

Detecting Fake News with Python

by

Team Python

Benjamin Abrahamsen Hagen

Patrick Alfei Sæbø

Mikael Knudsen

Glenn Robstad

Lars Hilden

Submitted for assessment in

UC1ST1104– Studio 1 Final Assessment

at

Noroff University College

1. Introduction.....	3
2. Literature review	3
2.1. FAKE NEWS.....	3
2.2. DATA COLLECTION.....	3
2.3. PRE-PROCESSING AND FEATURE EXTRACTION	4
2.4. MACHINE LEARNING.....	5
2.5. DEEP LEARNING.....	5
2.6. UNDERLYING TECHNOLOGIES – PYTHON PACKAGES	5
2.6.1. <i>Pandas</i>	5
2.6.2. <i>Mathplotlib</i>	6
2.6.3. <i>Seaborn</i>	6
2.6.4. <i>Tensorflow and Keras</i>	6
2.6.5. <i>Re</i>	6
2.6.6. <i>Io</i>	7
2.6.7. <i>Sci-kit Learn</i>	7
3. Design and Implementation.....	8
3.1. PROGRAM EXECUTION MODEL	8
3.2. DESIGN	8
3.3. IMPLEMENTATION.....	9
4. Results	9
4.1. MODEL TRAINING RESULTS.....	9
4.2. ACHIEVEMENTS	9
4.3. LESSONS LEARNED AND FUTURE WORK	10
5. Conclusion	10
5.1. CONCLUSION	11
5.2. GROUP'S REFLECTION ON THE PROJECT.....	11
References	13

1. Introduction

This report represents the capstone project in the subject Studio1 on the first year of an undergraduate degree program in Applied Data Science at Noroff University College. The grading in the subject is fully dependent on this project. During an intensive 3-week period in May-June 2022, we have been researching the field of fake news and investigated methods for separating fake news from real news using the Python programming language. The project deliverables consist of three parts, namely this report, a video presentation and an artifact in form of a piece of software executing a deep learning model. The project deliverables and requirements associated with them are designed for testing the level of proficiency within the topics covered in Studio1.

This project aims to study fake news as a phenomenon, including technical steps required to identify them using machine learning and deep learning, and ultimately explore a program example of such technology based on the Python programming language.

Initially, a literature review was conducted to understand the subject and help define the scope of the project. Initial focus was on the nature of fake news whereon the area of interest gradually shifted towards data collection and pre-processing of data before finally looking at methods for machine-learning and deep-learning. The literature review was completed with a study of the origin and documentation of the Python libraries used in the project artefact.

Based on the research performed, it was concluded that deep-learning approaches to fake news detection is currently the best available technology. From this conclusion, it was decided to proceed exploring a deep-learning model for the project artefact. The building blocks for the model was collected from Kaggle and combined into a working piece of code that employs a Recurrent Neural Network (RRN) that investigates a news article and concludes if it is fake or real with a certain degree of confidence. In the design and implementation chapter, a block diagram of the program execution is presented along with a description of the steps performed by the deep learning model.

2. Literature review

2.1. Fake news

Fake news is a wide term. It is defined by Dictionary.com (2022) as "news stories that contain false or misleading information, used for some sort of gain, that could be revenue, political gains, discrediting public figures, spreading false information (Propaganda) to gain support for a cause. ETC.". In the US, there have been Gallup polls stating that the trust in mainstream media was at 40%. In the United Kingdom, the trust was at 59% (de Oliveira et al., 2021). Studies have shown that exposure to fake news is associated with a decline in confidence in mainstream media (Ognyanova et al., 2020). Research and tests indicate that humans could only identify 54% of fake news (de Oliveira et al., 2021). Fake news has a massive impact on our society. It impacts humanity's trust in the government and creates polarisation within communities, impacting the economy. There was misleading and false information regarding Barrack Obama, stating he got injured in an explosion. The information caused a variation in the stock exchange of 130 billion USD (de Oliveira et al., 2021). In the 2016 US presidential election, many automated Twitter accounts spread misinformation, and these bots targeted users in a specific group to try and influence the election (Bovet & Makse, 2019).

2.2. Data collection

False information has been around for a long time, the new aspect is the speed with which it spreads. Virginia Woolf wrote in the 1938-novel *Three Guineas* as cited in Asr and Taboada (2019): "If you want to know any fact about politics, you must read at least three different papers, compare at least three different versions of the same fact, and come in the end to your own conclusion."

Present day's identification of fake news can be done manually (e.g. through professional fact checkers) or with aid of automatic information retrieval techniques, natural language processing (NLP) and machine learning (ML) (de Oliveira et al., 2021). When exploring automatic tools for fake news detection, the first challenge is to identify representative instances of fake news. Such instances can come in many forms such as fake product reviews, content written by amateurs that are recruited to promote a specific idea, or simply legitimate authors that are fabricating stories for various reasons. Many datasets that attempt to collect and classify fake news have been made in the past. Some classifications mentioned are *propaganda*, *satire*, *hoax*, *rumour*, *clickbait*, and the obvious *trusted* (Asr & Taboada, 2019).

Numerous fact-checking sites and networks exist (CSI Library; Wikipedia, 2022a, 2022b). One suggested approach is to take advantage of these websites and collect fake news classifications from them using web scraping (Asr & Taboada, 2019).

Various APIs exist both for information retrieval from verified news sources e.g. (The Guardian, 2022; The New York Times, 2022) and live fact-checking based on algorithms operating behind the API. WikiCheck is an algorithm with API-interface performing fact-checking based on Wikipedia (Trokhymovych & Saez-Trumper, 2021). The news API from Webz.io (2022) serves news based on more than 1 million sources in 76 languages in a structured way that is easily readable for machines. The Google Fact Check Tools allows independent fact checkers to contribute to a large dataset of fact checks that can be accessed by researchers and fact checking services (Google, 2022).

Data collected through various sources is raw data and have to be processed and structured in order to provide value to an analysis algorithm.

2.3. Pre-processing and feature extraction

The primary purpose of data pre-processing is to transform raw data into well-formed sets of data that can be analysed. Raw data is unformatted, incomplete noisy, and inconsistent. The success of any project involving the analysis of data is related to the quality of the data prepared. Due to the exponentially growing data generation and the increasing number of heterogeneous data sources, there exists a high probability of obtaining anomalous or incorrect data.

The production of accurate models and, ultimately, accurate predictions are dependent on high-quality data. Therefore, it is imperative that data be processed in the most efficient manner possible. The pre-processing of data is a vital component of data science, machine learning, and artificial intelligence (Joby, 2021).

The term *pre-processing* refers to the process of transforming raw data into a useful, understandable format. Data in the real world is typically formatted inconsistently, contains human errors, and is often incomplete. Data pre-processing can resolve such issues by increasing the completeness and efficiency of datasets. This process is crucial to the success of a machine learning project because the models will perform better when they are able to discover knowledge from datasets faster (Joby, 2021). In most cases *Raw data*, contains several flaws such as inconsistencies, values out of range, missing data, noise, and excessive amounts of data. Consequently, raw data must be processed at various stages during pre-processing to increase the quality of the data. There are three main types of pre-processing methods, namely *data cleaning*, *data reduction*, and *data transformation* (Çetin & Yıldız, 2022).

In the real world, data contains noisy or even missing values. These data have to be handled in a process referred to as *data cleaning* where such values are identified and dealt with in order to avoid inaccurate analysis results and unreliable decisions.

Data produced by various sensors and applications today are growing rapidly on both a row and column basis. In addition to increasing complexity and prolonging the time it takes to obtain results, it may also prevent extraction of accurate information due to its unnecessary and irrelevant contents. *Data reduction* is the process of reducing the size of such data without affecting the data quality in the process. In order to achieve this, *feature selection* and *instance selection* methods should be used (Çetin & Yıldız, 2022).

Noise, missing values, and unnecessary features are eliminated by using raw data cleaning and data reduction methods. However, the resulting new dataset is not yet appropriate for further analysis. The data must be structured into a high-performing format through a *data transformation*. Two such methods are *normalisation* and *data aggregation*.

2.4. Machine learning

Machine learning is a subtype of artificial intelligence, closely related to computational statistics that perform decisions and predictions without explicit programming. Machine learning algorithms can build models and improve their performance on some set tasks by leveraging sample data. The learning process can be divided into two broad categories, supervised and unsupervised learning. Supervised learning relies on human-labelled data, while unsupervised learning attempts to discover patterns in untagged data allowing for more complex processing (Mishra et al., 2022).

The most common approach to unsupervised learning is clustering, which segregates data into clusters based on semantic characteristics. Unsupervised machine learning algorithms are better suited for real-world fake news applications due to the difficulty of gathering and labelling network data. Because these algorithms do not have to rely on structured data, they also show significant potential in real-time fake news detection (Zhang & Ghorbani, 2020). One of the more proficient unsupervised models was the Bayesian graphical model based on sentiment analysis by extracting user opinions and credibility by Yang et al. (2019). This model outperformed previous attempts at unsupervised machine learning models and simple supervised models such as support vector machine and naïve Bayes. Despite the potential of unsupervised machine learning algorithms in fake news detection, little research has gone into its further development.

Much of the research on fake news detection utilises supervised machine learning algorithms. Supervised algorithms are further classified into regression algorithms predicting a continuous value and classification algorithms classifying discrete values. In the subject of fake news detection, classifying algorithms is the most proficient as the algorithm aims to deduce whether the given information is true or false (de Oliveira et al., 2021). A support vector machine is an algorithm using support vectors to calculate the correct hyperplane for classifying a given dataset. This method has been used in several fake news detection studies, such as in Mukherjee et al. (2021) utilising linguistic features from yelp reviews and was shown by Elmurngi and Gherbi (2017) to outperform other supervised machine learning methods. Other standard supervised machine learning algorithms used for fake news detection are K-Nearest Neighbors, Decision Tree, and Naïve Bayes.

The biggest issue with traditional machine learning algorithms is that the performance only scales according to data size up to a certain threshold. With the exponential increase in data volume over the past years, more sophisticated algorithms now have the potential to increase accuracy and overall performance of fake news detection models. *Deep learning algorithms* have the ability to process massive amounts of complex data, as such, excelling in more complex problems like natural language processing. These algorithms also provide the benefit of being less dependent on feature engineering. This has led to decreased interest in traditional machine learning, and an increase in the research on deep learning methods (Ansar & Goswami, 2021).

2.5. Deep Learning

Deep learning is a subset of machine learning, consisting of a neural network with three or more layers. These networks try to simulate the behaviours of the human brain, allowing it to learn from enormous amounts of data. A neural network with a single layer can make an approximate prediction, but additional layers can help to improve the network. Deep learning is the cornerstone of many artificial intelligence applications, such as digital assistants and self-driving cars (IBM, 2022).

Deep learning algorithms can learn without any pre-processing, but much like human learning, it is usually easier when sample data is clear and noise-free. The pre-processing steps are used for classification and prediction purposes, such as in this fake news project where 1 equal real and 0 equals fake news (Matsuo et al., 2022).

2.6. Underlying Technologies – Python packages

The following sub-sections briefly discuss the various python packages/libraries used in the implementation. A general description of the core functionality, including a relevant historical review of the package's origin, is followed by a brief description of the specific functions utilised in our implementation.

2.6.1. Pandas

Python's *Pandas* package provides fast, flexible, and expressive data structures for dealing with relational and labelled data. It provides the building blocks for performing practical, real-world data analysis in Python (Pandas, 2022). The library was

initially developed in 2008 at AQR Capital Management for financial data analysis applications by Wes McKinney. Today it plays a significant role in exploratory data analysis and many other use cases. The name derives from multidimensional panel data, a term used in statistics and econometrics (McKinney, 2011).

In this project, Pandas helps create data frames from csv-files and processes them before introducing the data into the machine learning model. After loading the data into the data frame, it is possible to find missing data values (NaN), null values, and the correlation between features and perform statistical analysis on the data before storing the data frame in the desired format.

2.6.2. Matplotlib

Matplotlib is a Python library for visualising data. It provides a variety of tools for displaying data and tools for creating 2D plots from lists or arrays of data. Matplotlib is one of the most potent libraries in Python for plotting data (Kumar, 2021). Hunter (2007) initially developed Matplotlib to visualise human electrocorticography (ECoG) data for his research in neurobiology.

In this project, Matplotlib creates scatter plots, subplots, bar plots, and histograms and applies many other visualisation techniques to visualise the data and specific features of the data set.

2.6.3. Seaborn

Seaborn is a Python library created by Michael Waskom for generating statistical graphics. It is tightly integrated with Matplotlib and operates on Pandas data structures. It works on whole datasets and internally performs semantic mapping and statistical aggregation to produce graphical plots. Seaborn allows the developer to focus on the meaning of the elements in the plots rather than on how to draw them (Waskom, 2021a, 2021b).

In this project, the Seaborn library is used to plot a heat map from the confusion matrix generated by Sci-kit learn.

2.6.4. Tensorflow and Keras

Google initially released Tensorflow in 2015. It is an open-source software library used for machine learning and artificial intelligence. Keras is an API for deep learning written in Python that runs on TensorFlow. It was initially created as a part of project ONEIROS and supported multiple languages, but at a later stage, it only kept TensorFlow. TensorFlow offers many different packages and allows for start to finish usage and machine learning development. Keras is one of the best APIs for machine learning and model trainers (Keras, 2022a; TensorFlow, 2022a).

Several tools and technologies in TensorFlow and Keras were used in this project. Pre-processing was performed using the *tokenizer* tool for transforming the text into sequences to perform text analysis, followed by indexing and sequencing to ensure that all sequences in the list have the same length (Keras, 2022b). The words were then added to a layer based on the specific density of the word. This allows the machine to weigh the word and learn what it means.

Three parameters for optimising the TensorFlow Keras model are *batch size*, *iterations* and *epoch*. The batch size refers to the number of samples propagated through the network. The advantage of processing the dataset in smaller batches is that less memory is required for training the neural network. This is especially relevant if the dataset is larger than the memory available in the computer. The disadvantage of using small batch sizes is that the estimate of the gradient becomes less accurate with fewer data. Iterations are the number of batches required to process the entire dataset. Epoch is the cycle of processing all samples in the dataset. For example, if the dataset contains 1000 samples and the batch size is 100, it will take 10 iterations to complete 1 epoch (StackExchange, 2022; StackOverflow, 2022a, 2022b).

2.6.5. Re

The Python re package provides the ability to work with regular expressions and was initially released by Secret Labs AB in 1998 (Python, 2022c). Regular expressions are patterns formed by combining characters that can be matched against

corresponding text patterns (Python, 2022b). The regular expression syntax used in the Python re package is based on Perl's regular expression syntax, a programming language written by Larry Wall in 1987 (Usenet, 1988).

In this project, the substitution `re.sub()` function is used for text replacement in the pre-processing stage (Python, 2022b).

2.6.6. Io

The python io module (Python, 2022a) facilitates stream and buffer handling through input and output operations. Each object within the module is called a file object and provides the ability to interact with file-oriented APIs.

In this project, the `io.open()` function is used for reading and storing files to disk.

2.6.7. Sci-kit Learn

Sci-kit Learn is a machine learning library Started by David Cournapeu as a Google Summer of Code project, later joined by Matthieu Brucher, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel. Developed to have simple and efficient tools for predictive data analysis (Scikit-Learn, 2022).

Sci-kit Learn was to prepare the training data by setting 'text' as the feature to extract and 'class' as the target, where 1 equal 'real' news, and 0 equals 'fake' news. The data is then split into training and testing, where the testing is set to 20% of the total articles. The `random_state` argument is a seed number which enables reproducibility.

Sci-kit Learn was also used to create a confusion matrix which is used to evaluate the accuracy of the classification.

It was also used to calculate accuracy, precision and recall. Accuracy means that value from `predicted_set[x]` must match the corresponding value in `true_set[x]` after all values is checked, the algorithm calculates an accuracy score based on the total number of correct predictions.

$$\text{correct predictions} / \text{total predictions} = \text{accuracy score}$$

Precision is the ability to not label fake news as real news. The score is calculated by taking the true positives divided by true positives plus false positive.

$$\text{tp} / (\text{tp} + \text{fp}) = \text{precision score}$$

Recall is the ability to not label real news as fake news. The score is calculated by taking the true positives divided by true positives plus false negatives

$$\text{tp} / (\text{tp} + \text{fn}) = \text{recall score}$$

3. Design and Implementation

This section presents a deep learning model based on a Recurrent Neural Network (RRN). A block diagram is used to illustrate the functionality and execution sequence, followed by a discussion on the design and implementation of the model.

3.1. Program execution model

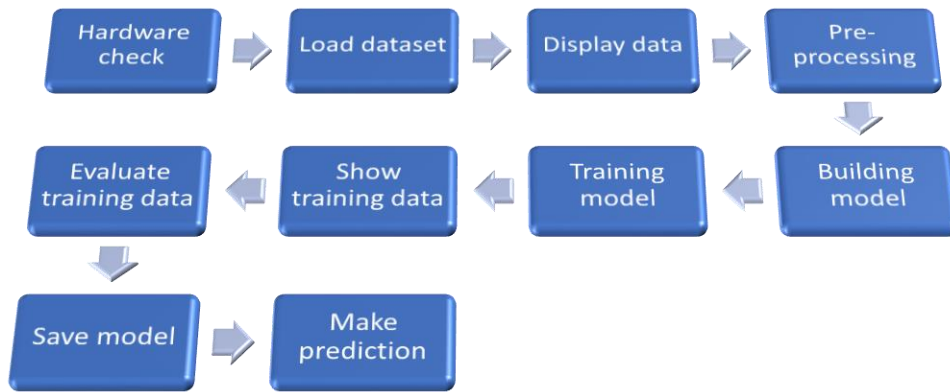


Figure 1 Block diagram of program execution

Hardware check Tensorflow automatically chooses either CPU or GPU if available, to use CUDA a NVIDIA GPU card with CUDA architectures 3.5, 5.0, 6.0, 7.0, 7.5, 8.0 and higher is required. **Loading dataset** Load real and fake news from 2 separate csv files with Pandas. **Display data** Pandas dataframe to check for null and unique values, matplotlib to display fake/real news distribution ratio on the dataset. **Pre-processing** Vectorizes the text with Tokenizer from keras, then normalise the data by removing extra spaces, non words, urls using regular expressions from "re" package. **Building the model** Uses keras to create a sequential bidirectional long short term memory (LSTM) recurrent neural network. **Training the model** Uses tensorflow / Keras to train the model based on parameters set. *Epochs* and *Batch* size are easily changed. If GPU is available, it should take a few minutes. The optional early stop variable, stops the training when validation loss no longer improves (no benefit to further training). *Batch* size means that for every Batch the network is recalculated, smaller Batch sizes require more training time, but 16-32 seems to provide the best accuracy. Epochs is similar to how many training rounds. **Show training data** Matplotlib shows figures for how the network improves after more epochs, until the optional early stop variable stops the training. **Evaluate testing data** Runs the model against testing data, to test accuracy, precision and recall on new data. using sklearn it can calculate how accurate the model is in theory when the testing data is similar to the training data. **Saving the model for re-usability** Saves the model with `model.save("name_of_model_file")` the model can be loaded with `tf.keras.models.load_model("name_of_model_file")` which can be seen on the top of the artefact-code in the try block. **Make prediction** Can copy paste in an article which gets pre-processed and vectorised and then the model makes a prediction whether it is fake or real news.

3.2. Design

The design process was highly influenced by trial and error, testing and trying to modify different implementations from Kaggle.com. A decision was made to use a deep learning model based on Bian (2021) as the research in this project concludes that deep learning approaches are the best option for predicting fake news. Bian's implementation came with a relatively large dataset of 116 MB with structured data in two separate csv files and the code provided excellent accuracy on the testing set.

Initial problems related to incompatible library versions in the development environment took some time to remedy by rolling back to older versions of the libraries in the development environment.

During the model's training, TensorFlow initially used CPU instead of GPU to process the training data. This resulted in low processing performance and long training times. Nvidias deep neural network library cuDNN (Nvidia, 2022) was added to the development environment in order to utilise the GPU. An increase in processing speed of about 10 times was observed as a result of this change.

3.3. Implementation

After finishing the experiments with the model, the program code was converted to a function to re-use the pre-processing stage on any news article imported for analysis. Further, functionality for saving and loading the trained data model was implemented. This allows for re-using the prediction model on different news articles without performing the training process every time. Finally, a function was made to import a news article by copying the text from a web page or document. The function would in turn send the article as a string as input to the prediction model.

The program yielded good results on the training data supplied and was able to predict the occurrence of real and fake news with a precision of nearly 80%. However, It was observed that recent news, especially related to Covid19 and the war in Ukraine was flagged as fake, even though they came from reliable sources. This is probably because the training data used by the model does not contain samples related to these events.

4. Results

This section presents the results of the literature study and design and implementation.

4.1. Model training results

Different batch and epoch parameters were tested to observe changes in the model's performance. The early_stop variable stops the code when no improvement from training is observed, so the epoch parameter could be set to 999 and still stop early. In a small experiment, it was found that a batch size of 64 and epoch with value 7 yielded the best prediction and overall result with a precision of nearly 80%, while the initial results from Bian (2021). were close to 75%.

Batch	Epoch	Time in seconds	Accuracy	Precision	Recall	Sum
30	6	250	0.7567	0.7480	0.7466	0.7504

Table 1 The original result from Bian's example

Batch	Epoch	Time in seconds	Accuracy	Precision	Recall	Sum
16	5	383	0.7967	0.7371	0.8218	0.7852
64	7	142	0.7937	0.7978	0.7787	0.7900
128	7	81	0.7775	0.7116	0.8031	0.7640
60	7	151	0.7501	0.7385	0.7409	0.7431
45	5	167	0.7807	0.8311	0.7432	0.7850

Table 2 The results of our testing of different epoch/batch values

4.2. Achievements

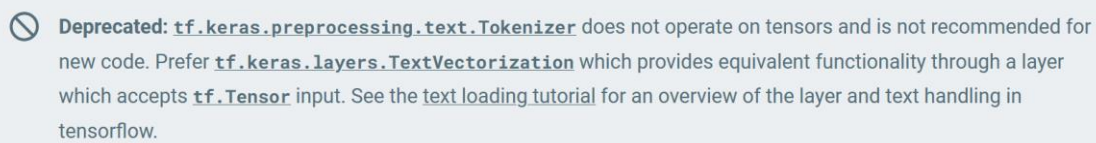
A limited literature study on topics covering the nature of fake news and various technologies used to identify them was conducted. According to (Ansar & Goswami, 2021) deep learning algorithms are superior to traditional machine learning algorithms when it comes to scalability and the ability to process massive amounts of data with less need for feature engineering. This finding led us to study a deep learning model for fake news detection as the artefact for this project.

As the artefact in this project, a working program for a deep learning model employing a Recurrent Neural Network was successfully created based on earlier works by Bian (2021). The performance was good on the accompanying training data

and yielded a precision of nearly 80%. The program was improved for fast re-use of the training model without re-training every time the program is executed. Additionally, functionality for importing news from other sources for analysis was implemented.

The model by Bian (2021) came with a dataset of 116 MB of training data that performed well on the model. However, when introducing recent news articles from legitimate sources about events like Covid19 and the Ukraine war, the model flagged the articles as fake news and performed very poorly. The dataset used should probably have been much larger with a continuous stream of new training data added. This would probably allow for increased accuracy when analysing recent news articles. Due to the complex nature of deep learning and the limited time available, it was impossible to modify the model further by adding more training data and performing adjustments.

In the TensorFlow Keras documentation (TensorFlow, 2022b), the Tokeniser function is deprecated and should be replaced by TextVectorization. New technologies evolve rapidly, rendering older technologies obsolete. The program example by Bian (2021) is less than a year old, but has already been outmaneuvered by newer functionality, thus rendering the implementation in this project obsolete in the process.



ⓘ **Deprecated:** `tf.keras.preprocessing.text.Tokenizer` does not operate on tensors and is not recommended for new code. Prefer `tf.keras.layers.TextVectorization` which provides equivalent functionality through a layer which accepts `tf.Tensor` input. See the [text loading tutorial](#) for an overview of the layer and text handling in tensorflow.

Figure 2 Warning message on using TextTokenizer

4.3. Lessons learned and future work

During the deep learning model design, problems related to incompatible library versions in the Python environment were observed. This caused the software to malfunction and caused interruptions in the design process. Troubleshooting the issue found that specific libraries had to be rolled back to an earlier version to work correctly. The lesson learned from this experience is that Python programs can be pretty picky on the library version. Thus, upgrades of the packages in the Python environment should be done with caution on systems that are in production.

TensorFlow initially used CPU instead of GPU to process the training data during the model's training. By taking measures to utilise the GPU, an increase in processing speed of about 10 times was observed. The lesson learned from this observation is to explore the possibilities that lie within using the GPU or other dedicated machine learning hardware on the system.

A future improvement would be to add new training data to the model in order to achieve better predictions on recent news. This would in turn probably require additional parameter tuning in order to achieve the best results.

5. Conclusion

This project aimed to study fake news as a phenomenon, including technical steps required to identify them using machine learning and deep learning, and ultimately explore a program example of such technology based on the Python programming language. According to the research conducted, deep-learning approaches are currently the most efficient means of detecting fake news.

A literature review of the nature of fake news was performed, followed by a study on data pre-processing, machine learning and deep learning. According to (Ansar & Goswami, 2021) deep learning algorithms are superior to traditional machine learning algorithms when it comes to scalability and the ability to process massive amounts of data with less need for feature engineering. Based on this finding, it was decided to explore a deep learning model utilising a recurrent neural network. An example from Bian (2021) was used as a basis for the software design. The model was trained with accompanying training data and further functionality was added in order to reuse the trained model and analyse news articles from external sources.

During the design of the software, it was observed compatibility issues with the Python libraries used by the example model. This was remedied by reverting some packages to older versions in the Python environment. During training of the model, it was found that it is possible to tune the prediction model for better precision than the original author had done. However, when introducing recent news articles, the model would perform poorly.

5.1. Conclusion

This project has been an educational journey with dedicated team members exploring fake news as a phenomenon and tools used to detect them with the aid of machine learning and deep learning, including supporting activities like data cleaning and feature extraction using the Python programming language. It was found that uncritical library upgrades in Python can lead to compatibility issues. Finally, it can be concluded that the domain of Deep Learning is very complex, rapidly evolving and heavily relies on training data of good quality.

5.2. Group's reflection on the project

Having completed this report, we have completed the capstone assessment for Studio1 in the first year of an undergraduate degree program in Applied Data Science at Noroff University College. During a three-week period in May and June 2022, our team examined methods of separating fake news from authentic news using Python programming language.

The group members have obligations in addition to the studies, and it was challenging to find timeslots where everyone could be available at the same time. The group scheduled two meetings per week on Wednesday and Sunday evenings to assess the progress and distribute tasks to the group members.

The project deliverables, tasks and progress were managed using a mind-map. Initially, all deliverables and requirements were structured in the mind-map. This helped a lot when defining the report template and video presentation structure. The meeting minutes were recorded as the mind-map to document the meeting and the decisions made. During meetings, tasks were defined, assigned to the person responsible for the task and recorded in the mind map.

OneDrive was used for collaboration and file-sharing. One user created the root folder in the file structure and shared this folder with the other parties. All literature was shared in a dedicated sub-folder, and the document file names were renamed to reflect the in-text citation used in the report. Co-writing in MS Word documents stored in OneDrive was a seamless experience where all participants could work on their sections of the document while they could see what the other parties were working on. All references were gathered in one reference handling system in order to present a complete and correctly formatted reference list in the report.

Conflicts were handled using a one-time negotiation model (Zohar, 2015) where a minimum of time is spent on the negotiation process yielding maximum success for one party while the other parties lose. This approach kept the conflict level low and the discussions short, favouring a focus on the tasks ahead. This approach might have caused the project's scope to extend above what is expected for the assessment.

The assignment is relevant for further studies in many levels: Research skills (locating sources, archival and reference management), file and folder management with a focus on naming conventions, defining a document structure for reports and presentations, teamwork, project management, working towards deadlines.

In addition to the soft skills mentioned above, the assignment yielded lots of interesting technical literature that can be relevant for future work. By adding the references, including documents to a reference manager, the literature will be easily available for future work.

The assignment had tight constraints concerning the report's size and available time. One important lesson learned is to read the assessment and mark instructions carefully to prioritise tasks that yield the most towards the final grade. The programming example does not pay anything in this assessment, except that it is a mandatory deliverable.

If there was additional time available for this project, we would have added more data to the data model, explored the benefits of adding additional pre-processing steps to the model, performed additional training and optimisation models, and explored additional algorithms. Within pre-processing, we especially would like to look closer at the feature extraction methods: feature selection and instance selection, and at the data transformation methods: normalisation and aggregation as they were briefly mentioned in the report.

In terms of collaboration, we would want to use the comment-functionality in MS Word more in order to give constructive criticism and add thoughts, notes, and questions. That would help avoid double work, allow for faster editing and help retrieve the writer's thought process. Adding the mark instructions and assessment details to each chapter would make sure that the chapters would be written with the correct focus from the start, rather than having to edit the chapter.

References

- Ansar, W., & Goswami, S. (2021). Combating the menace: A survey on characterization and detection of fake news from a data science perspective. *International Journal of Information Management Data Insights*, 1(2), 100052. <https://doi.org/https://doi.org/10.1016/j.ijime.2021.100052>
- Asr, F., & Taboada, M. (2019). Big Data and quality data for fake news and misinformation detection. *Big Data & Society*, 6, 205395171984331. <https://doi.org/10.1177/2053951719843310>
- Bian, X. (2021). *Fake News Detection Using RNN*. Retrieved 2022-05-25 from <https://www.kaggle.com/code/therealcyberlord/fake-news-detection-using-rnn/notebook>
- Bovet, A., & Makse, H. A. (2019). Influence of fake news in Twitter during the 2016 US presidential election [Article]. *Nature Communications*, 10(1), 7. <https://doi.org/10.1038/s41467-018-07761-2>
- Çetin, V., & Yıldız, O. (2022). A comprehensive review on data preprocessing techniques in data analysis. *Pamukkale University Journal of Engineering Sciences*, 28(2), 299-312. <https://doi.org/10.5505/pajes.2021.62687>
- CSI Library, C. o. S. I.-T. C. o. N. Y. *Websites for Fact-Checking*. Retrieved 2022-05-24 from <https://library.csi.cuny.edu/c.php?g=619342&p=4310783#s-lg-box-13619375>
- de Oliveira, N. R., Pisa, P. S., Lopez, M. A., de Medeiros, D. S. V., & Mattos, D. M. F. (2021). Identifying Fake News on Social Networks Based on Natural Language Processing: Trends and Challenges. *Information*, 12(1), 38. <https://www.mdpi.com/2078-2489/12/1/38>
- Dictionary.com. (2022). *Fake news*. Retrieved 2022-05-25 from <https://www.dictionary.com/browse/fake-news>
- Elmurugi, E., & Gherbi, A. (2017, Nov 12, 2017 - Nov 16, 2017). *Detecting Fake Reviews through Sentiment Analysis Using Machine Learning Techniques DATA ANALYTICS 2017* : The Sixth International Conference on Data Analytics, Barcelona, Spain. https://www.researchgate.net/publication/325973731_Detecting_Fake_Reviews_through_Sentiment_Analysis_Using_Machine_Learning_Techniques <http://iaria.org/conferences2017/DATAANALYTICS17.html>
- Google. (2022). *Google Fact Check Tools*. Retrieved 2022-05-29 from <https://toolbox.google.com/factcheck/apis>
- Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3), 90-95. <https://doi.org/10.1109/MCSE.2007.55>
- IBM. (2022). *Deep Learning*. Retrieved 2022-05-26 from <https://www.ibm.com/cloud/learn/deep-learning>
- Joby, A. (2021). *What Is Data Preprocessing? 4 Crucial Steps to Do It Right*. Retrieved 2022-05-28 from <https://learn.g2.com/data-preprocessing>
- Keras. (2022a). *About Keras*. Retrieved 2022-05-29 from <https://keras.io/about>
- Keras. (2022b). *Using pre-trained word embeddings*. Retrieved 2022-05-29 from https://keras.io/examples/nlp/pretrained_word_embeddings/
- Kumar, B. (2021). *What is Matplotlib and how to use it in Python*. PythonGuides.Com,. Retrieved 2022-05-25 from <https://pythonguides.com/what-is-matplotlib/>
- Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., Uchibe, E., & Morimoto, J. (2022). Deep learning, reinforcement learning, and world models. *Neural Networks*, 152, 267-275. <https://doi.org/https://doi.org/10.1016/j.neunet.2022.03.037>
- McKinney, W. (2011). pandas: a Foundational Python Library for Data Analysis and Statistics.
- Mishra, S., Shukla, P., & Agarwal, R. (2022). Analyzing Machine Learning Enabled Fake News Detection Techniques for Diversified Datasets. *Wireless Communications and Mobile Computing*, 2022, 1575365. <https://doi.org/10.1155/2022/1575365>
- Mukherjee, A., Venkataraman, V., Liu, B., & Glance, N. (2021). What Yelp Fake Review Filter Might Be Doing? *Proceedings of the International AAAI Conference on Web and Social Media*, 7(1), 409-418. <https://ojs.aaai.org/index.php/ICWSM/article/view/14389>
- Nvidia. (2022). *NVIDIA cuDNN*. Retrieved 2022-05-25 from <https://developer.nvidia.com/cudnn>
- Ognyanova, K., Lazer, D., Robertson, R. E., & Wilson, C. (2020). Misinformation in action: Fake news exposure is linked to lower trust in media, higher trust in government when your side is in power. *The Harvard Kennedy School Misinformation Review*, 1(4). <https://doi.org/10.37016/mr-2020-024>
- Pandas. (2022). *Package overview*. Retrieved 2022-05-25 from https://pandas.pydata.org/docs/getting_started/overview.html
- Python. (2022a). *io — Core tools for working with streams*. Retrieved 2022-05-28 from <https://docs.python.org/3/library/io.html>
- Python. (2022b). *re — Regular expression operations*. Retrieved 2022-05-28 from <https://docs.python.org/3/library/re.html>
- Python. (2022c). *Source Code for Python 3.10 re-package*. Retrieved 2022-05-31 from <https://github.com/python/cpython/blob/3.10/Lib/re.py>
- Scikit-Learn. (2022). *About us*. Retrieved 2022-05-30 from <https://scikit-learn.org/stable/about.html>
- StackExchange. (2022). *What is batch size in neural network?* Retrieved 2022-06-01 from <https://stats.stackexchange.com/questions/153531/what-is-batch-size-in-neural-network>
- StackOverflow. (2022a). *What is a batch in TensorFlow?* Retrieved 2022-06-01 from <https://stackoverflow.com/questions/41175401/what-is-a-batch-in-tensorflow>
- StackOverflow. (2022b). *What is an epoch in TensorFlow?* Retrieved 2022-06-01 from <https://stackoverflow.com/questions/40069524/what-is-an-epoch-in-tensorflow>
- TensorFlow. (2022a). *About TensorFlow*. Retrieved 2022-05-29 from <https://www.tensorflow.org/about>
- TensorFlow. (2022b). *TensorFlow documentation on tf.keras.preprocessing.text.Tokenizer*. Retrieved 2022-05-31 from https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer
- The Guardian. (2022). *The Guardian api*. Retrieved 2022-05-29 from <https://open-platform.theguardian.com/>
- The New York Times. (2022). *The New York Times API*. Retrieved 2022-05-29 from <https://developer.nytimes.com/apis>
- Trokhymovych, M., & Saez-Trumper, D. (2021). WikiCheck: An end-to-end open source Automatic Fact-Checking API based on Wikipedia. *arXiv*. <https://doi.org/10.48550/ARXIV.2109.00835>
- Usenet. (1988). *v13i001: Perl, a "replacement" for awk and sed, Part01/10*
- Perl Kit, Version 1.0 Copyright (c) 1987, Larry Wall*. Retrieved 2022-05-29 from <https://www.tuhs.org/Usenet/comp.sources.unix/1988-February/006962.html>
- Waskom, M. L. (2021a). *An introduction to seaborn*. Retrieved 2022-05-26 from <https://seaborn.pydata.org/introduction.html>
- Waskom, M. L. (2021b). seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60), 3021. <https://doi.org/10.21105/joss.03021>
- Webz.io. (2022). *Good News for Machines*. Retrieved 2022-05-22 from <https://webz.io/data-apis/news-api>
- Wikipedia. (2022a). *Category:Fact-checking websites*. Retrieved 2022-04-24 from https://en.wikipedia.org/wiki/Category:Fact-checking_websites
- Wikipedia. (2022b). *List of fact-checking websites*. Retrieved 2022-05-24 from https://en.wikipedia.org/wiki/List_of_fact-checking_websites
- Yang, S., Shu, K., Wang, S., Gu, R., Wu, F., & Liu, H. (2019). Unsupervised Fake News Detection on Social Media: A Generative Approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01), 5644-5651. <https://doi.org/10.1609/aaai.v33i01.33015644>
- Zhang, X., & Ghorbani, A. A. (2020). An overview of online fake news: Characterization, detection, and discussion. *Information Processing & Management*, 57(2), 102025. <https://doi.org/https://doi.org/10.1016/j.ipm.2019.03.004>
- Zohar, I. (2015). "The Art of Negotiation" Leadership Skills Required for Negotiation in Time of Crisis. *Procedia - Social and Behavioral Sciences*, 209, 540-548. <https://doi.org/https://doi.org/10.1016/j.sbspro.2015.11.285>