**Fake News Detection**

|  |  |
| --- | --- |
| Report Name | Project Outline |
| Author (User Id) | William Minton (wim12) |
| Supervisor | Richard Jensen (RKJ) |
|  |  |
| Module | CS39440 |
| Degree Scheme | G400 (Computer Science) |
|  |  |
| Date | February 5, 2019 |
| Revision | 0.4 |
| Status | Submitted Draft |

# Project Description

Fake news is a prevailing issue in modern society, being spread extensively across news and social media. Many social media platforms allow for information to be spread rapidly, without any methods of checking if the information is factually accurate, leading to a rise in deliberate misinformation, hoaxes and conspiracy theories. This can occur for political reasons, to purposefully ruin the reputation of others, or simply as a manner of seeking attention, and can have a major impact on the zeitgeist of the time. The spread of misinformation in this manner has been a major focus of the news cycle in the last few years, as seen after both the 2016 US presidential elections, as well as the 2016 EU referendum in the UK.

It is also important to differentiate between the types of Fake News that can be seen and evaluated. Rubin et al [13] suggested that there were three major types of deceptive news; Serious Fabrications, Large-Scale Hoaxes, and Humorous Fakes. Differentiation between these is important in various Data Driven and Natural Language Processing models such as Text classification and Stance Detection.

Fake News Detection is a project based around using trained AI systems to evaluate datasets and determine if the data presented is fake or non-fake. It will do this via training a fake-news Classification model on a mixed dataset of fake and real news articles. This is known as a data-driven Text classification approach. The first step of training is feature extraction, which will reduce the raw data (words) into more manageable groups for processing, which will save on computing resources and allow for large amounts of data to be parsed on personal computers. The words will be classified by their score in a predefined dictionary of words, and will be numerically represented by a vector. This is known as the Bag-of-words approach. There are multiple methods of scoring the words, such as counting the number of times each word appears, and calculating the frequency that each word appears in a document out of all the words in the document, with further study defining which will be used.

Term Frequency – Inverse Document Frequency, or TF-IDF may also be used. Term Frequency scores the frequency of the word in the current document, while Inverse Document Frequency scores the rarity of the word across all documents. This offers a natural filtration for common words such as “the”, “of”, and “and”, penalising these common words, and serving to highlight distinct words across the documents. This offers another option other than the bag-of-words approach and allows for further testing and comparison to see the most optimal strategy.

The machine learning algorithm models will then be fed with the feature-extracted training data. Four algorithms will be used, Multinomial Naïve-Bayes, Linear SVM, Ridge Classifier, and Decision Tree. The models will be trained on a training set, tuned using a validation set, and tested using a test set of data. Evaluation will be done using various evaluation metrics, including Accuracy, Precision, Recall, and the F1 Score.

The project is research-based, so deciding on a specific software methodology was fairly difficult. Ultimately, Scrum was decided upon, due to having the benefits of an iterative design process similarly to XP with each sprint, although it also allows for non-standard practices in design and testing. Burndown charts will be used with the product and sprint backlogs, with the total content of the product backlog to be completed in the near future. Sprints will be one-week long, as this allows for a substantial amount of time to be focused on each sprint, without getting distracted with extraneous features or issues within a particular sprint.

# Proposed Tasks

* **Setting up a local environment and version control system using GitHub:** Version control is a necessary system for projects like this. It lets the supervisor to check and monitor progress over time, functions as a way of backing up data, allows for the restoration of previous versions, and grants the ability to work and download information from any computer. Github will be used as a VCS due to utility and familiarity, although other python-based VCS’ could be used such as Bazaar or Mercurial.
* **Creation of product backlog:** Shows what should be done for the product, should be refined over time to accurately show the requirements for this project over the next few months.
* **Creation of first sprint backlog:** Should be done over the weekend in time for the first sprint, starting on Monday. Shows tasks to be completed during first sprint.
* **Investigation and research into previous attempts at similar work:** Self-explanatory – to be added to bibliography as it progresses
* **Investigation and research into ideal datasets that could be used:** Self-explanatory – to be added to bibliography as it progresses. Thought should be put into which dataset to go with early, will affect parameters & possible metadata.
* **Development of trained AI system capable of filtering through fake news:**
* **Development of further machine learning models for comparison and optimisation:**
* **Observe, report and Evaluate progress of trained AI fake news filtration systems:**
* **Preparation and planning for the two demonstrations expected of the major project:**

# Project Deliverables

**Final trained AI system** – The source code, demonstration and examples of how the trained AI system works, as well as the datasets included, the results, and anything else necessary for running and testing the final system

**Documentation** – Providing information on how to use the trained AI system, the Project Backlog and Various Sprint backlogs used during development, and information on installing and using the various third party libraries, as well as any information or statistics logged using programs such as matlab.

**Progress report** – Document that details progress on the project, changes to the initial ideas or parameters presented in the Project Outline, and evaluation of progress and issues associated at this point in development, being finished the week of the mid-project demonstration, and providing an accurate summary of

**Final Project report** – Full report, covering the entire process of research and development of this project, from inception to evaluation. Will reflect the entire scope of the project, including all code, files, test data, research, and evaluation, including statistical methods used to evaluate the final data. Acknowledgement of third party libraries, such as the scikit-learn python library, will also be included.

**Mid-project and final demonstration –** Not a documentation-based deliverable, but still important deliverables at specified points in the development lifespan. Functional and informative presentation, that will show how the AI system works and what the results indicate at that point in development, and what could be done in the future to improve the topic.

# Bibliography

|  |  |
| --- | --- |
| [1] | Github, Inc., 4 February 2020. [Online]. Available: http:://github.com/. |
| [2] | Aberystwyth University, “Aberystwyth-Bangor Common Intellectual Property Policy,” 5 February 2020. [Online]. Available: https://www.aber.ac.uk/en/hr/policy-and-procedure/au-and-bu/intellectual-property/. |
| [3] | R. C. Martin, Clean Code - A handbook of Agile Software Craftsmanship, Pearson, 2009. |
| [4] | K. Jarmal, “Detecting Fake News with Scikit-Learn,” DataCamp, 2017. [Online]. Available: https://www.datacamp.com/community/tutorials/scikit-learn-fake-news. [Accessed 5 February 2020]. |
| [5] | Fake News Challenge, “Fake News Challenge Stage 1 (FNC-I): Stance Detection,” [Online]. Available: http://www.fakenewschallenge.org/. [Accessed 4 February 2020]. |
| [6] | K. Shu, “FakeNewsNet Github Repository,” [Online]. Available: https://github.com/KaiDMML/FakeNewsNet. [Accessed 4 February 2020]. |
| [7] | S. Kumar, “Awesome Fake News - Fake news research github repository,” Github, 9 November 2019. [Online]. Available: https://github.com/sumeetkr/AwesomeFakeNews. [Accessed 4 February 2020]. |
| [8] | F. Pedregos, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhufer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research,* vol. 12, pp. 2825-2830, 2011. |
| [9] | K. Shu, S. Wang and H. Liu, “Exploiting Tri-Relationship for Fake News Detection,” 2018. |
| [10] | K. Shu, A. Sliva, S. Wang, J. Tang and H. Liu, “Fake News Detection on Social Media: A Data Mining Perspective,” 2018. |
| [11] | K. Shu, D. Mahudeswaran, S. Wangm, D. Lee and H. Liu, “FakeNewsNet: A Data Repository with News Content, Social Context and Spatiotemporal Information for Studying Fake News on Social Media,” 2018. |
| [12] | K. Shu, D. Mahudeswaran and H. Liu, “FakeNewsTracker: a tool for fake news collection, detection, and visualization,” *Computational and Mathematical Organization Theory,* 2018. |
| [13] | V. L. Rubin, Y. Chen and N. J. Conroy, “Deception Detection for News: Three Types of Fakes,” in *Association for Information Science and Technology Annual Conference*, St. Louis, Missouri, 2015. |