A Self-Reconfigurable System for Mobile Health Text Misinformation Detection

Sanjaikanth E Vadakkethil Somanathan Pillai

School of Electrical Engineering and Computer Science

University of North Dakota

Grand Forks, ND 58202-7165, USA

sanjaikanth@gmail.com

Wen-Chen Hu

School of Electrical Engineering and Computer Science

University of North Dakota

Grand Forks, ND 58202-7165, USA

wenchen@cs.und.edu

*Abstract*—Misinformation is always a serious problem for the general public, especially during pandemic. People constantly receive text messages of related coronavirus news and its cures from their smartphones, which have become a major device for communication in these days. These health text messages help people update their coronavirus knowledge continuously and better manage their health, but some of the messages may mislead people and may even cause a fatal result. This research tries to identify mobile health text misinformation by proposing a self-reconfigurable system, which includes the functions of preprocessing (involving lexical analysis, stopword removal, and stemming), a dataflow graph based on TensorFlow, and a reconfiguring method using a database for self-improvement. Preliminary experiment results show the proposed method significantly improves the accuracy of the mobile health text misinformation detection without self-reconfiguration. However, the results also show the accuracy is not consistent. More refinements and testing need to be done before the method could be put into use.

# Introduction

The World War II was a tragedy in the human history. Nevertheless, another tragedy is on its way to pass the WWII. By the end of 2021, the COVID-19 casualties were about to pass the World War II casualties, estimated at 50-56 million. These high casualties put everyone on the alert. People try to find any information that helps them fight the virus. Much information they receive is from their smartphones without doubt in these days because smartphones have become an indispensable device for everyone. One popular function used by smartphone users is sending and receiving text messages. Instead of reading newspapers or watching TV news, many of the mobile users especially younger generations receive their daily news or information via text messages. Other than useful and unbiased information, many of the messages are filtered or may even be distorted. During the pandemic, the problem has become even more serious because the health text misinformation not only gives wrong information, but may also cause fatal results such as advocacy of the ineffectiveness of vaccine. This research tries to mitigate the problem by identifying mobile health text misinformation, so the mobile users can use the findings to better judge the messages they receive and take actions accordingly.

This research proposes a self-reconfigurable system to identify mobile health text misinformation, which is briefly described as follows. This is a supervised learning system, so before the system is put into use, it needs training by using a set of known text messages. The initial parameters of the system are set by heuristics because the keywords of text messages are unknown in advance and have to be speculated. Afterwards the system starts its testing phase by receiving text messages, each will go through a series of steps: preprocessing (including lexical analysis, stopword removal, and stemming), indexing and storage, and testing (classification) by using a dataflow graph. Instead of applying the results immediately, the first round of testing is used to reconfigure the system in order to generate better results. It is because the initial configuration is usually not desirable because system parameters are unknown in the very beginning. After the first round of testing, better parameters could be found from the test results. The complete steps are repeated with better parameter and more accurate identifications are expected. Preliminary experiment results show the overall accuracy of the proposed method meets our expectation, but the accuracy is not consistent. An explanation for this may be because the messages do not provide much information. Small deviation may cause a great impact on the results. Further refinements are needed before it is put into actual work.

The rest of this paper is organized as follows. Section 2 shows the background information and related works on misinformation detection. The system structure and its components are given in Section 3. Section 4 proposes our major method, a self-reconfigurable dataflow graph, for detecting mobile health misinformation. The experiment results and evaluations are given in Section 5, followed by a conclusion and references.

# Background and Related Literature

This section gives the background information of this research and related research in case readers are interested in finding more relevant publications. Misinformation detection is critical and popular in these days because information could be created and sent by everyone, not just news agencies, and some may distribute misinformation unintentionally or on purpose. Many methods are used to detect all kinds of misinformation like politics, businesses, text messages, emails, or news. This research places the focus on mobile health text misinformation identification. If the results are favorable, the method may be extended to other kinds of information. Generic misinformation detection can be found from the articles [1], [2], [3], [4].

Misinformation is a difficult topic to tackle and a variety of misinformation exist. This research does not intend to solve all the misinformation problems at once. Instead, it focuses on two subjects: text messages and health-related misinformation, especially the COVID-19. Brennen, Simon, & Nielsen analyzed visual content in misinformation concerning COVID-19 [5]. It shows the value in both attending to visual content in misinformation and unnecessarity of a concern with only the representational aspects and functions of misinformation. Another study by Gupta, Gasparyan, Misra, Agarwal, Zimba, and Yessirkepov [6] shows while identifying social media as a potential source of misinformation on COVID-19, and a perceived high risk of plagiarism, more stringent peer review and skilled post-publication promotion are advisable. They recommend editors should play a more active role in streamlining publication and promotion of trustworthy information on COVID-19. Related articles can be found from the articles [7], [8], [9].

The media targeted by this research is text messages. Related research can be found from the article [7], whose task is defined as being able to classify a tweet as real or fake. The complexity of natural language constructs along with variegated languages makes this task very challenging. In this work, a deep learning model to learn semantic word embeddings is proposed to handle this complexity. The evaluations on the benchmark dataset (VMU 2015) show that deep learning methods are superior to traditional natural language processing algorithms. Ahmed, Ali, Hussain, Baseer, and Ahmed [10] analyze the performance of a fake news detection model based on neural networks using 3 feature extractors: TD-IDF vectorizer, Glove embeddings, and BERT embeddings. It was found that BERT embeddings for text transformation delivered the best performance. TD-IDF has been performed far better than Glove and competed the BERT as well at some stages. Other misinformation from social media can be found from the articles [11], [12], [13], [14], [15].

This research uses the artificial intelligence methods to identify misinformation. Kula, Choraś, Kozik, Ksieniewicz, and Woźniak [16] present an innovative solution for fake news detection that utilizes deep learning methods. Their experiments prove that the proposed approach is effective. Isha Priyavamtha, Vishnu Vardhan Reddy, Devisri, and Manek [17] propose a model to detect fake news that includes three main phases ‘preprocessing, feature extraction and classification’. Initially input is preprocessed to extract features using clustering algorithms. Subsequently, the model is developed to detect fake news. The Neural Networks and Linear Support Vector Clustering algorithms resulted in 99.90% and 97.5% accuracy respectively.

Our research is different from many others by focusing on mobile health text misinformation, especially mobile COVID-19 misinformation identification. Research of misinformation detection and management has been studied extensively. Interested readers can refer to the surveys from the articles [18], [19], [20], [21].

# the proposed system

This research is to identify mobile health text misinformation by using a self-reconfigurable system. This section gives an overview of the proposed system. Details of the reconfiguring method will be given in the next section.

*A. Five Classes of Mobile Text Message*

It is too simple to classify a mobile health text message as either true or fake because some other classes exist. This research categorizes a message into one of the following five classes:

* *True*, which is true information and is without a doubt. For example, it is true that a vaccine to prevent COVID-19 is available because COVID-19 vaccines have been authorized by the U.S. Food and Drug Administration (FDA) and vaccine programs have begun across the country.
* *Fake*, which could be either misinformation or disinformation. For example, it is an obviously fake news that the COVID-19 vaccines contain microchips for government tracking because the current technology has not been this advanced yet.
* *Misinformative*, which is false or out-of-context information that is intentionally or unintentionally presented as fact to deceive.
* *Disinformative*, which is a type of misinformation that is intentionally delivered the false or misleading information to deceive or mislead readers.
* *Neutral*, which cannot be decided by the proposed method.

The differences between misinformation and disinformation are not distinct. This research treats the former as a mistake. If the information is intentionally to deceive, it is classified as disinformation. Otherwise, it is misinformation. That is misinformation actively demonstrates the information that is communicated to mislead, whereas disinformation can be recognized as malicious tricks and computational publicity.

*B. System Structure of the Dataflow Graph*

Construction of the proposed system is rather complicated. To focus on the part of self-reconfiguration, Keras [23] and Tensorflow [24] are used to facilitate the system construction. Keras is a high-level open-source library for the neural network and TensorFlow is a software library for machine learning and artificial intelligence. A dataflow graph of TensorFlow consists of an operation and a tensor, where the operation is to find the outgoing tensor (class) of the ingoing tensor (text message). There are three phases before the system is ready for use in the final phase:

1. *Training phase*: Train the dataflow graph by using a set of known test messages.
2. *Initial testing phase*: Start testing the trained dataflow graph by using a set of unknown messages and receive the suggested results (classes).
3. *Self-reconfiguring phase*: The system will automatically reconfigure itself based on the previous phases and their messages and results.
4. *Final testing phase*: The system is ready for use. A mobile text message is submitted to the system, and one of the five class (true, fake, misinformation, disinformation, and neutral) will be recommended.

Fig. 1 shows the dataflow graph, which is with 10 input Boolean nodes and 5 output Boolean nodes. Each text message consists of at most 10 words (or phrases) for the input layer. The five output nodes of the output layer are the final classes: true, fake, misinformation, disinformation, and neutral. Only one of the five output nodes will be true for each message.

**Output Layer**

true

fake

word 10

**.**

**.**

**.**

**.**

**.**

**.**

**.**

**.**

**.**

neutral

disinformation

misinformation

**Input Layer**

**Hidden Layer**

word 1

word 2

Fig. 1. A sample dataflow graph from the Tensorflow.

*C. Training Phase*

Initially, we randomly pick related words or phrases for the nodes of the input layer. Of course, the chosen ones are not ideal because we have no idea what the ideal input nodes are. The self-reconfiguring phase will try to find better input nodes based on the input messages and their found results/classes. Fig. 2 shows the training phase of this system, which includes the components: lexical analysis, stopword removal, stemming, database, and a dataflow graph, and they will be described in this section.

Skeletal

messages

Raw

messages

Stemming

Lexical analysis

Stopword removal

(Trained) dataflow graph messages

Results (classes)

Database

Fig. 2. The training phase.

*D. Lexical Analysis*

Lexical analysis is the process of converting an input stream of characters into a stream of words or tokens, which are groups of characters with collective significance. It is the first stage of automatic indexing which is the process of algorithmically examining information items to generate lists of index terms. The lexical analysis phase produces candidate index terms that may be further processed, and eventually added to indexes. It also helps split the longer sentences into smaller chunks of the dataset to perform algorithms with better accuracy.

*E. Removal of Stopwords*

English stopwords such as is, has, an, the, etc. do not signify any importance as index terms when analyzing the dataset for information. It is crucial to remove the stopwords from the dataset as they do not help us find the true meaning of a sentence and can be removed without any negative consequences. Also, eliminating such words from consideration early in automatic indexing speeds processing, saves huge amounts of space in indexes. It has been recognized since the earliest days of information retrieval that many of the most frequently occurring words in English (like “the,” “of,” “and,” “to,” etc.) are worthless as index terms. A search using one of these terms is likely to retrieve almost every item in a database regardless of its relevance, so their discrimination value is low.

*F. Stemming*

It is a technique for improving retrieval effectiveness and reducing the size of indexing files is to provide searchers with ways of finding morphological variants of search terms. The stem need not be identical to the morphological root of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. It is a method for casting words into their original form which aims to the removal of inflectional endings from words. It performs morphological analysis on the words by returning the words into its dictionary meaning. For example, the stemming converts caring into care, troubled into trouble, geese into goose, etc.

*E. Database*

In addition, the skeletal messages and a set of keywords (e.g., the initial 10 words or phrases for the input layer) have to be saved in a database for reconfiguration later. The database includes three tables: (a) keyword table, (b) message table, and (c) message-keyword table. Fig. 3 shows sample values of database, of which kid, mid, and mkid are the primary keys of the tables keyword, message, and message-keyword-result, respectively. The mkid is the foreign key of the table message, and mid and kid are the foreign keys of the table message-keyword-result. The count is the number of the occurrences of the keyword in the messages. The next points to the next keyword in the message, and it is not needed at this moment, but is saved in case.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | KID | keyword | count | | 1 | covid-19 | 116 | | 2 | cdc | 28 | | 3 | vaccine | 41 | | 4 | credit card | 11 | | … | … | … | | k | sign up | 9 | |  | |  |  |  |  | | --- | --- | --- | --- | | MKID | MID | KID | next | | 1 | 4 | 3 | 2 | | 2 | 4 | 21 | 28 | | 3 | 4 | 122 | **–** | | 4 | 10 | 30 | 6 | | 5 | 46 | 32 | 49 | | 6 | 10 | 8 | 7 | | 7 | 10 | 21 | 118 | | 8 | 122 | 30 | **–** | | 9 | 55 | 8 | 72 | | 10 | 41 | 3 | 61 | | … | … | … | … | | n | 87 | 138 | 9 | |
| (a) |  |
| |  |  |  | | --- | --- | --- | | MID | MKID (start) | result (class) | | 1 | 103 | misinformative | | 2 | 26 | true | | 3 | 52 | misinformative | | 4 | 1 | disinformative | | … | … | … | | m | 94 | fake | |  |
| (b) |  | (c) |

Fig. 3. Database tables used: (a) keyword table, (b) message table, and (c) message-keyword-result table.

# self-reconfigurable dataflow graph

There are four phases to build the proposed system mentioned in Section III, where the Phase III is to reconfigure the dataflow graph used by the system automatically. A discussion of the reconfiguring algorithm is given in this section.

*A. Self-Reconfiguring Phase*

The initial input words or phrases of the dataflow graph are found by heuristic. Therefore, the results of the first round of testing are usually not satisfactory because the selected words are usually too subjective or some critical words are not selected. The self-reconfiguring phase tries to solve the problem partially at least. Fig. 4 shows the control/data flow diagram of the self-reconfiguring phase. The skeletal messages are the messages after being preprocessed by the system including lexical analysis, stopword removal, stemming, etc. After one round of training and testing, the database saves the skeletal messages and their discovered results/classes, which are fed to the reconfiguring algorithm described in the next sub-section. The algorithm is to produce better input words for the dataflow graph based on the results of the first round of training and testing. The graph is retrained by the same set of training data, but with different input words. It is expected the reconfigured dataflow graph will generate more accurate results during testing.

Skeletal messages

Results

(classes)

Reconfiguring algorithm

(Reconfigured) dataflow graph

Database

Fig. 4. The data/control flow of the self-reconfiguring phase.

*B*. *The Self-Reconfiguring Method*

The self-reconfiguring algorithm is to find better input words or phrases of the dataflow graph for the testing after the first round of training and testing. The idea is to replace the initial input words by better words based on the first-round results because the system will have a better idea what the good input words are supposed to be after training and testing. Figs. 5 and 6 give the proposed reconfiguring algorithm, which is to replace the input words based on the numbers of keyword occurrences and the LCSs (longest common subsequences) of the testing results and messages. LCSs are well known, but let’s briefly review here. Assume the two strings are given as follows:

*L1 = disease control prevent announce covid-19 omicron vaccine*

*L2 = vaccine control prevent disease illness omicron sickness symptom*

Their LCS is as follows according to its name, longest common subsequence:

*L=L1\*L2 = control prevent omicron*

Fig. 5 shows the first part of algorithm including the following steps: training, testing, calling the second part of algorithm described in Fig. 6, and retraining the dataflow graph. Afterward, the system is ready for use and better results are expected with the revised input words.

|  |
| --- |
| **Dataflow-Graph (** training[200], testing[200] **)**  **Input:** 200 training messages and 200 testing messages  **Output:** Reconfigured dataflow graph G’   1. (Training phase) Train the dataflow graph G with 200 known messages, training. For each message, submit the 10 input words and notify the Graph G what the supposed output class is. The Graph will organize its connection weights based on the inputs and outputs. 2. (Testing phase) Test the Graph G with 200 unknown messages, testing, to find their classes, results. 3. Save all data (including training, testing, and results) in database. 4. G’ ← Self-Reconfiguring( G, database ) 5. (Training phase) Re-train the dataflow graph G’ with 200 known messages, training. 6. **return**( G’ ) |

Fig. 5. First part of the algorithm of the self-reconfiguration method.

Fig. 6 shows the second part of the self-reconfiguring algorithm which takes the keyword occurrences and the LCSs of the testing results into consideration. Each of the five result sets (true, fake, misinformative, disinformative, and neutral) has a set of messages. The algorithm collects the top 20 keywords and retrieves the LCS from each set. Subsequently, collect the top 10 words after merging the keywords, LCSs, and the initial input words. The final 10 words will be new input words of the dataflow graph.

|  |
| --- |
| **Self-Reconfiguring (** G, database **)**  **Input:** graph G and database including messages and results  **Output:** reconfigured dataflow graph G’   1. There is a set of messages for each of the 5 result classes set[5] from the results[200] of database. 2. final ← ϕ 3. **for each** i **in** set[i] **do** 4. k[20] ← Find-Top-Keywords( set[i] ) 5. l ← | Find-LCS( set[i] ) | top 10 words   // message is one of the messages in set[i]   1. final ← final ∪ | (l∈message) ∪ k[20] | top 10 words 2. **end for**   // initial is the initial 10 input words or phrases   1. final ← | final ∪ initial | top 10 words 2. G’ ← G revised with the 10 new input words 3. **return**( G’ ) |

Fig. 6. Second part of the algorithm of the self-reconfiguration method.

# Experiment Results

In order to prove the method is effective, experiment results are provided in this section to validate the claim.

*A. System Setup*

A prototype system is built to prove the proposed method works. The following major software and tools are used to help the system construction:

* *Keras* [22], which is an open-source software library that provides a Python interface for artificial neural networks. It acts as an interface for the TensorFlow library.
* *TensorFlow* [22], which is a free and open-source software library for machine learning and artificial intelligence.
* *Xamarin* [24], which is a cross-platform app development platform that helps to build a single app for all the device systems.

The prototype system can be found at GitHub [25] including the following three files:

* *Round robin*, is the strategy used for TensorFlow. In this approach, the model gets the input parameters in a Round-robin fashion from each of the target labels. This is a combination of LCS or repeated words for each label.
* *Classification bucket*, which is the classification model used for TensorFlow. In this model, we assign two buckets for each classifier. Each bucket gets the words from a combination of LCS or repeated words. If the input message contains the words in the bucket it will set the input value is 1. 0 otherwise.
* *Source data*, which contains more than 4,000 text messages for training and testing.

*B. Experiments*

Fig. 7 shows the experiment setup for testing our system. The proposed system actually could be located at either client or server, but its construction should be done at the server for convenience.

**Server side**

Update

Result

Build

Test

**Client side**

Training data

The proposed system

Smartphone

User

Database

App

Fig. 7. Another view of the proposed system for experiments.

Fig. 8.a shows the proposed app displayed on the Android launcher window. After the user clicking on the app icon, a list of text messages is shown in Fig. 8.b.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| (a) |  | (b) |

Fig. 8. (a) the app on an Android launcher and (b) selecting which message to check.

*C. Evaluations and Discussions*

Fig. 9 shows the accuracy before and after the self-reconfiguration. Both of the accuracy and consistency improve after the reconfiguration. However, the consistency still has room for improvement.

Chart, line chart

Description automatically generated

Fig. 9. The accuracty before and after the self-reconfiguration.

Table 1 shows the evaluation data from testing 74 mobile pandemic messages randomly collected from various sources such as the Internet and online short message archives. It shows our method significantly improves after self-reconfiguration.

Table 1. Evaluation data from testing 74 mobile text messages.

|  |  |  |  |
| --- | --- | --- | --- |
| **Message Type** | **# of Messages** | **# of Correct Detection** | **Accuracy** |
| True | 20 | 15 | 75% |
| Neutral | 20 | 14 | 70% |
| Misinformative | 6 | 4 | 66% |
| Disinformative | 8 | 5 | 62% |
| Fake | 20 | 14 | 70% |
| Overall | 74 | 52 | 70% |

Identifying misinformation is intrinsically difficult. People are not able to tell whether the information is correct easily, let alone computers. The following observations are noticed:

* The accuracy is acceptable, but still not consistent. It may be because the information provided by the messages is limited. To fix the problem, more information needs to be discovered from the messages.
* This self-reconfiguration applies to the input words of the dataflow graph and one time only. Better results may be achieved if the reconfiguration can apply to other layers and more than once.
* The evaluation data may be bias because the messages are filtered beforehand and may not be generic enough. To be fair, the proposed method should be compared to other methods if possible.

# Conclusion

Smartphones have become the major news and information sources for people. During the coronavirus pandemic, mobile users receive numerous text messages about the disease and its cures. Most messages are helpful and useful, but some of them are incorrect intentionally or unintentionally. Incorrect health information not only gives wrong news, but may also causes harmful results. This research proposes a self-reconfigurable system for identifying health text misinformation, so mobile users will take appropriate actions based on the findings. When an incoming message is received by the system, it is processed as follows. The message is converted into a skeletal message by the preprocessor of the system including lexical analyzer, stopword remover, and stemmer. A dataflow graph is used to find the class (true, fake, misinformation, disinformative, or neutral) of the skeletal message. In addition, after the first round of testing, the system will use the results to reconfigure itself and expect to receive better results.

Experiment results show the proposed method is effective and satisfactory. However, there is still room for improvement. For example, the sequence information is basically ignored in this research. Consider the meaning of the sequence “Center of Disease Control and Prevention” is very different from the one of “Disease Prevention and Control Center,” but they have the similar keywords. Using RNN (recurrent neural networks) to handle the sequential data will be considered next. The ANN is considered because this problem has no definite answers. For example, a message may be considered true for some people, but others may think it is disinformative, especially if it is related to politics, and ANN is competent for this kind of ambivalence. In addition, other than using artificial neural networks to detect misinformation, statistical means will be considered too. The statistical means includes the methods of Bayesian classifiers and hidden Markov models. It is less innovative, but may be more effective. On the other end, the DL may be the most innovative one, has been applied to NLP (natural language processing) for some time, and has received great success. This problem, mobile health misinformation identification, could be classified as one of the NLP problems (Islam, Liu, Wang, & Xu, 2020). We will consider the variety of DL methods and adapt them to our problem, and see whether the results come up. Besides, there has been a rising interest in proactive intervention strategies to counter the spread of misinformation and its impact on society (Sharma, Qian, Jiang, Ruchansky, Zhang, & Liu, 2019). Methods to mitigate the ill effects caused by misinformation will be investigated too.

##### References

1. K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang, and Y. Liu, “Combating fake news: a survey on identification and mitigation techniques,” *ACM Transactions on Intelligent Systems and Technology*, 10(3):21, 1-42, May 2019.
2. X. Zhou and R. Zafarani, “A survey of fake news: fundamental theories, detection methods, and opportunities,” *ACM Computing Surveys*, 53(5):109, 1-40, October 2020.
3. T. Khan, A. Michalas, and A. Akhunzada, “Fake news outbreak 2021: can we stop the viral spread?” *Journal of Network and Computer Applications*, 190, 103112, September 15, 2021.
4. N. Savage, “Fact-finding mission,” *Communications of the ACM*, 64(3), 18-19, March 12, 2021.
5. J.S. Brennen, F.M. Simon, and R.K. Nielsen, “Beyond (mis) representation: visuals in COVID-19 misinformation,” *International Journal of Press/Politics*, 26(1), 277-299, 2021.
6. L. Gupta, A.Y. Gasparyan, D.P. Misra, V. Agarwal, O. Zimba, and M. Yessirkepov, “Information and misinformation on COVID-19: a cross-sectional survey study,” *Journal of Korean Medical Science*, 35(27): e257, July 13, 2020.
7. A. Mian and S. Khan, “Coronavirus: the spread of misinformation,” *BMC Medinine*, 18(89), March 2020.
8. N. Fleming, “Coronavirus misinformation, and how scientists can help to fight it,” *Nature*, 583(7814):155-156, June 17, 2020.
9. P. Ball, and A. Maxmen, “The epic battle against coronavirus misinformation and conspiracy theories,” *Nature*, 581(7809):371+, May 2020.
10. H. Sinha, Sakshi, and Y. Sharma, “Text-convolutional neural networks for fake news detection in Tweets,” in V. Bhateja, SL. Peng, S.C. Satapathy, and YD. Zhang (eds), *Evolution in Computational Intelligence. Advances in Intelligent Systems and Computing*, Springer, Singapore, vol 1176, 2021.
11. B. Ahmed, G. Ali, A. Hussain, A. Baseer, and J. Ahmed, “Analysis of text feature extractors using deep learning on fake news,” *Engineering, Technology, & Applied Science Research*, 11(2), 7001–7005, April 2021.
12. M. Aldwairi and A. Alwahedi, “Detecting fake news in social media networks,” *Procedia Computer Science*, 141, 215-222, November 5, 2018.
13. N. Sitaula, C.K. Mohan, J. Grygiel, X. Zhou, and R. Zafarani, “Credibility-based fake news detection,” in K. Shu, S. Wang, D. Lee, and H. Liu (Eds), Disinformation, misinformation, and fake news in social media, *Lecture Notes in Social Networks*, Springer, Cham, 163-182, June 18, 2020.
14. B. Collins, D.T. Hoang, N.T., Nguyen, and D. Hwang, “Trends in combating fake news on social media – a survey,” *Journal of Information and Telecommunication*, 5:2, 247-266, 2021.
15. B. Guo, Y. Ding, L. Yao, Y. Liang, and Z. Yu, “The future of false information detection on social media: new perspectives and trends,” *ACM Computing Surveys*, 53(4):68, 1-36, September 4, 2020.
16. M.R. Islam, S. Liu, X. Wang, and G. Xu, “Deep learning for misinformation detection on online social networks: a survey and new perspectives,” *Social Network Analysis and Mining*, 10:82, September 29, 2020.
17. S. Kula, M. Choraś, R. Kozik, P. Ksieniewicz, and M. Woźniak, “Sentiment analysis for fake news detection by means of neural networks,” in V.V. Krzhizhanovskaya, et al. (eds), *Computational Science – ICCS 2020. ICCS 2020. Lecture Notes in Computer Science*, Springer, Cham, vol 12140, 2020.
18. U.J. Isha Priyavamtha, G. Vishnu Vardhan Reddy, P. Devisri, and A.S. Manek, “Fake news detection using artificial neural network algorithm,” in K.R. Venugopal, P.D. Shenoy, R. Buyya, L.M. Patnaik, and S.S. Iyengar (eds), *Data Science and Computational Intelligence. ICInPro 2021. Communications in Computer and Information Science*, Springer, Cham, vol 1483, 2021.
19. Visser, J., Lawrence, J., & Reed, C. (2020, November). Reason-checking fake news. Communications of the ACM, 63(11), 38-40.
20. Reddy, H., Raj, N., Gala, M., & Basava, A. (2020, February 18). Text-mining-based fake news detection using ensemble methods. International Journal of Automation and Computing, 17, 210-221.
21. Schuster, T., Schuster, R., Shah, D.J., & Barzilay, R. (2020). The limitations of stylometry for detecting machine-generated fake news. Computational Linguistics, 46(2), 499-510.
22. Sethi, R.J., Rangaraju, R., & Shurts, B. (2019, May 30). Fact checking misinformation using recommendations from emotional pedagogical agents. In A. Coy, Y. Hayashi, and M. Chang, Eds, Intelligent Tutoring Systems, 99-104.
23. Keras, “Keras: the Python deep learning API,” 2022. [Online]. Available: https://keras.io. [Accessed: January 20, 2022].
24. TensorFlow, “An end-to-end open source machine learning platform,” 2022. [Online]. Available: https://www.tensorflow.org. [Accessed: January 21, 2022].
25. Xamarin, “Cross-platform with Xamarin,” 2022. [Online]. Available: https://dotnet.microsoft.com/en-us/apps/xamarin/cross-platform. [Accessed: January 11, 2022].
26. S. E Vadakkethil Somanathan Pilla, “A self-reconfigurable artificial neural network,” 2022. [Online]. Available: https://github.com/sanjaikanth/ANNSelfConfigure. [Accessed: January 12, 2022].