation()`. For our experiment, we iterated our CNN model for sentiment classification accuracy 500 times. We divided the iterations as distinct 5-set epochs for observing the accuracy and time taken for

training in each epoch set. In our distribution, each epoch consisted of 100 iterations. We present the training and testing time of each epoch set along with the accuracy, and also the total average accuracy in Table 1.

8 Results and analysis

8.1 Performance evaluation

We use two outline classifiers, Softmax [57, 58] and Rectified Linear activation [59, 60] with our convolutional networks to compare the performance growth. These two activations functions are widely used in wide experimental variations of CNN. We observe that our baseline convolutional network with Softmax outline achieves a high-test accuracy even before the first 100 epochs. The accuracy also peaks above 80% during this phase. Hence an overlap takes place between 0 and 100 epochs, where the error rate comes down while the accuracy emerges. Meanwhile, the accuracy simultaneously goes up to almost touching the peak margin of 85%, and it mostly settles within the high accuracy range of 85–90% for the rest of the training phase. Similarly, the convolutional network coupled with sigmoid performs, but the closer analytical observation reveals it tops an accuracy of 86.95%, falling short of the first module. The pre-trained vectorization inputs fed to each respective module are the same. Hence it is justified that the softmax function attached as the output classifier with a CNN performs better than the aspect-based classification tasks [61]. This pattern stands contrary to the previous observation of tanh providing better training performance than sigmoid or softmax [62], at least for the sentiment classification. Table 2 represents the training analysis concerning epochs. The performance visibility helps to narrow down the F-measure analysis further.

8.2 Error analysis

8.2.1 Quantitative analysis

We further compare the first two classifier combined modules in Table 3. Here the performance is representing the classification report generated from the training phase. The accuracy parameters from the generated report

Table 2 Summary information for training sets, where Set is the epoch count per 100 cycles, and Acc. indicates successful classification of sentiments from the data

Classifiers	Set	Acc.	Set	Acc.	Set	Acc.	Set	Acc.	Set	Acc.
CNN w. softmax	100	83.07	200	82.15	300	85.33	400	87.79	500	90.67
CNN w. ReLU	100	81.12	200	81.03	300	82.50	400	84.52	500	86.95

Bold value have been used to mention and display the better obtained results

Table 3 Comparative evaluation of the CNN outer classifiers w.r.t classification report

Module	Classification report			
	precision	Recall	F1-score	Accuracy
CNN w. softmax	0.9030	0.9016	0.9064	0.9067
CNN w. ReLU	0.8605	0.8590	0.8683	0.8695

Table 4 Dataset details for train and test partition

Label	Train set	Test set
Total	4,80,000	1,20,000
Polarity rated	4,80,000	1,19,764
Positive	89,360	12,700
Negative	2,59,000	75,051
Neutral	1,31,640	32,249

Table 5 Tweet insights representing tripolar sentiment category

	Tweets	Rating	Sentiment
1	@ <i>stpiindia</i> The govt is taking all necessary step to ensure successful national lockdown by warranting essential services to citizens	1	Positive
2	@ <i>nicd_sa</i> Who is eligible for #COVID19 testing? The guideline definition for patients under investigation has been updated	0	Neutral
3	@ <i>JoeBiden</i> Donald Trump is not responsible for the coronavirus, but he is responsible for failing to prepare our nation to respond to it	-1	Negative
4	@ <i>BarackObama</i> Due to the Corona virus it is difficult to get the money together	-1	Negative
5	@ <i>benhabib6</i> EU failures highly exposed as coronavirus forces it to abandon very core of project: state aid laws	-1	Negative
6	@ <i>DianaLogue2</i> Fox news sued for 'willfully and maliciously deceiving public' about the dangers of the coronavirus	-1	Negative

suggest that CNN with softmax classifier scores stable and better outputs for all the metrics provided.

8.2.2 Qualitative analysis

For qualitative analysis, we showcase the insight of our train and test partition in Table 4. For a comparative reference, here we mention the total size of both the cases. For training and set iterations, we can observe that, while the test is completed on fully polarity rated data, during the testing phase the network was unable to identify 236 tweets.

Post-training the network, we opt for the accuracy evaluation with the test set partition of data. For the given test set tweets, our network successfully evaluates the tweets with corresponding sentiments. However, it also misses out on a few occasions. We present the tweets rated on par sentiments in Table 5. As expected, most of the tweets reflect mass pessimism and negativity about the pandemic and the virus itself.

9 Comparative evaluation

We start our experimental analysis with the analytical details of our model and proceed for equivalent comparisons with a handpicked number of previous state-of-art systems. For the comprehensive approach for an overall comparison, we are not comparing fine-grained performances along with phase-by-phase accuracy. Rather we scrutinize the overall accuracy achieved by each respective model. For the equipollent comparison, we have divided the test case comparisons into two subsequent parts, followingly:

- Direct accuracy comparisons on two same corpora consisting of users' rating and reviews for movies. This comparative evaluation comprises the previous experimentations on similar deep neural network models with any distinct baseline model(s) used.
- A comparison among the tasks on similar real event data scraping and processing for sentiment analysis.

At first, we intend to scale comparative performance measurements by neural network models in a similar large public dataset. Table 6 shows various models in comparison based on the Stanford Sentiment Treebank (SST) [63] in terms of accuracy. This corpus is built from movie reviews and contains over 10,000 review snippets accumulated from the Rotten Tomatoes.¹⁴ Stanford's dataset has been designed to train a model to identify sentiment in longer phrases. This is a good roundup validation of how our model can detect the sentiments, especially when the stream of input vectors comes in longer channels, i.e., from the multi-sentence tweets. We test our proposed model with the aforesaid dataset itself for an analogous comparison with the discussed previous works. Ten models are evaluated here (including ours) to compare the benchmark scale with identical data. We report the accuracy only on the phrase level sentiment identification. When compared with the previous variations of deep neural models [64–72], our model performs significantly better, just falling short of the state-of-the-art result achieved by Li et al. [64]. In this work, the authors have

¹⁴ <https://www.rottentomatoes.com/>.

Table 6 Performance comparison with various neural networks for textual review evaluation

Dedicated test results on SST	Overall accuracy (%)
CNN-LSTM (Li et al.) [64]	95.01
Ours	93.81
RTC + attention (Zhang and Zhang) [65]	93.40
BiConTree (Teng and Zhang) [66]	92.80
Two-channel CNN-Tree-LSTM (Van et al.) [67]	89.70
TreeGRU with attention (Kokkinos and Potamianos) [68]	89.50
BiLSTM+NBSVM (Tien and Le) [69]	88.58
AdaMC-RNTN (Dong et al.) [70]	88.50
SAAN (Lei et al.) [71]	49.70
SC-EAV (Shin et al.) [72]	48.80

merged two channels of BiLSTM with CNN, while taking the sentiment lexicons as inputs. In contrast, as we have already performed the sentiment analysis of our training data using Logistic Regression, we test our single but deep CNN model with the SST phrase data. Also, we manage to perform just slightly better than that of [65], where the researchers have considered the input word vectors to the LSTM as the tree node traversal from left to right. Also, while testing the proposed model with SST, they exploited the attention mechanism for successfully analyzing the contextual sentiments. However, on the contrary, among the top-of-the-line approaches here we use deep CNN without any alteration, but only with the outline classifier tuning. Hence, the resulting outcome from our model stands as the state-of-the-art for a vanilla deep network.

Another popular public dataset is the Stanford large movie review dataset, or popularly known as the IMDB movie review dataset [73]. A significant number of peak researches for feedback style sentiment analysis, word vector generation, and polarity dentitional works have been carried

out on the same. Here are proposed approach manages to outperform several recent works. Our deep CNN with sentiment analyzed word vectors perform slightly better than that of a contextual bi-directional long-short network [74]. While the ensemble approach of statistical kernels for movie review analysis [76] scores a dip lower than the deep neural counterparts. Also, a similar approach with sentiment lexicons [20] also performs well in parity with the other mentioned works here. The latter mentioned works here signifies the dominance of sentence-based sentiment analysis with the primary RNN or CNN approaches, from where the recent works have mostly evolved. In the comparative test case scenario with the identical corpus, our work has scored a better overall accuracy, because of two key points; considering the gradient parameters for the training data to meticulously identify the region-based impacts for successful predictive analysis, and the weightage sentiment vector assignments from our native test data using the GloVe vectorization (Table 7).

Followingly, we put our experimentation performance with the likings of similar research works of the same fundamentals; tracking down a live event as it unfolds. It is rather more complex since the data gathered is novel, and still in a raw unstructured state, without any prior insights and practical decision-making results. In the past, several works of such nature have been carried out, for instance, the 2012 U.S. presidential election [81], identifying the sentiment analysis trend for the 2014 Indian general election [82], political discourse analysis from Twitter for the 2016 US presidential elections [83], and analyzing the Twitter sentiments on the introduction of GST in India [84]. While the first three experimentations did not proceed for a sentiment classification or predictive accuracy approach, rather the researchers chose to represent an application-based approach for Twitter sentiment visualization on the discussed topics. Simultaneously, we surely outperform our previous work on

Table 7 Performance comparison with similar neural networks for large movie review set

Dedicated test results on IMDB	Overall accuracy (%)
Ours	93.27
Context. Bi LSTM-RNN (Schuller and Mousa) [74]	91.55
Dilated CNN with CRF (Strubell et al.) [75]	90.54
Ensemble NB + SVM + RF + LR (Ghosh and Sanyal) [76]	90.18
Deep CNN-LSTM (Yenter and Verma) [77]	89.22
Sentiment lexicons (Araque et al.) [20]	89.01
Multi-branch CNN (Vielma et al.) [78]	88.90
Recursive RNN on IMDB reviews (Timmaraju and Khanna) [79]	83.88
CNN-multichannel on open movie reviews (Kim) [80]	81.50
Non-static CNN on open movie reviews (Kim) [80]	81.50

Table 8 Performance comparison with state-of-art experiments on public benchmark datasets

Popular open-source datasets	Previous state-of-art results (%)	Our results (%)
Sanders' twitter analysis dataset	79.50 [85]	84.72
Amazon customer review dataset	90.00 [86]	90.61

Bold values have been used to mention and display the better obtained results

GST (83.85% prediction-based accuracy) by a significant margin (90.67% for the present work).

We also represent the performance of our convolutional network for three of the most widely accepted public text corpora. The Stanford large movie review dataset (popularly known as IMDB movie review dataset) corpus is available on the Keras library module to directly download and load into the network. While we manually read the remaining two corpora using the pandas data frame. We divide three of the corpora in the same split validation of 70:30 amount for training and validation, which is the same ratio as that of our original experiment. We maintain a side-by-side comparison with one of the peak results achieved for each discussed dataset, if not the best. This comparison helps us to scale the scope of our model. In any of the test runs, the predictive performance for accuracy did not drop below 80%. Maintaining good overall performance linearity for these datasets surely ensures the capability of our model. For this test-based comparison, we show our conducting results in Table 8.

10 Predictive analysis

After the evaluation, we have two distinct sets of labeled tweet sets, one train set, and one refined test set. These corpora can now be merged into one. Now, this data is a valuable data insight into our predictive analysis.

For a timeline-based prediction of the number of case reports and how the volume of the Coronavirus research data will rise, we are applying the concept of Bayesian regression here. Bayes regression is a special categorization of linear regression, in which the prior predictions take place based on the parameters, and even the posterior distribution coverage determined from the parameters is also considered when making predictions. Hence, we compute a mean be subject to the report of cases, and the growth of data w.r.t the timeline from March 15th, 2020.

In Bayesian linear regression, we consider:

$$\text{Prior condition : } p(\varphi) = M(S, D, C) \in T_n \quad (15)$$

$$\text{Likelihood layout : } p(x|y, \varphi) = M((x|y)\varphi_n^T) \quad (16)$$

Here we predict the probability p with condition φ , which combines the mapping M of sentiment, the date that belongs to the complete entry of the tweets. Furthermore, now we propose the likelihood as p for scaling the previously mentioned parameters in x and y coordinates, with the generated mapping values of the number of tweets. N which turns the parameter vector into a random variable. We explicitly place the φ or the probabilistic factor in the likely scenario to connect it to the tweets, since we are dealing with a large number of tweets to generate insight from.

Combining Eqs. (15) and (16), the final prior probabilistic model generates Eq. (17), which depicts the combined distribution of observed and unobserved variables in the x and y scale.

$$p(x|y, \varphi) = p(x|y), p(\varphi) \quad (17)$$

Once the prior conditions are organized and ready to be mapped in a coordinate plane the immediate inevitable question arises, how can we shape and predict the Coronavirus cases for the near future? One thing to remember is that we cannot predict a long-term scenario just as of now; maybe that is best kept for a further extension of this work. Also, we are no health experts, we are trying to fit a large data into a model for predictive analysis. Now, from the parameter inputs plotted in x and y plane, both a and b depends on the same set of parameters vs. tweets to plot the data growth. Which is $x, y \in P(\varphi)$.

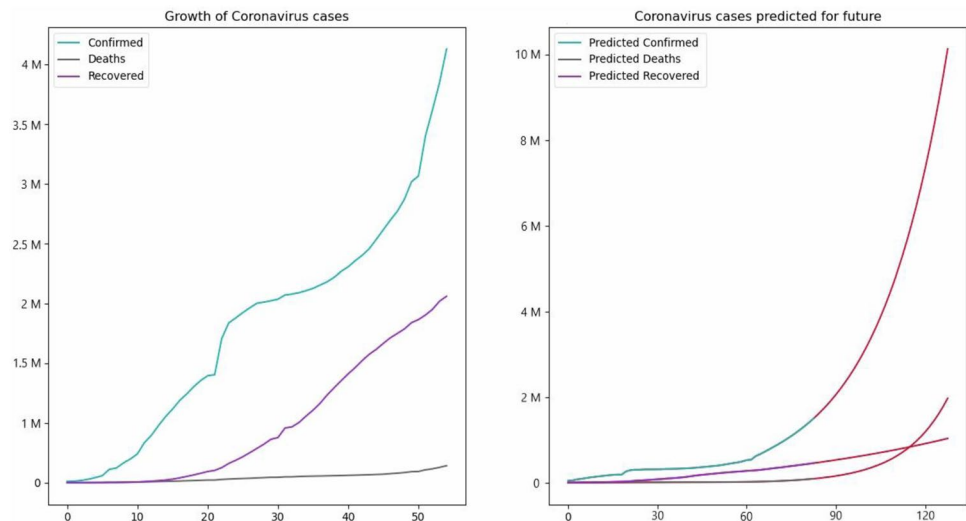
Hence, the posterior distribution using Bayes regression as:

$$p(\varphi) = \frac{P(x|y)p(\varphi \in M)}{P(y|x)} \quad (18)$$

where x holds the set of parameters plotting and y observes the ascent of the case predictions w.r.t to time. The collection of the related target parameters is already achieved from the previous sections. Consequently, $p(x, y)$ is the conditional distribution plane and $p(\varphi)$ is the grouped combinations of plotting parameters. We represent the plotted graph in Fig. 7.

From the figure, we can visualize the comparison in contrast between the present scenario of the first 50 days from March 2020, and how the Coronavirus can grow even larger in the coming days. it is also evident from Fig. 7 that within 120 days' time frame from the time of this work, the number of predicted confirmed cases can grow past 10 million people, while the predicted number of deaths can rise to 2 million. Also, the recovery is predicted to reach an exceeding point of 2 million people. The blue, purple, and grey lines on the left sub-graph define the present trend of confirmed, recovered, and death cases. In analogy, the red lines in the right side predicted sub-graph depicts the upward rising

Fig. 7 Present vs. future
Coronavirus metrics growth
prediction



trend of the same parameters, as well as indicate the growth prediction here. Hence, these lines are the continuation of the lines from the left sub-graph. There are a lot of external and probable parameters like the regional weather, progress in vaccine research, natural immunity build-up, that can curb down this growth and help us to get control over the spread of the virus in the near future. Since the Bayesian Regression is derived to predict continuous values in a linear scale, it can be determined that from the current context US and India may likely lead the Covid affected curve in the future. Along with the other European countries, highly populated and dense south Asia Pacific countries can also be heavily affected on the same. The BR-based prediction also depends on the progressive validation, i.e., a prediction is possible only when an observation scenario appears. It can be stated as analyzing the trend from the current data. Therefore, the probability estimation for our predictive analysis relies on the statistical probabation of where the numbers can peak from the present status.

11 Conclusion and future work

In this work, we proposed a deep CNN for sentiment accuracy evaluation, followed by a predictive analysis of the tweets focused on Coronavirus. We collected live-streamed tweets from Twitter as a phase of this large-scale global pandemic. Then we build our data model using logistic regression for insight and visualization, which is followed by the current trend analysis of the reported cases with a gradient number of parameters. Then we train our convolutional network with the train data and input word vectorization explicitly. Thereafter we perform the accuracy prediction for the polarity of the test data. Our study showcased that we were able to find accuracy with 90.67%, even outperforming a handpicked number of previous state-of-art works.

Followingly, we perform Bayesian regression on the evaluated corpus for constant point to point predictability based on the prior and posterior conditional distribution on a common plane. Since no research is perfect, we would like to address that we have not considered multilingual tweets and clinical data from the respective governments from the affected countries. As the situation is in the present state to date, the medical data is ought to evolve much more in the coming months and years. That is exactly where the contribution of this work is founded. This work can further be extended for a multimodal prediction model for any infectious disease, gathering data from audio, video, and obviously from text sources. This work can lead the way for data scientists and frontend engineers to develop up-to-date data modeling and prediction software using Python web frameworks such as Django,¹⁵ Flask,¹⁶ and GUI library such as Tkinter.¹⁷ Also, before our work, one of the largest textual data on Covid-19 was developed by both Google¹⁸ and Johns Hopkins University Centre for Systems Science and Engineering¹⁹ individually. Currently, we can also contribute our sophisticated Twitter data consisting of 600 k tweets with indexed parameters. We aim to make our data openly available on Github²⁰ for any further extension of this research work. Furthermore, we are keen to exploit this work with an extension towards lexical level disambiguation of the similarly used words in different contexts for Coronavirus, and a hybrid neural network embedded with a rule-based core kernel as the classifier for even enhanced analysis.

¹⁵ <https://docs.djangoproject.com/en/3.0/>.

¹⁶ <https://flask.palletsprojects.com/en/1.1.x/>.

¹⁷ <https://docs.python.org/3/library/tkinter.html>.

¹⁸ <https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset>.

¹⁹ <https://systems.jhu.edu/research/public-health/ncov/>.

²⁰ <https://github.com/SouravD-Me>.

Acknowledgments We dedicate our work to the countless people who lost their lives or closed ones in this catastrophe. Also, to the frontline warriors who are constantly risking their lives to restrain the catastrophe.

References

- Corvey WJ, Vieweg S, Rood T, Palmer M (2010) Twitter in mass emergency: what NLP can contribute. In: Proceedings of the NAACL HLT 2010 workshop on computational linguistics in a world of social media, pp 23–24
- Sriram B, Fuhry D, Demir E, Ferhatosmanoglu H, Demirbas M (2010) Short text classification in twitter to improve information filtering. In: Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval, pp 841–842
- Ekman P (1999) Basic emotions. *Handb Cognit Cmot* 98(45–60):16
- Pak A, Paroubek P (2010) Twitter as a corpus for sentiment analysis and opinion mining. In: *LREc* vol. 10, no. 2010, pp 1320–1326
- Pandarachalil R, Sendhilkumar S, Mahalakshmi GS (2015) Twitter sentiment analysis for large-scale data: an unsupervised approach. *Cognit Comput* 7(2):254–262
- Das S, Das D, Kolya AK (2020) Sentiment classification with GST tweet data on LSTM based on polarity–popularity model. *Sadhana*. <https://doi.org/10.1007/s12046-020-01372-8>
- Turney P (2002) Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th annual meeting of the association for computational linguistics, pp 417–424
- Wiebe J, Bruce R, O’Hara TP (1999) Development and use of a gold-standard data set for subjectivity classifications. In: Proceedings of the 37th annual meeting of the association for computational linguistics, pp 246–253
- Hu M, Liu B (2004) Mining and summarizing customer reviews. In: Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining, pp 168–177
- Liu B (2012) Sentiment analysis and opinion mining. *Synth Lect Hum Lang Technol* 5(1):1–167
- Montoyo A, MartiNez-Barco P, Balahur A (2012) Subjectivity and sentiment analysis: an overview of the current state of the area and envisaged developments. *Decis Support Syst* 53(4):675–679
- Jiang L, Yu M, Zhou, M, Liu X, Zhao T (2011) Target-dependent twitter sentiment classification. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, pp 151–160
- Chih-Cheng L, Shih T-P, Ko W-C, Tang H-J, Hsueh P-R (2020) Severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) and corona virus disease-2019 (COVID-19): the epidemic and the challenges. *Int J Antimicrob Agents* 55(3):105924
- Kabir MY, Madria S (2020) CoronaVis: a real-time COVID-19 tweets data analyzer and data repository. *arXiv preprint arXiv:2004.13932*
- Yang Q, Alamro H, Albaradei S, Salhi A, Lv X, Ma C, Zhang X (2020) Senwave: monitoring the global sentiments under the Covid-19 pandemic. *arXiv preprint arXiv:2006.10842*
- Samuel J, Ali GG, Rahman M, Esawi E, Samuel Y (2020) Covid-19 public sentiment insights and machine learning for tweets classification. *Information* 11(6):314
- Lamsal R (2020) Design and analysis of a large-scale COVID-19 tweets dataset. *Appl Intell*. <https://doi.org/10.1007/s10489-020-02029-z>
- Chakraborty K, Bhatia S, Bhattacharyya S, Platos J, Bag R, Hassanien AE (2020) Sentiment analysis of COVID-19 tweets by deep learning classifiers—a study to show how popularity is affecting accuracy in social media. *Appl Soft Comput* 97:106754
- Vielma C, Verma A, Bein D (2020) Single and multibranch CNN-bidirectional LSTM for IMDb sentiment analysis. In: 17th international conference on information technology—new generations (ITNG 2020). Springer, Cham, pp 401–406
- Araque O, Zhu G, Iglesias CA (2019) A semantic similarity-based perspective of affect lexicons for sentiment analysis. *Knowl-Based Syst* 165:346–359
- Conneau, A, Schwenk, H, Barrault, L, Lecun, Y (2016) Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*
- Wehrmann J, Becker W, Cagnini HE, Barros RC (2017) A character-based convolutional neural network for language-agnostic twitter sentiment analysis. In: 2017 international joint conference on neural networks (IJCNN). IEEE, pp 2384–2391
- Pennington J, Socher R, Manning, CD (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
- Kouloumpis E, Wilson T, Moore J (2011) Twitter sentiment analysis: the good the bad and the omg! In: 5th international AAAI conference on weblogs and social media
- Severyn A, Moschitti A (2015) Twitter sentiment analysis with deep convolutional neural networks. In: Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, pp 959–962
- Das S, Kolya AK (2017) Sense GST: text mining & sentiment analysis of GST tweets by naive bayes algorithm. In: 2017 3rd international conference on research in computational intelligence and communication networks (ICRCICN). IEEE, pp 239–244
- Wang Y, Zhang J (2017) Keyword extraction from online product reviews based on bi-directional LSTM recurrent neural network. In: 2017 IEEE international conference on industrial engineering and engineering management (IEEM). IEEE, pp 2241–2245
- Basaldella M, Antolli E, Serra G, Tasso C (2018) Bidirectional LSTM recurrent neural network for keyphrase extraction. In: Italian research conference on digital libraries. Springer, Cham, pp 180–187
- Moriya S, Shibata C (2018) Transfer learning method for very deep CNN for text classification and methods for its evaluation. In: 2018 IEEE 42nd annual computer software and applications conference (COMPSAC), vol. 2. IEEE, pp 153–158
- Catanzaro B, Sundaram N, Keutzer K (2008) Fast support vector machine training and classification on graphics processors. In: Proceedings of the 25th international conference on machine learning, pp 104–111
- Zhang H, Zheng Z, Xu S, Dai W, Ho Q, Liang X, Xing EP (2017) Poseidon: an efficient communication architecture for distributed deep learning on GPU clusters. In: Proceedings of the 2017 USENIX conference on usenix annual technical conference, pp 181–193
- Ciresan DC, Meier U, Masci J, Gambardella LM, Schmidhuber J (2011) Flexible, high performance convolutional neural networks for image classification. In: 22nd international joint conference on artificial intelligence
- Cong J, Xiao B (2014) Minimizing computation in convolutional neural networks. In: International conference on artificial neural networks. Springer, Cham, pp 281–290
- Xiao T, Xu Y, Yang K, Zhang J, Peng Y, Zhang Z (2015) The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 842–850
- Sun M, Song Z, Jiang X, Pan J, Pang Y (2017) Learning pooling for convolutional neural network. *Neurocomputing* 224:96–104

36. Wang Y, Huang M, Zhu X, Zhao L (2016) Attention-based LSTM for aspect-level sentiment classification. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp 606–615
37. Sherstinsky A (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D Nonlinear Phenom* 404:132306
38. Neil D, Pfeiffer M, Liu SC (2016) Phased LSTM: accelerating recurrent network training for long or event-based sequences. In: *Advances in neural information processing systems*, pp 3882–3890
39. Lavin A, Gray S (2016) Fast algorithms for convolutional neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 4013–4021
40. Albeahdili HM, Han T, Islam NE (2015) Hybrid algorithm for the optimization of training convolutional neural network. *Int J Adv Comput Sci Appl* 1(6):79–85
41. Güera D, Delp EJ (2018) Deepfake video detection using recurrent neural networks. In: 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS) IEEE, pp 1–6
42. Collobert R, Weston J (2008) A unified architecture for natural language processing: deep neural networks with multitask learning. In: *Proceedings of the 25th international conference on machine learning*, pp 160–167
43. McCann B, Bradbury J, Xiong C, Socher R (2017) Learned in translation: contextualized word vectors. In: *Proceedings of the 31st international conference on neural information processing systems*, pp 6297–6308
44. Er MJ, Zhang Y, Wang N, Pratama M (2016) Attention pooling-based convolutional neural network for sentence modelling. *Inf Sci* 373:388–403
45. Alamoodi AH, Zaidan BB, Zaidan AA, Albahri OS, Mohammed KI, Malik RQ, Almahdi EM, Chyad MA, Tareq Z, Albahri AS, Hameed H, Alaa M (2020) Sentiment analysis and its applications in fighting COVID-19 and infectious diseases: a systematic review. *Expert Syst Appl* 167:114155. <https://doi.org/10.1016/j.eswa.2020.114155>
46. Rebiai Z, Andersen S, Debrenne A, Lafargue V (2019) SCIA at SemEval-2019 task 3: sentiment analysis in textual conversations using deep learning. In: *Proceedings of the 13th international workshop on semantic evaluation*, pp 297–301
47. Ma Y, Peng H, Cambria E (2018) Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM. In: *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1
48. Majumder N, Hazarika D, Gelbukh A, Cambria E, Poria S (2018) Multimodal sentiment analysis using hierarchical fusion with context modeling. *Knowl-Based Syst* 161:124–133
49. Yadav A, Vishwakarma DK (2020) Sentiment analysis using deep learning architectures: a review. *Artif Intell Rev* 53(6):4335–4385
50. Howard J, Ruder S (2018) Universal language model fine-tuning for text classification. In: *Proceedings of the 56th annual meeting of the association for computational linguistics*, vol 1: Long Papers, pp 328–339
51. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781)
52. Wehrmann J, Mattjie A, Barros RC (2018) Order embeddings and character-level convolutions for multimodal alignment. *Pattern Recognit Lett* 102:15–22
53. Akhtar MS, Kumar A, Ekbal A, Bhattacharyya P (2016) A hybrid deep learning architecture for sentiment analysis. In: *Proceedings of COLING 2016, the 26th international conference on computational linguistics: technical papers*, pp 482–493
54. Mitchell TJ, Beauchamp JJ (1988) Bayesian variable selection in linear regression. *J Am Stat Assoc* 83(404):1023–1032
55. Snyman JA (2005) A gradient-only line search method for the conjugate gradient method applied to constrained optimization problems with severe noise in the objective function. *Int J Numer Methods Eng* 62(1):72–82
56. Asadi B, Jiang H (2020) On approximation capabilities of ReLU activation and softmax output layer in neural networks. arXiv preprint [arXiv:2002.04060](https://arxiv.org/abs/2002.04060).
57. Zhang Y, Zhang Z, Miao D, Wang J (2019) Three-way enhanced convolutional neural networks for sentence-level sentiment classification. *Inf Sci* 477:55–64
58. Attia M, Samih Y, Elkahky A, Kallmeyer L (2018) Multilingual multi-class sentiment classification using convolutional neural networks. In: *Proceedings of the 11th international conference on language resources and evaluation (LREC 2018)*
59. Qiu S, Xu X, Cai B (2018) FReLU: flexible rectified linear units for improving convolutional neural networks. In: 2018 24th international conference on pattern recognition (ICPR). IEEE, pp 1223–1228
60. Qiumei Z, Dan T, Fenghua W (2019) Improved convolutional neural network based on fast exponentially linear unit activation function. *IEEE Access* 7:151359–151367
61. Tay Y, Tuan LA, Hui SC (2018) Learning to attend via word-aspect associative fusion for aspect-based sentiment analysis. In: *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1
62. Nwankpa C, Ijomah W, Gachagan A, Marshall S (2018) Activation functions: comparison of trends in practice and research for deep learning. arXiv preprint [arXiv:1811.03378](https://arxiv.org/abs/1811.03378).
63. Socher R, Perelygin A, Wu J, Chuang J, Manning CD, Ng AY, Potts C (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp 1631–1642
64. Li X, Wu X (2015) Long short-term memory based convolutional recurrent neural networks for large vocabulary speech recognition. In: 16th annual conference of the international speech communication association
65. Zhang Y, Zhang Y (2019) Tree communication models for sentiment analysis. In: *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp 3518–3527
66. Teng Z, Zhang Y (2017) Head-lexicalized bidirectional tree LSTMs. *Trans Assoc Comput Linguist* 5:163–177
67. Van VD, Thai T, Nghiem MQ (2017) Combining convolution and recursive neural networks for sentiment analysis. In: *Proceedings of the 8th international symposium on information and communication technology*, pp 151–158
68. Kokkinos F, Potamianos A (2017) Structural attention neural networks for improved sentiment analysis. In: *Proceedings of the 15th conference of the European chapter of the association for computational linguistics*, vol 2: Short Papers, pp 586–591
69. Nguyen HT, Le Nguyen M (2017) An ensemble method with sentiment features and clustering support. In: *Proceedings of the 8th international joint conference on natural language processing*, vol 1: Long Papers, pp 644–653
70. Dong L, Wei F, Xu K, Liu S, Zhou M (2016) Adaptive multi-compositionality for recursive neural network models. *IEEE/ACM Trans Audio Speech Lang Process*. <https://doi.org/10.1109/TASLP.2015.2509257>
71. Lei Z, Yang Y, Yang M (2018) SAAN: a sentiment-aware attention network for sentiment analysis. In: *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp 1197–1200
72. Shin B, Lee T, Choi JD (2017) Lexicon integrated CNN models with attention for sentiment analysis. In: *Proceedings of the 8th*

- workshop on computational approaches to subjectivity, sentiment and social media analysis, pp 149–158
73. Maas A, Daly RE, Pham PT, Huang D, Ng AY, Potts C (2011) Learning word vectors for sentiment analysis. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, pp 142–150
 74. Mousa A, Schuller B (2017) Contextual bidirectional long short-term memory recurrent neural network language models: a generative approach to sentiment analysis. In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics, vol 1: Long papers, pp 1023–1032
 75. Strubell E, Verga P, Belanger D, McCallum A (2017) Fast and accurate entity recognition with iterated dilated convolutions. In: Proceedings of the 2017 conference on empirical methods in natural language processing, pp 2670–2680
 76. Ghosh M, Sanyal G (2018) An ensemble approach to stabilize the features for multi-domain sentiment analysis using supervised machine learning. *J Big Data* 5(1):44
 77. Yenter A, Verma A (2017) Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis. In: 2017 IEEE 8th annual ubiquitous computing, electronics and mobile communication conference (UEMCON), IEEE, pp 540–546
 78. Vielma C, Verma A, Bein D (2020) Single and multibranch CNN-bidirectional LSTM for IMDb sentiment analysis. In: 17th international conference on information technology—new generations (ITNG 2020). Springer, Cham, pp 401–406
 79. Timmaraju A, Khanna V (2015) Sentiment analysis on movie reviews using recursive and recurrent neural network architectures, pp 1–5
 80. Kim Y (2014) Convolutional neural networks for sentence classification. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1746–1751
 81. Wang H, Can D, Kazemzadeh A, Bar F, Narayanan S (2012) A system for real-time twitter sentiment analysis of 2012 us presidential election cycle. In: Proceedings of the ACL 2012 system demonstrations. Association for computational linguistics, pp 115–120
 82. Almatrafi O, Parack S, Chavan B (2015) Application of location-based sentiment analysis using twitter for identifying trends towards indian general elections 2014. In: Proceedings of the 9th international conference on ubiquitous information management and communication, pp 1–5
 83. Yaqub U, Chun SA, Atluri V, Vaidya J (2017) Analysis of political discourse on twitter in the context of the 2016 US presidential elections. *Gov Inf Q* 34(4):613–626
 84. Das S, Das D, Kolya AK (2020) An approach for sentiment analysis of GST tweets using words popularity versus polarity generation. In: Computational intelligence in pattern recognition. Springer, Singapore, pp 69–80
 85. Kavak H, Vernon-Bido D, Padilla JJ (2018) Fine-scale prediction of people's home location using social media footprints. In: Thomson R, Dancy C, Hyder A, Bisgin H (eds) International conference on social computing, behavioral-cultural modeling and prediction and behavior representation in modeling and simulation. Springer, Cham, pp 183–189
 86. Güner L, Coyne E, Smit J (2019) Sentiment analysis for amazon.com reviews

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.