**Source Based Fake News Classification**

Brinda Ashar 1, Avinash Bharadwaj 2 , Parshva Barbhaya 3 , Ruchi Bhatia 4

U.G. Student, Department of Computer Engineering, K. J. Somaiya College of Engineering, Mumbai, Maharashtra, India1,2,3,4

**ABSTRACT**: The spread of misinformation on social media platforms is an ever-growing problem. Post the 2016 US Presidential elections, the rise in the propagation of fake news has been quite troubling. Organizations, politicians, individuals looking for personal gain, and even certain news media outlets engage in propagating fake news to sway people's decisions as well as distorting events to fit a bias or prejudice. The measure of authenticity of the news posted online cannot be definitively measured, since the manual classification of news is tedious and time-consuming, and is also subject to bias. To tackle the growing problem, detection, classification, and mitigation tools are a need of the hour. We present in this paper detailed and comprehensive steps in using Machine Learning models in tandem with Feature Engineering and Extraction processes on a fake news dataset to classify a given news article. The dataset contains attributes like the source of the articles, authors, title, text, etc We will finally show the difference in results gathered, as well as a practical application of the project undertaken.

**KEYWORDS**: Feature engineering, Feature Extraction, Fake news, Fake News classification, Machine Learning

1. **Introduction**

Fake news is a type of yellow journalism, which consists of unethical practices to catch the attention of readers. It is often spread via traditional media (such as newspapers) or by posting online. The misinformation passed, even though marked fake, usually finds its way to thousands and lakhs of people through social media. It is commonly written to mislead the reader, in order to damage the reputation or image of a person, entity, company, or a product.

Fake news is most commonly detected by headlines in large font, arrogant language, lavish use of pictures, use of edited and non-existing pictures, and so on. The frequency of fake news has been observed to increase in the political hemisphere, especially during the time of elections. Since it is easy to access an online platform with more than one lakh people, users take advantage to induce public polarization and create hostile situations by publishing misinformation to gain sympathy and, in the case of politics, votes.

Spreading of misinformation has drastic repercussions, and hence, maximum efforts should be taken to minimise its influence on civilians, as well as to eradicate the sources of these fake news. Our primary goal is to develop an automated system which will tell fake news and authentic news apart. This might not decrease the scale at which falsities spread, but can act as a medium to double-check the authenticity of any suspicious news posted online.

The project and the paper covers the implementation for predicting the accuracy of a news article, and also the scope for future work. Thus the project report consists of the research, implementation, the result of each approach and potential future advancements in the following chapters:

I. Chapter 2: Related work - This chapter presents the technical papers reviewed as a part of a literature survey, which have played an important role in deepening our understanding.

II. Chapter 3: Methodology - In this chapter, the architecture of the system designed is explained, along with the requirement specifications and related diagrams.

III. Chapter 4: Experimental results - This chapter includes the information and pictures of the implemented tools, the data processing and extracting techniques, and the explanation for the Machine Learning models used.

IV. Chapter 5: Conclusion – This chapter concludes the idea behind the project along with indication toward future scope of the same.

1. **RELATED WORK**

*Hunt Allcott and Matthew Gentzkow* [1] primarily focused on highlighting what fake news is and how the content can be relayed from one user to another without any third party filtering or fact checking. Fake news spreads at the speed of wildfire and manages to fool credible news sources at times. The birth of fake news stems from the fact that it is cheaper to provide than precise facts and consumers may enjoy partisan news. The definition and usage of fake news has evolved over time and got a surge in the term’s popularity during the 2016 US Presidential elections. [2] mentions that while it was previously defined as sensational or false information that was disseminated under the guise of news reporting, it became synonymous with the spread of false information. This involves fabricated content, imposter content, manipulated as well as false context that could manipulate and mislead readers. Our thorough research also brought our attention towards satire, hate speech as well as bias. *Karishma Sharma et al* [2] in their paper *Combating Fake News: A Survey on Identification and Mitigation Techniques* have provided a proper breakdown on what exactly constitutes fake news, excluding the likes of satire and rumor, also categorizing methods of identification, viz, 1) Content based identification 2) Feedback based identification 3) Intervention based solutions.

In [10], *Saad et al* went about with the detection of fake news using N-gram analysis and Deep Learning models. Prior to feature engineering, they brought about the importance of preprocessing techniques for the data, which needed to be subjected to certain refinements like stop-word removal, lowercasing, tokenization, sentence segmentation and punctuation removal. Thus, for the machine learning models to be employed, stop word removal, sentence breakdown structures and stemming[8] were observed as well. *Saad et al* also sheds light onto the TF-IDF (Term Frequency - Inverse Document Frequency) : a weighting metric often used in information retrieval and natural language processing. It is a statistical metric used to measure how important a term is to a document in a dataset. Hence, these text preprocessing techniques, along with various classification models like Naive Bayes, SVM, Random Forest classifier and so on are applied to classify a given article into eight categories, i.e bs, bias, conspiracy, hate, state, satire, junksci and fake.

1. **METHODOLOGY**

Among the datasets explored, KaggleFN seemed the most promising for preprocessing, feature extraction, and model classification. The reason is due to the fact that all the other datasets lacked the sources from where the article/statement text was produced and published from. Citing the sources for article text is crucial to check the trustworthiness of the news and further helps in labeling the data as fake or untrustworthy. Features to be considered:

* Author
* Published
* Language
* Site\_URL
* Image\_URL
* Type
* Label
* Title ( generated attribute without stop words after preprocessing)
* Text (generated attribute without stop words after preprocessing)
* hasImage (generated attribute to check if the source contains images)

The dataset has 8 labels by which the articles in the corpus are classified with. These are: bias, bs, hate, state, conspiracy, satire, junksci, fake.

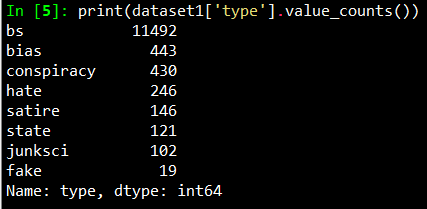


Fig. 1: Data Exploration in kaggleFN

The first step i.e text preprocessing was performed using the following: 1) Taking care of null/missing values 2) Transforming categorical data with the help of label encoders 3) Uppercase to lowercase 4) Removed numbers 5) Tokenization 6) Stop Word Removal, Stemming and Lemmatization (with POS tagging) using the Natural Language Toolkit Library. For feature engineering, TF-IDF technique was used, while for feature extraction, the techniques Word2Vec and GloVe were used. This processed and embedded text is provided as an input to Machine learning models, where the data is made to fit the model, to get a prediction as an output. The models implemented include: 1) Support Vector Machine 2) Naive Bayes 3) Random Forest 4) Logistic Regression 5) AdaBoost 6) Decision Tree 7) Neural Networks.

1. **Experimental results**

The following tables show the results of applying different pre-processing techniques to the Machine learning models:

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Accuracy** | **F1 - score** | **Recall** |
| **SVM** | 0.946 | 0.96 | 0.99 |
| **Naive Bayes** | 0.94 | 0.97 | 1.00 |

Table 1: Metrics Collection in SVM / Naive Bayes implementation

**TF-IDF:**

Right after preprocessing, the output is a corpus of raw texts that has been stripped of stop words, that have been stemmed and lemmatized. In order to get a sparse matrix of TF/IDF values, the following steps are taken:

* Tokenization of texts
* Counting of the tokens and
* Transforming the raw tokens into TF/IDF values.

The TfidfVectorizer performs the above three steps.

TfidfVectorizer in Python transforms text to feature vectors that can be used as input to estimators/classifiers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Partition Ratio**  **ML Models used** | **60:40** | **70:30** | **80:20** |
| **Naive Bayes** | 43.3 % | 41.0 % | 46.0 % |
| **SVM** | 67.4 % | 67.9 % | 68.0 % |
| **Random Forest** | 84.6 % | 81.0 % | 82.0 % |
| **Logistic Regression** | 72.9 % | 75.0 % | 77.0 % |
| **AdaBoost** | 95.6 % | 96.91 % | 95.36 % |
| **Neural Network** | 55.0 % | 54.3 % | 30.0 % |

Table 2: Results of using TF-IDF with Machine learning models

**Word2Vec:**

The word embedding according to the document or text is created by checking the word from the input document with an embedding of all words of the English language already available on GoogleNews. This provides a matrix of vectors as an output that can be provided to Machine Learning models as an input.

|  |  |  |  |
| --- | --- | --- | --- |
| **Partition ratio**  **ML model used** | 80:20 | 70:30 | 60:40 |
| **Adaboost** | 24% | 28% | 25% |
| **Random Forest** | 47% | 51% | 46% |
| **SVM** | 40% | 38% | 39% |
| **Logistic Regression** | 45% | 43% | 45% |

Table 3: Results of using Word2Vec with Machine learning models

**GloVe:**

The first step carried out was importing the pre-trained word vectors publicly available on Stanford NLP.

After iterating through the corpus, word indices of all words were created.

Then an embeddings matrix was created for words present in the training set of our dataset using the pre-trained word vectors. Words were accessed with the help of the word indices.

**Neural Network**

|  |  |  |  |
| --- | --- | --- | --- |
| **Partition ratio** | 80:20 | 70:30 | 60:40 |
| **Neural Network** | 32.44% | 32.08% | 28.82% |

Table 4: Results of using GloVe with a Neural network

**ML models**

|  |  |  |  |
| --- | --- | --- | --- |
| **Partition ratio**  **ML Models** | 80:20 | 70:30 | 60:40 |
| **AdaBoost** | 29.5% | 21.14% | 23.1% |
| **Random Forest** | 48.4% | 47.1% | 45.9% |
| **SVM (linear)** | 42.5% | 42.6% | 41.8% |
| **SVM (rbf)** | 39.9% | 38.8% | 39% |
| **SVM (sigmoid)** | 32% | 30.2% | 32% |
| **Logistic Regression** | 45.1% | 43.1% | 41.6% |
| **Decision Tree** | 39.6% | 37.8% | 36.5% |

Table 5.5: Results of using GloVe with Machine learning models

As seen from the results above, TF-IDF performs the best with the ML models compared to GloVe and Word2Vec.

As a part of implementing the above methodology, the incorporation of the pickle files of the model into a Web app tool was carried out.

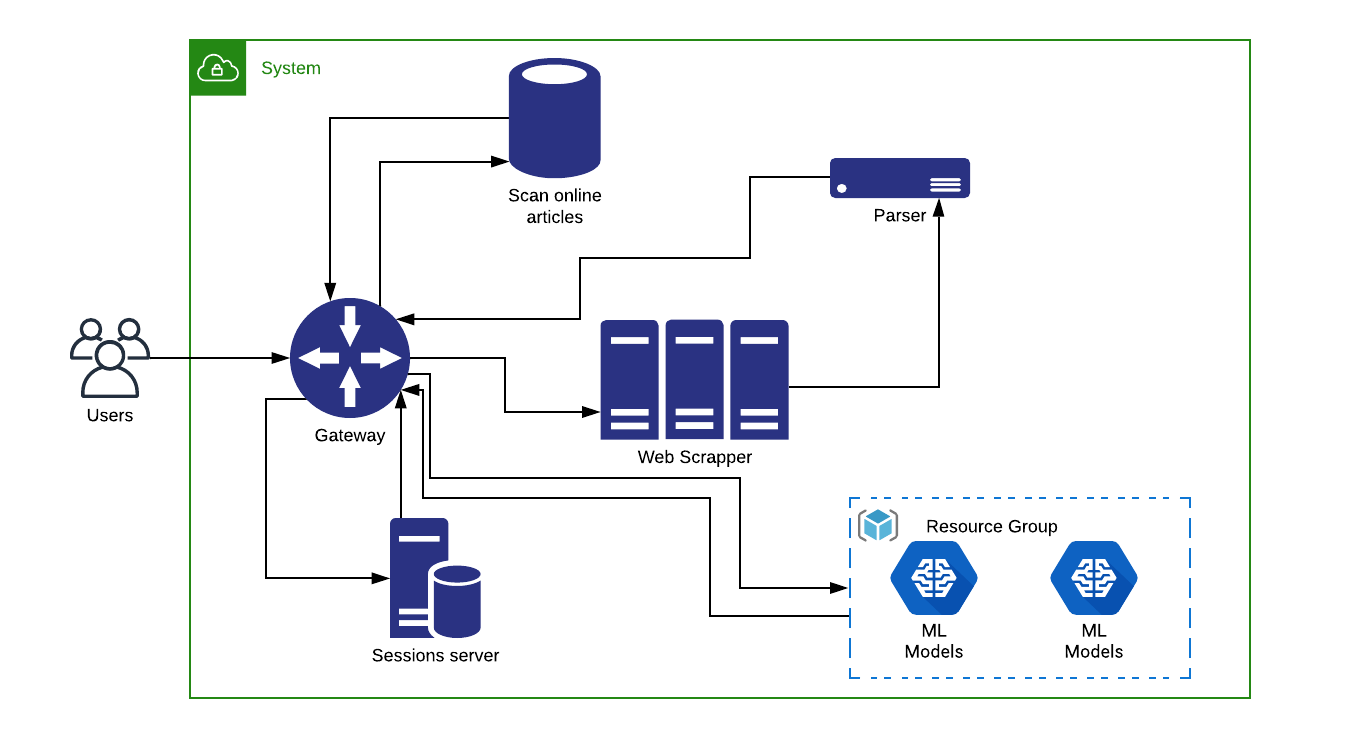


Fig. 2: System Architecture of the tool for Fake News Classification

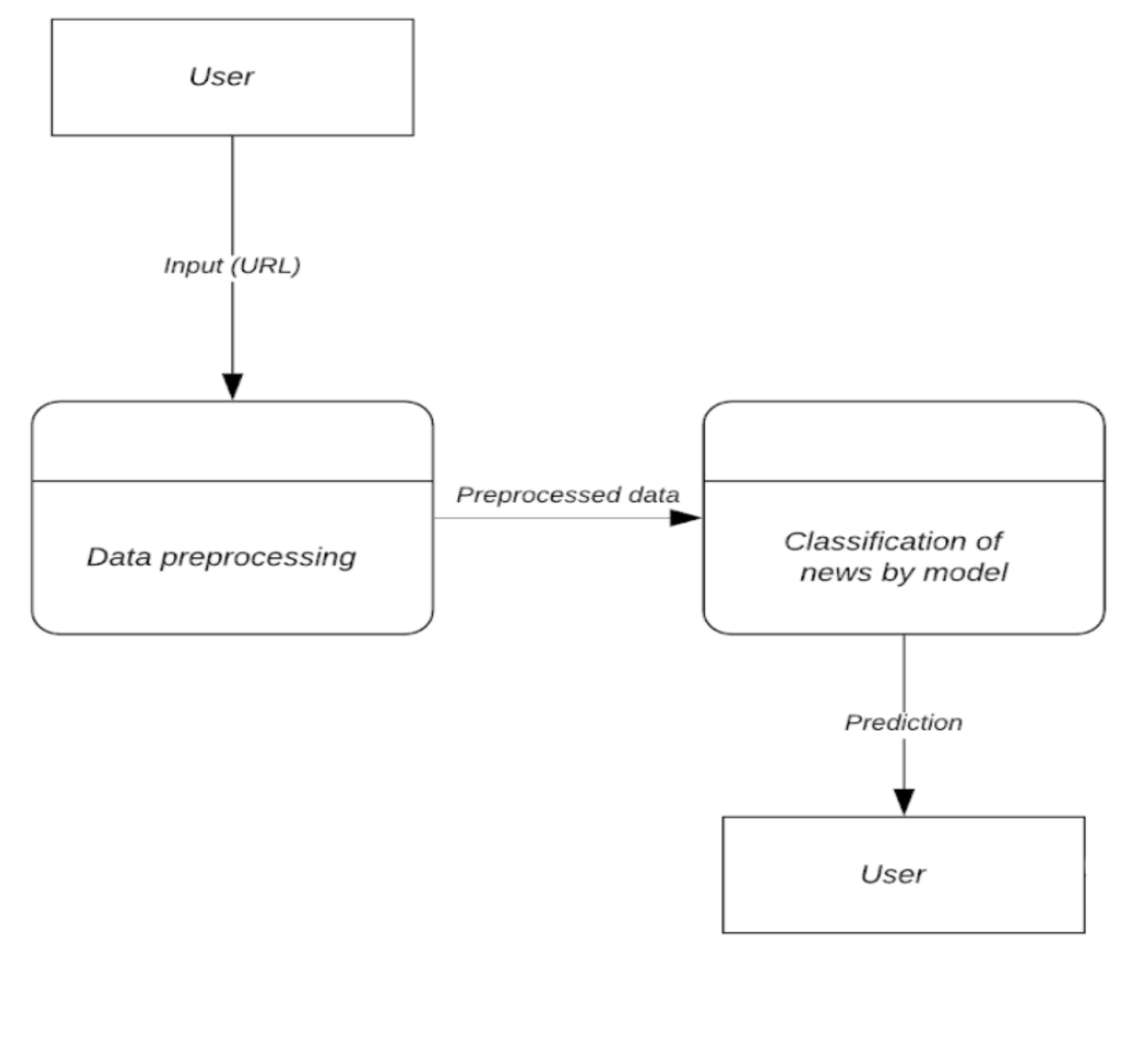
****

Fig. 3: Data Flow Diagram

The work of tool was performed in increments as:

* Building a web scraper, with the ability to parse the content of an article, in a format acceptable for machine learning models.
* Pickling the models developed.
* Converting the previous work to conform to RESTful architecture (done in flask)
* Testing the response of flask REST API using POSTMAN
* Creating, listing and brainstorming endpoints required for web portal
* Starting backend server work in NodeJS
* Basic error handling and maintaining modular files
* Creation of routes and basic testing
* Integration of Flask and JS services
* API Testing using POSTMAN
* Development of User-friendly frontend using ReactJS
* Integration of frontend and backend
* Developing Login functionality
* Showcasing important words listed in article
* Development of Custom search functionality that scrapes through similar online articles
* Refactoring
* Some basic error handling
* Pickling of newer models

To improve accessibility for users, a Chrome extension was created. This extension accesses a database created in Cloud Firestore which contains a list of URLs and a label indicating the type of article on that page.

For the purpose of authenticating a user, users are required to login using their Google account credentials the first time they use the extension. If the login is successful, the user is permitted to make use of the extension to check for the authenticity of a news article. When the user navigates to a news website and clicks on the extension logo, the URL of the active tab is captured and a query is fired which checks whether that article has been classified before or not. Values of the type of articles that have been previously classified are directly returned to the user in a popup window. For other articles, the user is informed that the article is yet to be classified.

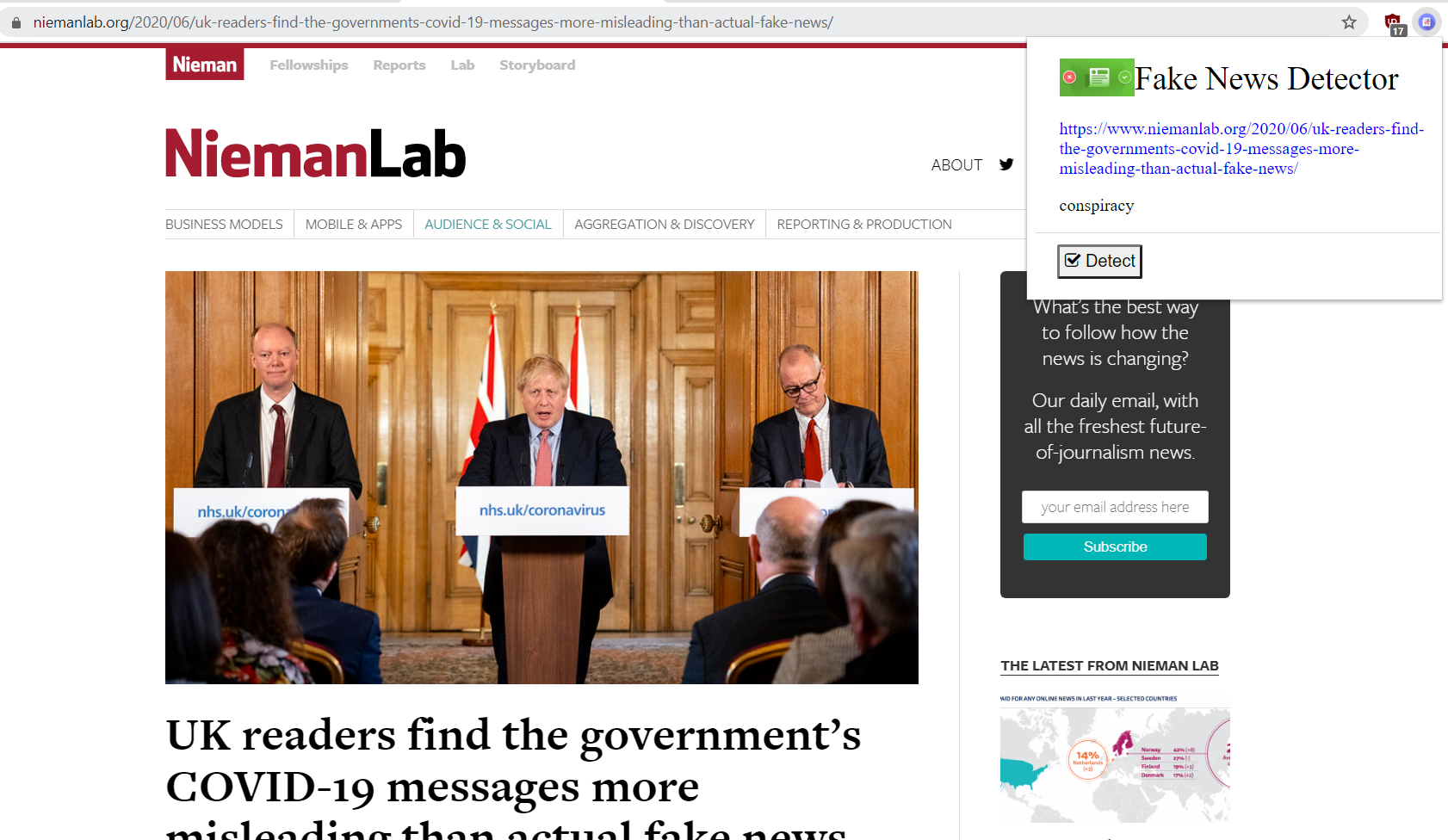


Fig. 4: Chrome extension - For articles that have been previously classified

Some of the major challenges faced in this endeavour include:

* Lack of comprehensive datasets for people to work on and understand the subject better
* Preventing skew in the datasets that would lead to overfitting/underfitting
* Understanding how to grab relevant data as well as how to clean and preprocess data for classification

Like KaggleFN (Getting Real With Fake News), there is a dire need for such comprehensive datasets that require collaboration, curation and crowdsourcing. Further, one of the biggest problems faced was the lack of reliable Indian news article datasets dealing with this issue. In an era where fake WhatsApp forwards and Tweets are capable of influencing naive minds, such tools and knowledge have to be put to practical use in not only mitigating the spread of misinformation but also to inform people about the type of news they consume. Apart from this notion, it is also concluded that the next steps require work pertaining to semantic and syntactic analysis of the text as well as Natural Language Processing methods that work with Deep Learning models. The research related to this project even concluded that promising work has been done on Propagation Trees and Neural Networks, and as such, seems to be the path forward. Development of practical applications for users to gain insight from the articles they consume, fact-checking websites, built-in plugins and article parsers can further be refined, made easier to access, and more importantly, should create more awareness.

1. **conclusion**

The project undertaken provides a comprehensive and detailed approach to the machine learning paradigm towards the topic chosen. Not only the different facets of fake news but also how to identify its sources, how it propagates and what are the possible techniques one can adopt for mitigation was understood. This project also provides a broad understanding of how Machine Learning can be used as a tool to combat fake news. For the uninitiated, this project covers everything, from dataset accumulation, to preprocessing, feature extraction and engineering and finally building accurate classifiers. With the aforementioned implementation, it is possible for people to make practical and rudimentary tools, like the web prototype as well as the Chrome extension to inform users about the articles they read and thus, help them make better decisions.

**References**

1. Hunt Allcott and Matthew Gentzkow. 2017. “Social Media and Fake News in the 2016 Election.” In *Journal of Economic Perspectives—Volume 31, Number 2—Spring 2017—Pages 211–236*
2. Karishma Sharma, Feng Qian, He Jiang and Natali Ruchansky. 2019. “Combating Fake News: A Survey on Identification and Mitigation Techniques.” In *ACM Transactions on Intelligent Systems and Technology, Vol. 10, No. 3, Article 21. Publication date: April 2019.*
3. Yimin Chen, Niall J. Conroy and Victoria L. Rubin. 2015. “Misleading Online Content: Recognizing Clickbait as “False News”.” In *ACM WMDD’15, November 9, 2015, Seattle, Washington, USA.*
4. Yang Yang, Lei Zheng. Jiawei Zhang and Qingcai Cui. 2018. “TI-CNN: Convolutional Neural Networks for Fake News Detection.” In *arXiv:1907.00181v1 [cs.CL] 29 Jun 2019*
5. Lauren Dyson and Alden Golab. 2017. “Exploring the Application of NLP Methods to Machine Identification of Misleading News Sources.” In *Lauren Dyson & Alden Golab, CAPP 30255: Advanced Machine Learning for Public Policy, University of Chicago, Winter 2017*
6. Mykhailo Granik and Volodymyr Mesyura. 2017. “Fake News Detection using Naïve Bayes Classifier.” In *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*
7. Sherry Girgis, Eslam Amer and Mahmoud Gadallah. 2018. “Deep Learning Algorithms for Detecting Fake News in Online Text.” In *2018 13th international conference on computer engineering and systems (ICCES)*
8. Yang-Liu and Yi-Fang Brook Wu. 2018. “Early Detection of Fake News on Social Media Through Propagation Path Classiﬁcation with Recurrent and Convolutional Networks.” In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*
9. Hadeer Ahmed, Issa Traore and Sherif Saad. 2017. “Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques” In *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. Springer, 127–138, 2017*
10. Myle Ott, Yejin Choi, Claire Cardie and Jeffrey Hancock. 2011. “Finding deceptive Opinion Spam by any stretch of the imagination*.”* In *Proceedings of ACL 2011: HLT, pp. 309-319, 2011*