Electronics and Computer Science
Faculty of Engineering and Physical Sciences
University of Southampton

Author:
Lukas Kakogiannos

April, 2024

Scientometrics: constructing a machine learning model that predicts the success of destabilising papers.

Project supervisor: Dr. Markus Brede
Second Examiner: Prof Leslie Carr

A project report submitted for the award of
MEng Computer Science

Abstract

The field of Scientometrics is a field that encompasses several key areas of study, such as publication analysis, citation analysis, collaboration networks, scientific mapping, and science policy and evaluation. In this dissertation, an attempt is made to construct a machine-learning model that predicts whether a scientific paper will be destabilising or consolidating at the time it is published. The regression model is trained on a citation network that consists of scientific papers published in a variety of fields.

The model uses the CD index to predict whether the paper is destabilising, a metric ranging from -1 to 1 (-1 being consolidating, 1 being destabilising). The machine learning techniques used to construct the model are: Feedforward Networks, 1D Convolutional neural networks, Recurrent neural networks, Long Short-Term Memory networks, Gated Recurrent Units, Graph Neural Networks, Linear regression, Logistic Regression and Ridge Regression. The best results were achieved with the Ridge Regression model, however, due to the nature of the dataset used, results were not satisfactory, as the dataset contained a class imbalance.

Keywords: Scientometrics, Neural Networks, Regression models, CD index, Ridge Regression, Destabilising, Citation Networks.

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

I have not used any resources produced by anyone else.

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have not submitted any part of this work for another assessment.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

My work did not involve human participants, their cells or data, or animals.

Acknowledgements

I would like to extend my gratitude to my supervisor Dr Markus Brede for his invaluable guidance during our meetings. His deep knowledge of the studied topic as well as his motivation have contributed to the completion of this project.

I would like to express my appreciation to my friends and girlfriend present here with me, whose questions about my project helped me refine my ideas and approach. Their intellectual curiosity is a testament to their dedication to learning, and I am privileged to have had their company throughout my years spent at this university.

Finally, I would like to dedicate my work to my beloved family in Greece who, despite the distance, always felt close thanks to their continuous support and encouragement. Their love, guidance, and sacrifices have been instrumental in my success. Their unwavering belief in me and my abilities has been a source of motivation and inspiration.

Contents

1 Introduction

1.1 Overview

According to the Oxford English Dictionary, "Scientometrics" stands for "The branch of information science concerned with the application of bibliometrics to the study of the spread of scientific ideas; the bibliometric analysis of science."[OED, 2023] It is a scientific discipline that involves the quantitative study of science and scientific research output. It plays a crucial role in providing valuable insights into the structure and evolution of science, aiding researchers, policymakers, funding agencies, and institutions in making informed decisions. It helps in understanding scientific productivity, impact, and trends, contributing to the advancement of knowledge and innovation.

The field of scientometrics encompasses several key areas of study such as publication analysis, citation analysis, collaboration networks, scientific mapping, and science policy and evaluation. With the increase in computational power, scientometrics has experienced an increase in popularity over the past few decades [Bornmann and Leydesdorff, 2014], with publication and citation analysis being the two most prominent areas.

Publication analysis focuses on examining patterns and trends in scientific publications, such as the number of publications, citations, authors, journals, and research topics, while citation analysis focuses on investigating the citations received by articles, authors, journals, or institutions to measure the impact and influence of scientific work. Citation metrics, such as the h-index[Hirsch, 2005], impact factor[Garfield, 1955], and CD index are commonly used to evaluate the significance of research output, with the CD index being the most recent one [Funk and Owen-Smith, 2016].

Moreover, scientometrics uses mathematical, statistical, and more recently, computational methods to analyse various aspects of science. When it comes to publication and citation analysis, computational methods have been used to examine the production, dissemination, and impact of scientific literature [Martin, 2019, Pobiedina and Ichise, 2014, Robson and Mousquès, 2016, Ponomarev et al., 2014], as well as the impact of scientists, and institutions [Dong et al., 2016, Haslam et al., 2008, Yousef et al., 2022]. Such computational methods include the use of different machine-learning techniques to analyse vast amounts of data and aid in prediction models based on mathematics and statistics.

Artificial intelligence is a topic that has been trending in recent years, especially with the release of foundation model-based applications such as chatGPT to the general public. This gain in popularity among the public is a product of an ever-increasing number of scientific papers being released that focus on artificial intelligence, including machine learning [Borghesani and Pedregosa, 2023, Sarker, 2021].

Machine learning focuses on developing a range of models and algorithms, such as linear regression, decision trees, neural networks, support vector machines, and more. These can be used for different purposes, for instance, analysing patterns, relationships, and structures within a dataset. Finally, if a predictive model's training has been completed successfully, it can be used on previously unseen data to make predictions based on its existing knowledge[Sarker, 2021, Géron, 2017].

This report presents a detailed explanation of the goal of the research, followed by the implementation of different machine learning models and their evaluation.

1.2 Project Goals

The goals of this project are:

1. To build a machine learning model that will be used to predict whether a scientific paper (at the time it is published) will be destabilising or not.
2. To analyse the impact that the content of scientific papers has on the nature of future citations.

The scope of this project is:

1. Comparing the performance of different machine learning models for this specific task.
2. Analysing whether the structure of the citation network can be a potential feature used for predictions.
3. Taking the first step in a new direction in the field of scientometrics.

2 Background

2.1 Scientometrics

Over the past few decades, an increasing amount of scientific papers has been released that revolve around Scientometrics, many papers invent new indices to try and measure scientific progress, with the most notable ones being the h-index, proposed by J. E. Hirsch in 2005, the impact factor, mentioned for the first time by Eugene Garfield in 1955, and more recently, the CD index, being published online for the first time in 2016 by Russell J. Funk and Jason Owen-Smith. Recent studies have adopted the use of the CD index[Bornmann and Tekles, 2019, Wu et al., 2019], however, it has been used mainly to examine past bodies of work and analyse their impact on academia over time.

2.1.1 Impact factor

The impact factor is a widely used bibliometric measure that assesses the influence and prestige of scientific journals by quantifying how frequently articles published in a particular journal are cited by other researchers[Garfield, 1955]. The mathematical representation is shown below:

$$\text{Impact Factor (IF)} = \frac{\text{Total Citations in Year}}{\text{Total Number of Articles Published in Year}} \tag{1}$$

A higher impact factor indicates that articles published in that journal are more influential and frequently cited, however, it has certain limitations, for instance, it focuses only on citations within a specific year and does not account for variations in citation patterns across disciplines.

2.1.2 H-index

The H-index (also known as the Hirsch index) is a widely used scientometric indicator that quantifies the impact and productivity of a researcher's work[Hirsch, 2005]. It combines both the number of publications and the number of citations received by those publications to provide a concise measure of a researcher's influence in the scientific community. The mathematical representation is shown below:

$$h = \max\{i : \text{citations}(i) \geq i\} \tag{2}$$

where:

citations(i) = the number of citations for the i-th paper in the list.

For instance, an H-index of 20, means that the researcher has 20 papers with at least 20 citations each. It provides an estimate of the importance, significance, and broad impact of a scientist's cumulative research contribution. Its main limitation is that it is discipline-dependent, and different fields have varying citation practices and publication rates, therefore, comparing H-indices across disciplines may not be meaningful [Dong et al., 2016].

2.1.3 CD index

The CD index is used to quantify the effect of a scientific paper or patent based on whether it consolidates or destabilises previous existing knowledge, the index ranges from -1 to 1, respectively[Funk and Owen-Smith, 2016]. The main factor that is used to calculate whether an academic paper is destabilising or not, is whether future work that cites paper A, for instance, also cites any of paper A's citations. Each future paper is then mapped to a value: -1, 0, or 1, a paper gets a value of -1 when it cites both paper A and a predecessor paper (any work cited by paper A). It gets a value of 0 when it only cites a predecessor paper, and finally, it gets a value of 1 when it cites paper A but not a predecessor paper.

These values are then added according to the following equation (also demonstrated in Figure 1):

$$CD_t = \frac{1}{n_t} \sum_{i=1}^{n} \frac{-2f_{it}b_{it} + f_{it}}{w_{it}} \tag{3}$$

where:

$\mathrm{i} = (i_1, i_2, ..., i_{n-1}, i_n)$

is a vector that contains all future work that cites paper A and/or its predecessors at time t, and

$$f_{it} = \begin{cases} 1, & \text{if i cites paper A,} \\ 0, & \text{otherwise,} \end{cases}$$

$$b_{it} = \begin{cases} 1, & \text{if i cites any of paper As predecessors,} \\ 0, & \text{otherwise,} \end{cases}$$

finally, w_{it} indexes a matrix W of weights for paper i at time t, for an in-depth study, the paper's metadata could be taken into consideration, i.e. the values could vary according to the issue date of i such that older citations have a different impact over the index, or by weighting citations differently depending on the size of the team that made the publication [Wu et al., 2019].

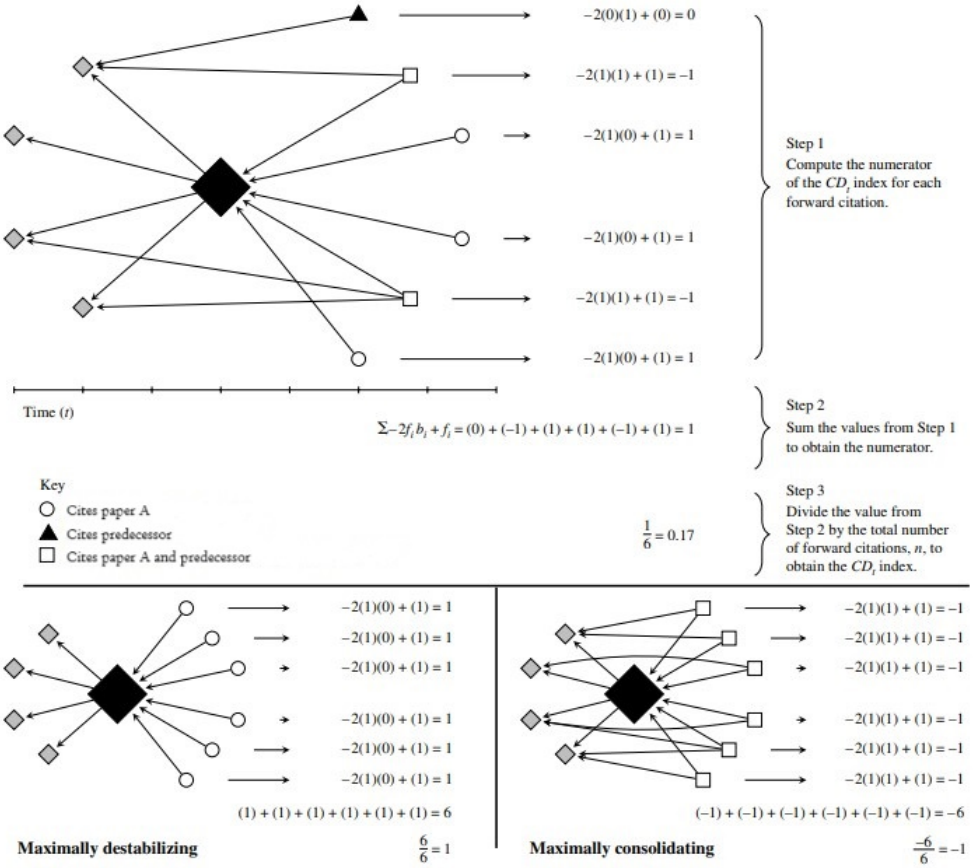


Figure 1: The process of calculating the CD_t index for three different papers. Adapted from [Funk and Owen-Smith, 2016]

However, it is worth noting that this equation captures whether a paper is consolidating or destabilising, but does not take into account the magnitude of its effects. In other words, the CD index does not differentiate between bodies of work that have a greater impact on academia and those that focus on certain niches. To solve this problem, the equation has to be slightly modified by adding a magnitude factor[Funk and Owen-Smith, 2016]:

$$mCD_t = \frac{m_t}{n_t} \sum_{i=1}^{n} \frac{-2f_{it}b_{it} + f_{it}}{w_{it}} \tag{4}$$

where m_t is a parameter that represents the number of future bodies of work that cite paper A but not its predecessors. With this addition, the new index, mCD_t, captures not only the "direction" of an invention's effects but also its magnitude.

Despite the increase in the use of the CD index, a thesis by A. I. Martinez from 2019 utilises the h-index and the impact factor to construct predictive machine learning models. During the study, Bayesian networks were the primary technique used in the development of machine learning models[Martin, 2019].

From the research conducted, CD indices have the following limitations:

- The CD index is sensitive to the underlying network structure, small changes in the network (such as adding or removing edges) can significantly impact the CD index.
- The CD index does not incorporate temporal information, temporal dynamics can impact collaboration patterns[Bornmann and Leydesdorff, 2014].

2.2 Feature Selection

There is a noticeable gap in existing literature, no predictive model has been trained on a dataset that has been filtered based on the CD index of scientific papers. When it comes to predictive models, feature selection is a very important part that can have a significant impact on the model's performance[Zhou, 2021], features are the individual measurable properties or characteristics of the data used for training or prediction[Géron, 2017].

To find possible features that could be used for this model, one can look at existing literature, [Park et al., 2023] demonstrate through a series of comparisons that scientific bodies of work are becoming less destabilising over time, meaning the model could use as a feature the paper's time of publishing, solving one of the CD index's limitations. The same study has a section that analyses the language used in papers from the 1950s compared to papers from the 2010s to showcase the decline in disruptive science over time, this idea could be taken and used solely as a feature for the model, trying to identify words used within the paper that indicate novelty.

Furthermore, another study by [Wu et al., 2019] advocates that larger teams of scientists tend to produce bodies of work that are consolidating, whereas bodies of work published by 1 to 3 people tend to place in a higher average disruptive percentile. This could be further explored, and potentially used as a feature for the machine learning model.

2.3 Machine Learning Algorithms

This section contains a brief outline of the background in machine learning, for clarity, the theory behind each algorithm is explored in its own section.

2.3.1 Neural Networks

Neural network models are inspired by the structure of the human brain's neural networks, consisting of many interconnected nodes which are organised in layers that process information and learn patterns in data[Géron, 2017]. There are three different types of layers in a neural network, the input, hidden, and output layers. The input layer receives the initial data, where each node represents either a feature or an input, the hidden layers are in between the input and the output layer and this is where the computations occur, a single neural network can have many hidden layers, which will affect the model's efficiency, and

executing and training times. Lastly, the output layer is the layer that provides the final result, usually a classification or a prediction[Zhou, 2021].

There exist many types of neural networks, e.g. feedforward neural networks (FNN), convolutional neural networks (CNN), recurrent neural networks (RNN), and long short-term memory networks (LSTM), with each containing its applications and limitations[Sarker, 2021], the report aims to explore how different neural networks can predict the mCD index of a paper based on its features, with content, and citations being the focal points.

2.3.2 Regression Algorithms

Regression algorithms are simpler models, especially when compared to neural networks, they are used to model relationships between input features and numeric outputs[Sarker, 2021]. This is achieved simply by providing the algorithm with true values (labels) for each data point during training, depending on the type of regression (Linear, LASSO, Logistic, Ridge, etc), a different equation is used to model the relationships between inputs and outputs (explored in section 3.3).

3 Methodology

3.1 Dataset Generation

3.1.1 Design

Citation networks play a pivotal role in scholarly communication. When one paper cites another, it establishes a thread that weaves together theories, research, and intellectual lineage. These networks reveal patterns of influence, allowing us to trace the evolution of scientific thought across disciplines and time[CRONIN, 1984]. In a citation network, each node represents a scholarly publication (such as a research paper or article), and the links between nodes are citations, when one paper cites another, a directed connection is established from the citing work to the cited one.

Citation networks serve as a rich source of information for training machine learning models as they can potentially include a plethora of relevant features such as title, abstract, text, keywords, network structure, and temporal information (the publication dates of papers). From these potential features, the two that will be used for training and validation are the text and the network structure.

Initially, the plan was to use Python libraries such as Requests or Scrapy to retrieve the papers necessary to build the citation network from online repositories such as Google Scholar, Nature, and Springer. However, upon researching online, a citation network comprising over 141,000 papers was found[Chenxin et al., 2021]. Before working on the machine learning models, a few pre-processing steps have to be completed.

Firstly, the CD index had to be calculated for each paper based on the formula previously explored. When calculating all of the CD indices, a few observations into the dataset showed that an overwhelming number of papers had a CD index of either -1, 1, 0, or NaN, this happened due to the number of papers that either had a near zero number of references or were referenced by a very low number of papers. To fix this problem, a threshold of 5 was established for both "cited_by" and "references" sections, a number that was enough to get rid of the outliers while still maintaining a dataset of a decent size.

Furthermore, the size of the text is crucial, as machine learning models typically operate on fixed-length feature vectors. Having an inconsistent number of words per text could lead to padding and truncation, padding introduces noise, and truncation discards information. Therefore, any texts that contain less than 1000 words and more than 2500 words were removed.

3.1.2 Implementation

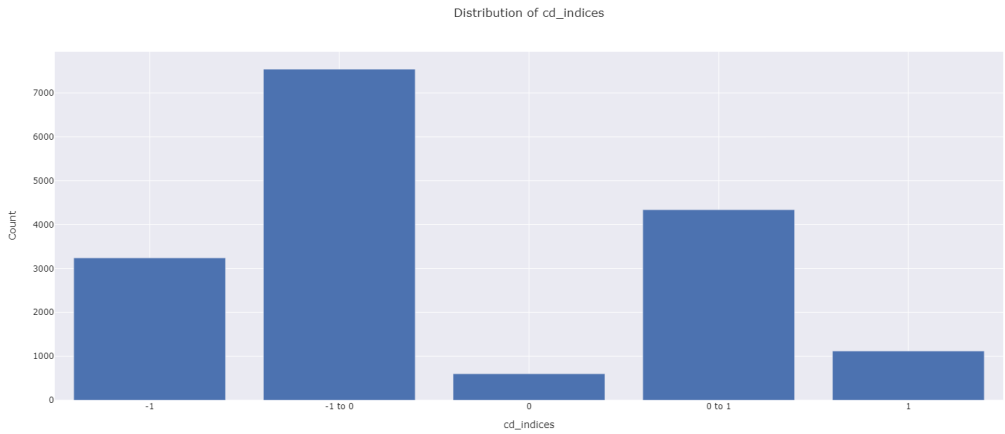
Raw data collected from various sources, whether it's sensor readings, user-generated content, or scientific measurements—often contains noise. This noise can arise due to measurement errors, inconsistencies, or, in this case, irrelevant information. Pre-processing steps aim to reduce or eliminate this noise, ensuring that the data used for training is as accurate and reliable as possible.

In the realm of natural language processing (NLP), pre-processing is critical for working with textual data. In this scenario when working with scientific papers, the pre-processing steps include:

- Tokenisation: It involves breaking down a text into individual units, typically words or subwords. Most NLP tasks require this step in order to prepare the data for further processing.
- Stop word removal: Common words like "the," "and," "in," and "of," known as stop words, appear frequently across documents but often carry little semantic meaning[Wilbur and Sirotkin, 1992]. Their removal streamlines text data, enhances model performance, and focuses on meaningful content.
- Lemmatisation: In linguistic terms, lemmatisation involves grouping together the various inflected forms of a word so that they can be analysed as a single item, identified by the word's lemma or dictionary form. Unlike stemming, which operates on individual words without context, lemmatisation considers the intended meaning of a word within the sentence and its larger context.
- Cleaning: Finally, when going through the dataset, some texts had pointers to images, references, or special characters, such as "inlineform1", "/path / to / image1.tif", and "secref3". For the purpose of this report, these carry little to no semantic meaning, therefore, any words that are not included in the english dictionary were removed from the texts. Moreover, punctuation variations (e.g., "run," "run!", "run.") can lead to multiple representations of the same word, which led to the removal of any punctuation.

3.1.3 Results and Discussion

The dataset was reduced from over 141,000 papers to a total of 16,845 papers, with the text length ranging between 1000 and 2500 words, and an average length of 1707 words. When calculating the mCD index for these papers, the imbalance was too high, as most papers where near zero, this can introduce model bias, meaning it will favor predicting the larger class (majority class) more accurately. Taking that into account for the training of the model, the CD index was used instead. The final CD indices are displayed below:



3.2 Implementing Neural Networks

Numerical representation of data The initial step of the algorithm involves converting raw text data into a numerical representation suitable for machine learning. This can be done in many different ways, including word embedding, TF-IDF (Term Frequency-Inverse Document Frequency), and others. Depending on the choice of neural network deployed, one of the two following approaches is taken:

- Tokenisation and Sequence Padding: Firstly, the text is split into individual words (or tokens), a vocabulary is created by assigning a unique index to each word and each document is represented as a sequence of word indices based on the vocabulary. After converting the texts to sequences, it needs to be ensured that all sequences have the same length. Padding involves adding zeros to the beginning or end of sequences to make them uniform.
- TF-IDF: It captures the importance of words in the documents [Ramos, 2003], term frequency (TF) measures how often a word appears in a document, whereas Inverse Document Frequency (IDF) reflects how unique a word is across all documents. The formula for the weight of a single word w is shown below [Berger et al., 2002]:

$$w_d = f_{w,d} \times \log\left(\frac{|D|}{f_{w,D}}\right) \tag{5}$$

This formula represents the weight (w_d) of a word in a document (d), which is calculated as the term frequency ($f_{w,d}$) of the word in the document, multiplied by the logarithm of the ratio of the total number of documents ($|D|$) to the number of documents where the word appears ($f_{w,D}$).

Dataset Partitioning For all the machine learning model implementations that take textual data as an input, the dataset is split into two smaller datasets, one used for training and one for validation, they are split in a 3:1 ratio, ending up with 12,634 papers in the

training set and 4,211 papers in the validation set, this allows for efficient validation of the model's performance, as it can train on unseen data[Yousef et al., 2022].

3.2.1 Feedforward Neural Network

Theory The feedforward neural network (FNN) was implemented using the Keras Sequential API. Its architecture allows for information to flow uni-directionally from input to output, there are no cycles or loops[Bebis and Georgiopoulos, 1994].

FNNs are trained using the backpropagation method[Bebis and Georgiopoulos, 1994], which fine-tunes the model to minimise prediction errors. Errors between predicted and actual outputs are propagated backwards through the layers, and then, the connection weights are adjusted accordingly. This method is implicitly handled by the Keras library.

Model Architecture The model consists of three layers described in detail below:

- Input Layer: Comprising 10,000 neurons (matching the number of features in the TF-IDF matrix).
- Hidden Layer: A fully connected layer with 128 neurons and a ReLU activation function. The dropout layer (with a dropout rate of 25%) helps prevent overfitting.
- Output Layer: A single neuron with a linear activation function (since this is a regression task).

ReLU activation function The activation function plays a crucial role in transforming the summed weighted input from a node into the activation or output for that input. The ReLU activation function is a piecewise linear function that behaves as follows [Schmidt-Hieber, 2020]:

1. If the input is positive, it directly outputs the same positive value.
2. If the input is negative, it outputs zero.

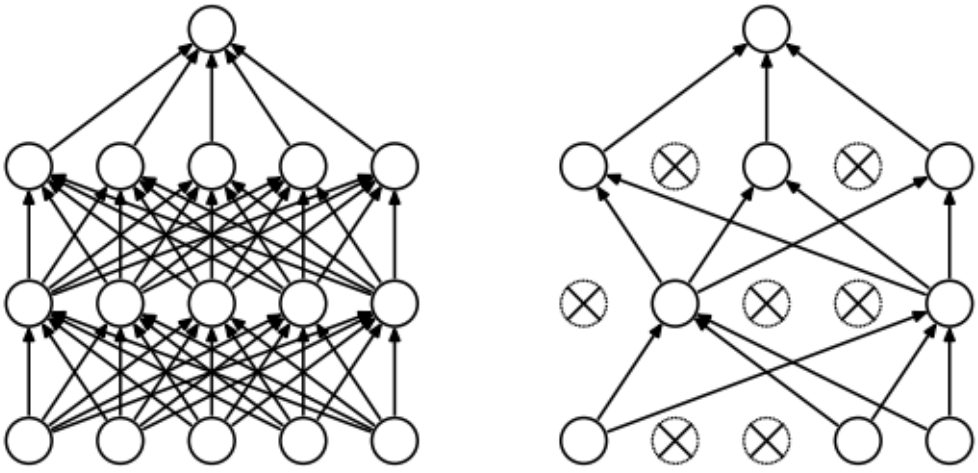
Mathematically, ReLU can be defined as:

$$f(x) = \max(0, x) \tag{6}$$

ReLU was chosen because it overcomes the vanishing gradient problem associated with other activation functions like sigmoid and hyperbolic tangent.

Dropout Layer The dropout layer is a powerful regularisation technique used to combat overfitting [Srivastava et al., 2014] in deep learning models by randomly dropping out or deactivating a subset of neurons during training. These dropped-out neurons do not

contribute to both the forward pass (prediction) and backward pass (gradient update), effectively training on different subsets of neurons, leading to better generalisation (depicted in the figure below).



(a) Standard Neural Net (b) After applying dropout.

Figure 2: Difference between a model that does not use a dropout layer (left) and one that does(right). Adapted from [Srivastava et al., 2014]

Optimiser and loss function The model is then compiled using the Adam optimiser[Kingma and Ba, 2014] and the mean squared error (MSE) loss function (shown below) [Das et al., 2004].

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)^2 \tag{7}$$

where:

n = number of data points

X_i = observed values

Y_i = predicted values

The choice of MSE is suitable for regression tasks, where the aim is to minimise the squared differences between predicted and actual values.

The validation set is used after each epoch to calculate two metrics, accuracy and the MSE loss, after the final epoch, it is also used to calculate the coefficient of determination (R^2)[NAGELKERKE, 1991].

$$R^2 = 1 - \frac{\text{SSres}}{\text{SStot}} \tag{8}$$

where:

SSres = the sum of squares of the residual errors of the data model.

SStot = the total sum of the errors.

Results The FNN is trained twice with two different random states (0 and 1) for 25 epochs (an average of 65 seconds each), the accuracy after 1 epoch was **3.7%** (±0.1) and the loss **0.3500** (±0.01). It is important to highlight that after the seventh epoch, both values stagnated at around **4.2%** (±0.1) and **0.4200** (±0.02), respectively, which they stayed at all the way to the last epoch (even when testing up to 100 epochs). Finally, the average of the R^2 scores was **0.01** (±0.002). These values will be discussed further in section 3.4, together with the results of all the other algorithms.

3.2.2 1D Convolutional Neural Network

Theory A 1D Convolutional Neural Network (CNN) is a variant of the traditional 2D CNN commonly used for image classification, the core operation of this neural network is **Convolution**[Kiranyaz et al., 2021]. Given a 1D sequence (e.g., time series data or text data) as an input, a small filter (kernel) slides over it, at each position, the filter computes a weighted sum of nearby elements, resulting in a new sequence (feature map) that captures local patterns. The kernel is a small window of weights (coefficients) that learns to detect specific features (edges, patterns, etc)[Kiranyaz et al., 2021].

Model Implementation This model uses tokenisation and sequence padding to convert raw text data into a numerical representation, it creates a vocabulary of the 2000 most common words and uses it to generate sequences of indices that point to specific words. The next step is to calculate the average size of these sequences, which is 1500, this number is used to minimise potential noise insertion and data loss from padding and truncation, respectively.

The model consists of five layers described in detail below:

- Embedding Layer: This is the input layer, it converts the sequences to dense vectors (50 dimensions).
- Conv1D Layer: A hidden layer, applies 1D convolution to learn local patterns (64 filters, kernel size 5), similarly to the FNN, it also uses a ReLU activation function.
- Dropout Layer: Similar to the FNN, but with a dropout rate of 12.5%, it helps prevent overfitting.
- GlobalMaxPooling1D Layer: Another hidden layer, it is often used as a downsampling technique before the final output layer, it aggregates information from the entire sequence by computing the global maximum value across the sequence[Tolias et al., 2016].
- Dense Layer: Final output layer, a single neuron with a linear activation function for regression output.

The model is then compiled using the Adam optimizer and the mean squared error (MSE) loss function. The validation set is used after each epoch to calculate two metrics, accu-

racy and the MSE loss, after the final epoch, it is also used to calculate the coefficient of determination (R^2).

Results The 1D CNN is trained twice with two different random states (0 and 1) for 100 epochs each (an average of 700 seconds each), the accuracy after 1 epoch was **3.47%** ($\pm$0.01) and the loss **0.385** ($\pm$0.01). It is important to highlight that after the tenth epoch, both values stagnated at around **3.65%** ($\pm$0.2) and **0.3900** ($\pm$0.05), respectively, which they stayed at until the last epoch. However, the average of the R^2 scores when training with 25 or 50 epochs was **0.01** ($\pm$0.005), but when training up to 100 epochs, it was **0.08** ($\pm$0.01). These values will be discussed further in section 3.4, together with the results of all the other algorithms.

3.2.3 Recurrent Neural Network

Theory A Simple Recurrent Neural Network (RNN) is a type of neural network designed to handle sequential data, they are specifically suited for sequences, such as time series data, natural language, or audio signals[Sherstinsky, 2020, Géron, 2017]. However, unlike feedforward neural networks (where information flows in one direction), RNNs maintain an internal state that captures information from previous steps. At each time step, the hidden layer receives input from both the current input and the previous hidden state, allowing the network to maintain memory of past information[Ghojogh and Ghodsi, 2023].

However, RNNs suffer from vanishing gradients, making it difficult to learn long-term dependencies, this happens because recurrent neural networks update their weights based on the partial derivatives (gradients) of the error function with respect to each weight, during backpropagation, gradients are multiplied together for each layer[Sherstinsky, 2020]. If these gradients are less than 1, their product decreases exponentially with the number of layers. The two neural networks explored in sections 3.2.4 and 3.2.5, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), are advanced RNN variants that address this issue.

Model Implementation Similar to the 1D CNN, this model uses tokenisation and sequence padding to convert raw text data into a numerical representation. The model consists of the following three layers:

- Embedding Layer: This is the input layer, it converts the sequences to dense vectors (25 dimensions).
- SimpleRNN Layer: A basic RNN cell with 128 hidden units and ReLU activation, in this case, ReLU is used because the larger gradients minimise the effect of the vanishing gradient problem.
- Dense Layer: Final output layer with linear activation for regression output.

It is important to note that the use of a dropout layer did not have any significant impact on the results, therefore it was removed.

The model is then compiled using the Adam optimizer and the mean squared error (MSE) loss function. The validation set is used after each epoch to calculate two metrics, accuracy and the MSE loss, after the final epoch, it is also used to calculate the coefficient of determination (R^2).

Results The RNN is trained twice with two different random states (0 and 1) for 20 epochs each (an average of 1400 seconds each), the accuracy after 1 epoch was **3.61%** ($\pm$0.01) and the loss **0.411** ($\pm$0.005). It is important to highlight that after the tenth epoch, values started increasing very slowly, practically stopping at **4%** ($\pm$0.2) and **0.63** ($\pm$0.02). The average of the R^2 scores was **-0.57** ($\pm$0.03). These values will be discussed further in section 3.4, together with the results of all the other algorithms.

3.2.4 Long Short-Term Memory Network

Theory Long Short-Term Memory (LSTM) networks are a type of RNN designed to handle sequential data, they were introduced to address the limitations of traditional RNNs [Zhou, 2021], such as the vanishing gradient problem explored in the previous section.

An LSTM cell consists of three main components:

- Cell State (C_t): The memory that stores information over time.
- Hidden State (h_t): The output of the cell at each time step.
- Gates: Mechanisms that control the flow of information into, out of, and within the cell.

There are three different types of gates:

- Input Gate (x_t): Determines how much new information should be added to the cell state.
- Forget Gate (f_t): Controls what information should be discarded from the cell state.
- Output Gate (y_t): Regulates the flow of information from the cell state to the hidden state.

By learning from its input sequence when to use short-term dependencies and when to use the long-term memory, the LSTM model can effectively retain information over long sequences, while addressing limitations of traditional RNNs.

Model Implementation Similar to the simple RNN, this model uses tokenisation and sequence padding to convert raw text data into a numerical representation. The model consists of the following three layers:

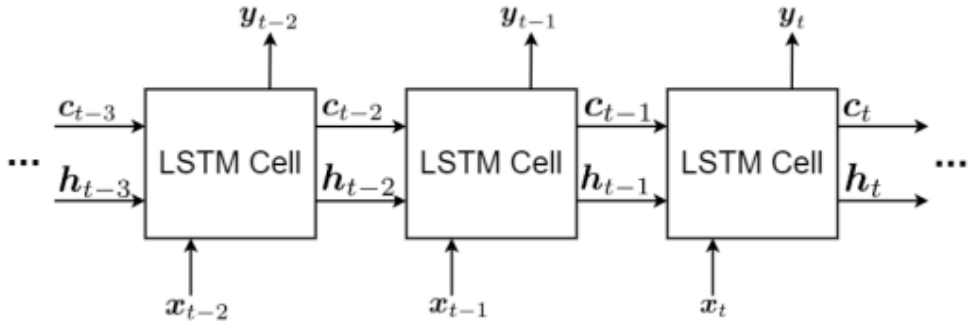


Figure 3: A sequence of LSTM cells processing the input sequence. Adapted from [Ghojogh and Ghodsi, 2023]

- Embedding Layer: This is the input layer, it converts the sequences to dense vectors (25 dimensions).
- LSTM Layer: An LSTM cell with 128 hidden units and tanh activation.
- Dense Layer: Final output layer with linear activation for regression output.

The model is then compiled using the Adam optimizer and the mean squared error (MSE) loss function. The validation set is used after each epoch to calculate two metrics, accuracy and the MSE loss, after the final epoch, it is also used to calculate the coefficient of determination (R^2).

Results The LSTM is trained twice with two different random states (0 and 1) for 20 epochs each (an average of 3840 seconds each), the accuracy after 1 epoch was **3.6%** ($\pm$0.2) and the loss **0.4** ($\pm$0.02). It is important to highlight that after the fourteenth epoch, values started increasing very slowly, practically stopping at **4.3%** ($\pm$0.05) and **0.68** ($\pm$0.02). The average of the R^2 scores was **-0.66** ($\pm$0.04). These values will be discussed further in section 3.4, together with the results of all the other algorithms.

3.2.5 Gated Recurrent Unit

Theory Similarly to LSTMs, Gated Recurrent Units were introduced to address limitations of traditional RNNs. However, the structure of an individual GRU cell differs from that of an LSTM, it consists of[Ghojogh and Ghodsi, 2023]:

- Hidden State (h_t): Represents the output of the cell at each time step.
- Update Gate (z_t): Controls how much past information should be passed to the output.
- Reset Gate (r_t): Determines how much past information to forget.

At each time step, the update gate decides which values to update in the hidden state, the reset gate determines which values to forget, and finally, the new hidden state is computed based on the updated information.

Model Implementation Similar to the RNN and LSTM implementations, this model uses tokenisation and sequence padding to convert raw text data into numerical representations. The model consists of the following three layers:

- Embedding Layer: Converts word indices to dense vectors (50 dimensions).
- GRU Layer: A Gated Recurrent Unit with 128 hidden units and hyperbolic tangent (tanh) activation.
- Dense Layer: Final output layer with linear activation for regression output.

The model is then compiled using the Adam optimizer and the mean squared error (MSE) loss function. The validation set is used after each epoch to calculate two metrics, accuracy and the MSE loss, after the final epoch, it is also used to calculate the coefficient of determination (R^2).

Results The GRU is trained twice with two different random states (0 and 1) for 20 epochs each (an average of 3550 seconds each), the accuracy after 1 epoch was **3.6%** ($\pm$0.02) and the loss **0.41** ($\pm$0.01). It is important to highlight that after the fourteenth epoch, values started increasing very slowly, practically stopping at **4.1%** ($\pm$0.1) and **0.67** ($\pm$0.05). The average of the R^2 scores was **-0.64** ($\pm$0.05). These values will be discussed further in section 3.4, together with the results of all the other algorithms.

3.2.6 Graph Neural Network

Theory Graph Neural Networks are different from the other algorithms explored, instead of learning based on the text data from the scientific papers, they are designed to process and analyse graph-structured data[Zhou et al., 2020], such as citation networks. They leverage the inherent structural information of graphs to learn powerful node and graph representations, enabling them to capture complex dependencies and propagate information effectively across the graph[Zhou et al., 2020].

The core concept in GNNs is message passing, each node in the graph aggregates information from its neighbouring nodes by collecting information from its neighbours and updating its representation. This process allows nodes to learn from their local context within the graph. Furthermore, similarly to CNN layers, GNNs use graph convolutional layers, and the output of a graph convolutional layer becomes the input for the next layer.

When deciding how the design of the GNN pipeline will look like, the following aspects have to be considered [Zhou et al., 2020]:

- Directed/Undirected: whether edges in the graph are directed from one node to another, for this citation network, edges are directed.
- Homogeneous/Heterogeneous: whether edges and nodes in the graph have the same

types or not, for this model, scientific papers are treated as one type of node irrespective of the domain of the paper, and even though in the dataset the citation relations are split in two categories, "cited_by" and "references", with a few lines of code, they can be added to the same list (pointing in the same direction, meaning all edges now point from the main paper to its citations). This means the graph is homogeneous.

- Static/Dynamic: whether input features or the topology of the graph may be affected by time, for this study, this is a static graph.
- Node-level/Edge-level/Graph-level: many different tasks can be done using a graph, node classification, node regression, edge classification and link prediction, graph classification, graph regression, and others. For this study, the main task is node regression.
- Supervised/Semi-supervised/Unsupervised setting: whether the data is labelled, partially labelled, or unlabelled, for this case, it is completely labelled, as all CD indices are calculated before the training of the model.

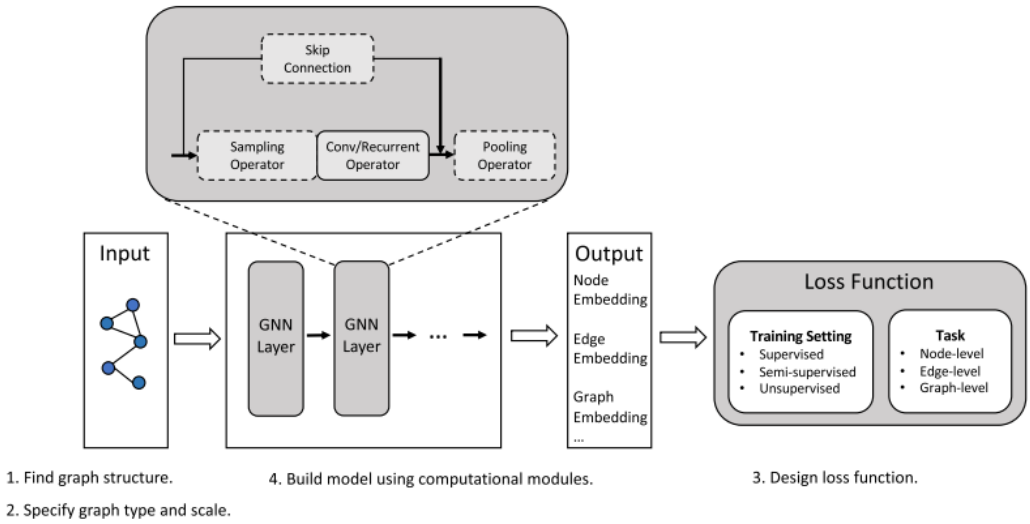


Figure 4: The general design pipeline for a GNN model. Adapted from [Zhou et al., 2020]

Model Implementation When considering all of the above, the model (shown in the code below) consists of the following:

- Two layers (conv1 and conv2) using the GCNConv (Graph Convolutional Network) layer from PyTorch Geometric [pyt, 2024].
- The input features are represented by x, and the edge connections are represented by edge_index.
- The model applies ReLU activation after the first layer and uses dropout for regularisation.
- The final output is obtained by applying log-softmax to the second layer's output.

```python
# Defining the Graph Neural Network model
class GNN(torch.nn.Module):
    def __init__(self, num_features, num_classes):
        super(GNN, self).__init__()
        self.conv1 = GCNConv(num_features, 16)
        self.conv2 = GCNConv(16, num_classes)

    def forward(self, data):
        x, edge_index = data.x, data.edge_index

        x = self.conv1(x, edge_index)
        x = torch.relu(x)
        x = torch.dropout(x, training=self.training)
        x = self.conv2(x, edge_index)

        return torch.log_softmax(x, dim=1)
```

The GNN model is initialised with the number of input features and the number of unique classes (labels), similarly to the other neural networks, the Adam optimiser is used with a learning rate of 0.01. Finally, the model is trained for 100 epochs, using techniques discussed in previous sections like backpropagation.

Results However, upon running the code, it **returned an error** due to the value of the edge_index variable being 0. This leads to the conclusion that even though the dataset contains 16,845 papers, they all reference or are referenced by other papers that are not in this dataset, this happens because, during pre-processing steps, the size of the dataset was reduced significantly. Unfortunately, due to the time constraint of this project, another data set could not be gathered in time to complete the training of the model, this will be discussed in section 5.

3.3 Implementing Regression Algorithms

3.3.1 Linear Regression

Theory Linear regression is a statistical technique used to model the relationship between two quantitative variables, it aims to estimate how a dependent variable (also called the response variable) changes as the independent variables (also known as the predictor variables) change [Montgomery et al., 2021].

The fundamental idea is to find a straight line (a linear equation) that best fits the observed data points, the linear regression equation takes the form [Montgomery et al., 2021]:

$$y = b_0 + b_1 \cdot x \tag{9}$$

where:

y = the dependent variable.

x = the independent variable.

(b_0) = the intercept (the value of y when x is zero).

(b_1) = the slope (how much y changes for a one-unit change in x).

It aims to find the best-fitting line through the data points by minimising the sum of squared differences between the observed values and the predicted values (based on the line).

Results The model utilises a TF-IDF vectorizer to transform the text into numerical representations. The validation set is used after the training to calculate the MSE and the coefficient of determination (R^2), the resulting value for MSE was **0.38** ($\pm$0.01) and for R^2 was **0.08** ($\pm$0.01). These results will be discussed further in section 3.4, together with the results of all the other algorithms.

3.3.2 Logistic Regression

Theory Logistic regression is primarily used to model the probability that an input belongs to a particular category or class. It uses the sigmoid function to model the relationship between the input and output variables. The equation of the sigmoid function is shown below[Géron, 2017]:

$$\sigma = \frac{1}{1 + e^x} \tag{10}$$

where x represents a linear combination of the input features and their respective weights.

The model establishes a linear relationship between the input features and the log odds of the probability of the target class. The linear combination of features (weighted sum) is then passed through the sigmoid function to obtain the predicted probability, which will always be between zero and one. Typically, a threshold is set that dictates whether the sample will be classified as belonging to one class or the other, in this case, whether the scientific paper will be disruptive or not.

Binning However, since Logistic regression is primarily a classification algorithm and the task to be completed belongs to the regression category, bin cutting needs to be done first. It is the process of grouping continuous data into discrete intervals or bins by converting a continuous variable (like age or income) into a categorical variable with meaningful groups [Naeini et al., 2015]. It can be used to create histograms, analyse trends, and simplify complex data. In this case, the CD index variable is split into four different bins described below:

- (-1, -0.5]: Numbers between -1 (exclusive) and -0.5 (inclusive).
- (-0.5, 0]: Numbers between -0.5 (exclusive) and 0 (inclusive).

- (0, 0.5]: Numbers between 0 (exclusive) and 0.5 (inclusive).
- (0.5, 1]: Numbers between 0.5 (exclusive) and 1 (inclusive).

Grid Search Furthermore, for the Logistic regression algorithms, when running the first few iterations of the code, it was observed that the training time of these models is significantly lower than that of the neural networks previously trained. Therefore, grid search becomes a viable option within the time constraints of this project. Grid search is an optimisation technique used to systematically explore a predefined set of hyperparameters (parameters that cannot be learned from the data during training) for a machine learning model [Lerman, 1980], its primary goal is to find the best combination of hyperparameters that yields the highest performance on a given task. Since it brute-forces through all possible combinations of hyperparameters, it can be computationally expensive, especially for large grids, leading to avoiding grid search when training the neural networks. An example of a grid search is displayed in the code below:

```
# Define the parameter grid
param_grid = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000],
    'penalty': ['l1', 'l2'],
    'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']
}

# Create a base model
logreg = LogisticRegression(max_iter=1000)

# Instantiate the grid search model
grid_search = GridSearchCV(estimator=logreg, param_grid=param_grid, cv=5, n_jobs=-1,
    verbose=2)

# Fit the grid search to the data
grid_search.fit(X_train, y_train)

# Get the best parameters
best_params = grid_search.best_params_
print(f'Best Parameters: {best_params}')
```

The best combination of parameters was the following:

$$\text{Best Parameters: } \{\text{'C': 1, 'penalty': 'l2', 'solver': 'liblinear'}\} \tag{11}$$

The three parameters are described in detail below:

The C hyperparameter controls the inverse of the regularisation strength in logistic regression by determining how much the model should penalise large coefficients (weights) during training [scikit-learn developers, 2024]. Higher values indicate less regularisation, allowing the model to fit the training data more closely while smaller values lead to more regularisation, usually opted in simpler models.

The penalty hyperparameter specifies the type of regularisation used in logistic regression and can take one of two values:

- 'l1' (Lasso Regularization): Encourages sparsity by adding the absolute values of coefficients to the loss function. Some coefficients may become exactly zero. [Sarker, 2021]
- 'l2' (Ridge Regularization): Adds the squared values of coefficients to the loss function. It discourages large coefficients but does not force them to be exactly zero. [Sarker, 2021, Cortes et al., 2012]

Lastly, the solver hyperparameter determines the optimisation algorithm used to find the best coefficients during training [Schmidt-Hieber, 2020], different solvers have different convergence properties and computational efficiency:

- 'newton-cg': A quasi-Newton method, supports only L2 regularisation.
- 'lbfgs': Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm, supports only L2 regularisation.
- 'liblinear': Suitable for small datasets; uses coordinate descent and supports both L1 and L2 regularisation.
- 'sag' (Stochastic Average Gradient): Suitable for large datasets; uses stochastic gradient descent, supports only L2 regularisation.
- 'saga': An improved version of SAG that supports both L1 and L2 regularisation.

Results The model utilises a TF-IDF vectorizer to transform the text into numerical representations. The validation set is used after the training to calculate three metrics, accuracy, MSE and the coefficient of determination (R^2), the resulting values for accuracy were **45.5%** (± 0.5), for MSE were **2.12** (± 0.03), and for R^2 were **-0.62** (± 0.03).

Additionally, since after binning this turns into a classification task, we can plot two confusion matrices for each random state that showcase the effectiveness of the algorithm. Confusion matrices provide a detailed breakdown of a model's performance by displaying how many instances were correctly or incorrectly classified into different classes [Zhou, 2021]. This includes True Positives (correctly predicted positives), True Negatives (correctly predicted negatives), False Positives (incorrectly predicted positives), and False Negatives (incorrectly predicted negatives). The two matrices are displayed below:

The results above will be discussed further in section 3.4, together with the results of all the other algorithms.

3.3.3 Ridge Regression

Theory Ridge is a powerful variant of linear regression, its primary goal is to enhance regular linear regression by addressing issues like overfitting and multicollinearity

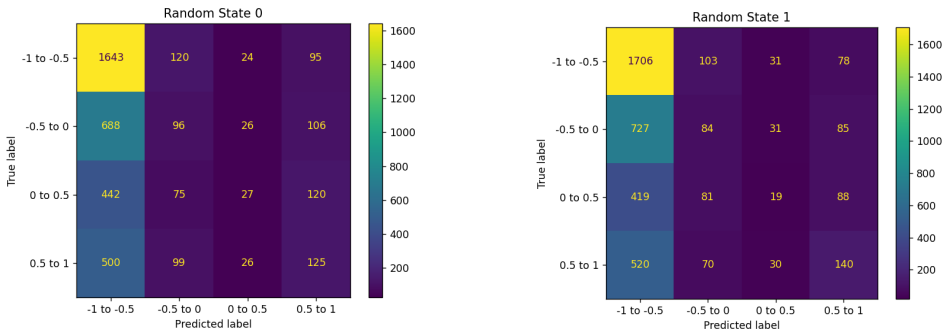


Figure 5: Confusion Matrices

[Sarker, 2021, McDonald, 2009], it slightly modifies the cost function by introducing L2 penalty regularisation (shown below) [Sarker, 2021, Cortes et al., 2012], resulting in more robust models.

$$\text{L2 Penalty} = \lambda \sum_{j=1}^{p} \beta_j^2 \tag{12}$$

where:

β_j represents the coefficient associated with the (j)-th feature.
λ (lambda) is the regularisation parameter, it controls the trade-off between fitting the data well and minimising the magnitude of the coefficients.

The goal is to find the set of coefficients β_j that best predict the output y_i from the input features x_i, the residual sum of squares (RSS), while also keeping these coefficients as small as possible [McDonald, 2009].

$$\text{RSS} = \sum_{i=1}^{n} (y_i - x_i)^2 \tag{13}$$

where:

y_i = observed output (dependent variable) for the (i)-th data point.
x_i = independent variable for the (i)-th data point.

Finally, the objective function of the Ridge regression model can be expressed as the addition of equations (12) and (13):

$$\text{RSS} + \text{L2 Penalty} \tag{14}$$

Grid Search Grid search can also be implemented in this model, as the optimal value of λ (depicted as alpha in the code below) and the solver have to be found.

```
# Define the hyperparameter grid for tuning
param_grid = {'alpha': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
              'solver': ['lsqr', 'sparse_cg', 'sag'] }

# Perform grid search using cross-validation
ridge_grid_search = GridSearchCV(ridge_model, param_grid, scoring='neg_mean_squared_error
    ', cv=5)
ridge_grid_search.fit(X_train, y_train)

# Get the best parameters
best_params = ridge_grid_search.best_params_
print(f'Best Parameters: {best_params}')
```

The best combination of parameters was the following:

$$\text{Best Parameters: } \{\text{'alpha': 3, 'solver': 'sparse\_cg'}\} \tag{15}$$

Results The model utilises a TF-IDF vectorizer to transform the text into numerical representations. The validation set is used after the training to calculate two metrics, mean squared error (equation 2) and the coefficient of determination (R^2), the resulting values for MSE were **0.35** (± 0.005) and for R^2 were **0.148** (± 0.001). The results above will be discussed further in section 3.4, together with the results of all the other algorithms.

3.4 Result Discussion and Critical Evaluation

This project uses a total of nine different machine-learning models, six neural networks and three regression models, to demonstrate the impact that the content of scientific papers has on the nature of future citations. The empirical findings of this research are shown in the table below:

Machine Learning Model	Accuracy	MSE	R^2
Feedforward Neural Network	4.2% (± 0.1)	0.42 (± 0.02)	0.01 (± 0.002)
1D Convolutional Neural Network	3.65% (± 0.2)	0.39 (± 0.05)	0.08 (± 0.01)
Simple Recurrent Neural Network	4% (± 0.2)	0.63 (± 0.02)	-0.57 (± 0.03)
Long Short-Term Memory	4.3% (± 0.05)	0.68 (± 0.02)	-0.66 (± 0.04)
Gated Recurrent Unit	4.1% (± 0.1)	0.67 (± 0.05)	-0.64 (± 0.05)
Graph Neural Network	Training could not be completed.		
Linear Regression	Not calculated.	0.38 (± 0.01)	0.08 (± 0.01)
Logistic Regression	45.5% (± 0.5)	2.12 (± 0.03)	-0.62 (± 0.03)
Ridge Regression	Not calculated.	0.35 (± 0.005)	0.148 (± 0.001)

3.4.1 Accuracy

Firstly, it is important to note that accuracy is not the best metric to determine the performance of a model on regression tasks (the Sklearn library does not allow for the calculation of accuracy in regression models, such as Linear and Ridge) [Wheeler and Calder, 2007, Davis and Goadrich, 2006], as it calculates the proportion of correctly classified instances out of the total using the formula below [Wheeler and Calder, 2007]:

$$\text{Accuracy} = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{16}$$

where:

TP = true positives

TN = true negatives

FP = false positives

FN = false negatives

The main reason it was calculated however is to compare the performance of Logistic Regression (given it essentially turns into a classification task) to that of the neural networks. Since Logistic Regression tries to classify the CD index into four broad categories, its accuracy is going to be greater than that of the other models, on average eleven times greater.

3.4.2 MSE

However, accuracy does not necessarily outline a better model, as observed from the MSE values (the closer to 0 they are, the better the predictive capabilities), Logistic Regression presents a value that is on average over four times higher (2.12) than other models.

Many models' MSE values stabilised around a certain value after some point, barely improving until the end of training (FFN, and 1D CNN), while others got worse (RNN, LSTM and GRU). These trends can be a sign of over-fitting [Lever et al., 2016], but can also be present in cases where the pre-processing steps are insufficient, leading to an overcomplicated model.

3.4.3 Coefficient of determination

It is important to note that the value of R^2 can be negative if the model fits the data points worse than a straight line (this is the case for RNN, LSTM, GRU and Logistic Regression). It should be highlighted that the closer the value is to 1, the closer the predicted values are to the original values.

The greatest R^2 value of a neural network training was 0.08, while for a regression model, it was approximately 0.15, this is in line with existing theory, as regression models should perform better in regression tasks [Sarker, 2021].

3.4.4 Limitations

The purpose of this section is to thoroughly address the shortcomings and restrictions in the chosen models to mitigate them in subsequent work.

While results illustrate that the Ridge and 1D CNN models perform slightly better than predicting the average CD index every time, the results are nowhere near satisfactory.

It is important to highlight that during the training of neural networks, the values of accuracy and MSE on the training set improved significantly, giving on average an accuracy of 10% and an MSE value of 0.04, this leads to the potential conclusion that the training caused overfitting, leading to poor prediction capabilities in unseen data. However, even with the use of hyperparameter tuning techniques such as grid search, the values achieved are not optimal. Moreover, in the implementation of different regularisation techniques such as L2 (Logistic and Ridge) and dropout layers (FNN, 1D CNN, and GNN), the difference in values was great when compared to the rest (RNN, LSTM and GRU), but the values of R^2 still point to underperforming models.

After these observations, the next step was to analyse whether the integrity of the raw text data played a role in predicting the CD index, this was done by using the abstract as an input for the model (after going through the same pre-processing steps), instead of using the entire scientific paper's content. This way, a different input of lower complexity that points to the same CD index was used to train the model (TF-IDF vectorizers, vocabularies and density vectors were adjusted to match the size of the inputs). The resulting accuracy and MSE values of these trainings were similar to the results obtained when using the entire text. This showcases two possibilities:

- The complexity of the model is too low to predict the CD index based on text, leading to a high number of inaccurate predictions (addressed in section 5).
- The content of the scientific paper has a minimal impact on its CD index, meaning it does not affect the nature of future citations [Haslam et al., 2008, Robson and Mousquès, 2016].

Furthermore, instead of focusing on the textual content as an input for the models, it is viable to focus on the structure of the citation network instead, this was to be explored with the implementation of a Graph neural network. Unfortunately, under the time constraints of this project, another dataset could not be constructed in time, even though the coding implementation was completed and tested on a small scale (10 random paper IDs that had "fictional" citations pointing to each other) to ensure it worked (further addressed in section 5).

Moreover, under the time restriction, grid search was not run on the neural network models, potential hyperparameters to be tuned are the dropout rate, the number of hidden layers, the number of units per hidden layer, the batch size, initial learning rate for the Adam

optimiser, and the type of activation functions, they were chosen solely based on existing literature. These could lead to better performances from the neural network models.

Additionally, from the confusion matrices displayed in Figure 4, it is clear that the Logistic Regression model mostly predicts values between -1 and -0.5, this could be due to a class imbalance, as the number of values in the interval [-1,0) are nearly double the number of values in the interval [0,1). This class imbalance can possibly affect the performance of the other machine-learning models.

Finally, a different dataset could have been used to train all the models, to test whether the nodes and edges of the dataset itself could be affecting the performance of the models and potentially leading to better results.

4 Conclusion

4.1 Summary

The project explores the performance of different predictive machine-learning models on a medium-sized citation network of scientific papers. Nine different models are constructed, with 1D Convolutional Neural Network and Ridge Regression having the best overall performances based on three metrics, accuracy, mean squared error, and coefficient of determination. However, empirical results point to a possible class imbalance in the dataset, leading to below-average performances for all the models.

Due to time constraints, the structure of the citation network could not be successfully explored as a potential feature used for predictions, but the use of CD indices proved to be a useful metric to be predicted, based on the R^2 value achieved by the Ridge Regression model (approximately 0.15), as it did show potential to demonstrate whether a scientific paper is destabilising or not.

4.2 Lessons Learnt

Below are a few lessons learnt throughout the project:

1. Some machine-learning models, namely simple RNNs, LSTMs and GRUs that are based on the same mathematical concepts have very similar results.
2. The choice of dataset has a massive impact on the model training, and therefore, on the empirical results.
3. Regularisation techniques such as L1, L2 and dropout layers are extremely important to prevent overfitting.
4. Researching the nature of citation networks played a crucial role in the code implementation of some models, such as the Graphic neural network model.

5 Future Work

5.1 Graph Neural Network

Even though the code implementation was completed, the dataset was not optimal for the training of the model, therefore, using Python libraries such as Scrappy to construct a new citation network and use that as an input for the model would be the next step to analyse whether the structure of a citation network can be a potential feature used for predictions.

5.2 Alternative dataset

Utilising another dataset altogether would be a viable option to better test the models developed, as the class imbalance of CD indexes present in the current citation network had a significant effect on the performance of the models.

5.3 Implementation of grid search

Under the time constraint of this project, grid search was not a viable option for the training of the neural network models, however, hyperparameter tuning by using grid search algorithms would be optimal for future endeavours.

5.4 Different Feature Selection

This project used mainly textual data as an input feature for the models, future iterations might work better by including different features, such as temporal data, and authorship, among others.

6 Project Management

Given certain setbacks such as health issues and parallel projects (analysed in Appendix A), effective time management and allocation of resources were pivotal for the completion of the project. This section discusses skills, resources, time, as well as approaches required to complete the project.

6.1 Skills Audit

On the theoretical side, a basis for the knowledge required was established in semester one, however, with a complete change of topic in late October, more research had to be conducted throughout semester two, mainly on machine learning models and techniques, as well as Scientometrics.

On the technical side, with a basis in machine learning from previous modules in semester one, familiarity with certain libraries such as Numpy, Pandas and Scikit-learn was already established. Semester two was invested in learning mostly about PyTorch, and enhancing knowledge of the aforementioned libraries.

Finally, for soft skills, upon the delivery of the progress report, potential improvements were identified, such as conducting more thorough research on Scientometrics and different machine learning models, before starting any code implementation. Moreover, report writing skills were already on par with the requirements, however, improving research skills also led to better report writing.

6.2 Risk Assessment

Risk assessment for the project is performed before assigning assignments for the semester and calculating their durations. This is done to set up a buffer time to make sure the project is completed by the deadline. Risks and their likelihood are identified, and their impact is evaluated (as outlined in Appendix A).

6.3 Time Management

The Gantt chart in Appendix A (figure 6) displays the originally planned timeline for semester two submitted with the progress report, taking into account the skills audit. However, due to the circumstances discussed in section 6.6, and following Gantt chart (figure 7) displays the actual timeline of tasks completed in semester two.

6.4 Resource Allocation

For the training of the models, the following specs were utilised:

- 12th Gen Intel(R) Core(TM) i7-12650H 2.30 GHz
- NVIDIA GeForce RTX 4070 GPU
- 16.0 GB DDR5 RAM

This resulted in efficient training times even when training more complex models such as LSTMs and GRUs.

6.5 Additional precautions

The report was written using OverLeaf, a cloud-based LaTeX compiler, and Github was used for code backup and version control, this resulted in no loss of information, while also allowing work in different environments, with different machines.

6.6 Reflection

The project management for the second semester had the following takeaways:

- Utilising and testing a vast amount of models ultimately led to sacrificing code quality and validation, leading to a class imbalance in the dataset to be found late into the code implementation. In retrospect, not all models with similar mathematical backgrounds, such as Simple RNNs, LSTMs, and GRUs, had to be included in the research, minor testing could have been done in the beginning to prevent the implementation of all three, leaving more time to focus in other sectors of the project.
- Buffer time is crucial when managing risks, the health of a family member being at risk delayed the start of the project until the third week of March, which affected significantly the outcome of the project.
- Understanding the theory behind the project is completely different from the code implementation, many more hours than originally planned were spent on learning code implementations for different algorithms and models such as Grid Search and Graph Neural Networks.

References

[OED, 2023] (2023). s.v. "scientometrics (n.)". *Oxford English Dictionary*.

[pyt, 2024] (2024). Pyg documentation. Available online: `https://pytorch-geometric.readthedocs.io/en/latest/`. Accessed on April 3rd, 2023.

[Bebis and Georgiopoulos, 1994] Bebis, G. and Georgiopoulos, M. (1994). Feed-forward neural networks. *IEEE Potentials*, 13(4):27–31.

[Berger et al., 2002] Berger, A., Caruana, R., Cohn, D., Freitag, D., and Mittal, V. (2002). Bridging the lexical chasm: Statistical approaches to answer-finding. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*.

[Borghesani and Pedregosa, 2023] Borghesani, V. and Pedregosa, F. (2023). Journal of machine learning research. Available online: `https://www.jmlr.org/stats.html`. Accessed on December 3rd, 2023.

[Bornmann and Leydesdorff, 2014] Bornmann, L. and Leydesdorff, L. (2014). Scientometrics in a changing research landscape: Bibliometrics has become an integral part of research quality evaluation and has been changing the practice of research. *EMBO reports*, 15(12):1228–1232.

[Bornmann and Tekles, 2019] Bornmann, L. and Tekles, A. (2019). Disruptive papers published in scientometrics. *Scientometrics*, 120:331–336.

[Chenxin et al., 2021] Chenxin, A., Ming, Z., Yiran, C., Danqing, W., Xipeng, Q., and Xuanjing, H. (2021). Enhancing scientific papers summarization with citation graph. Accessed on March 26th, 2024.

[Cortes et al., 2012] Cortes, C., Mohri, M., and Rostamizadeh, A. (2012). L2 regularization for learning kernels. *CoRR*, abs/1205.2653.

[CRONIN, 1984] CRONIN, B. (1984). The citation process. *The role and significance of citations in scientific communication*, 103.

[Das et al., 2004] Das, K., Jiang, J., and Rao, J. N. K. (2004). Mean squared error of empirical predictor. *The Annals of Statistics*, 32(2):818 – 840.

[Davis and Goadrich, 2006] Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

[Dong et al., 2016] Dong, Y., Johnson, R. A., and Chawla, N. V. (2016). Can scientific impact be predicted? *IEEE Transactions on Big Data*, 2(1):18–30.

[Funk and Owen-Smith, 2016] Funk, R. J. and Owen-Smith, J. (2016). A dynamic network measure of technological change. *Management Science*, 63(3):791–817.

[Garfield, 1955] Garfield, E. (1955). Citation indexes for science. *Science*, 122(3159):108–111.

[Ghojogh and Ghodsi, 2023] Ghojogh, B. and Ghodsi, A. (2023). Recurrent neural networks and long short-term memory networks: Tutorial and survey.

[Géron, 2017] Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media.

[Haslam et al., 2008] Haslam, N., Ban, L., Kaufmann, L., Loughnan, S., Peters, K., Whelan, J., and Wilson, S. (2008). What makes an article influential? predicting impact in social and personality psychology. *Scientometrics*, 76(1):169 – 185.

[Hirsch, 2005] Hirsch, J. E. (2005). An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102(26):16569–16572.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

[Kiranyaz et al., 2021] Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., and Inman, D. J. (2021). 1d convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, 151:107398.

[Lerman, 1980] Lerman, P. (1980). Fitting segmented regression models by grid search. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 29(1):77–84.

[Lever et al., 2016] Lever, J., Krzywinski, M., and Altman, N. (2016). Points of significance: Model selection and overfitting. *Nature Methods*, 13(9):703.

[Martin, 2019] Martin, A. I. (2019). Machine learning in scientometrics. Available online: `https://oa.upm.es/36488/`. Accessed on November 20th, 2023.

[McDonald, 2009] McDonald, G. C. (2009). Ridge regression. *WIREs Computational Statistics*, 1(1):93–100.

[Montgomery et al., 2021] Montgomery, D. C., Peck, E. A., and Vining, G. G. (2021). *Introduction to linear regression analysis*. John Wiley & Sons.

[Naeini et al., 2015] Naeini, M. P., Cooper, G., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

[NAGELKERKE, 1991] NAGELKERKE, N. J. D. (1991). A note on a general definition of the coefficient of determination. *Biometrika*, 78(3):691–692.

[Park et al., 2023] Park, M., Leahey, E., and Funk, R. J. (2023). Papers and patents are becoming less disruptive over time. *Nature*, 613(7942):138–144.

[Pobiedina and Ichise, 2014] Pobiedina, N. and Ichise, R. (2014). Predicting citation counts for academic literature using graph pattern mining. In Ali, M., Pan, J.-S., Chen, S.-M., and Horng, M.-F., editors, *Modern Advances in Applied Intelligence*, pages 109–119, Cham. Springer International Publishing.

[Ponomarev et al., 2014] Ponomarev, I. V., Williams, D. E., Hackett, C. J., Schnell, J. D., and Haak, L. L. (2014). Predicting highly cited papers: A method for early detection of candidate breakthroughs. *Technological Forecasting and Social Change*, 81:49–55.

[Ramos, 2003] Ramos, J. E. (2003). Using tf-idf to determine word relevance in document queries.

[Robson and Mousquès, 2016] Robson, B. J. and Mousquès, A. (2016). Can we predict citation counts of environmental modelling papers? fourteen bibliographic and categorical variables predict less than 30 *Environmental Modelling Software*, 75:94–104.

[Sarker, 2021] Sarker, I. (2021). Machine learning: Algorithms, real-world applications and research directions. *Springer*, 2(160).

[Schmidt-Hieber, 2020] Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics*, 48(4):1875 – 1897.

[scikit-learn developers, 2024] scikit-learn developers (2007 - 2024). Sklearn. Available online: `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#:~:text=The%20'newton%2Dcg'%2C,only%20for%20the%20L2%20penalty.` Accessed on April 7th, 2024.

[Sherstinsky, 2020] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Ruslan, S. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

[Tolias et al., 2016] Tolias, G., Sicre, R., and Jégou, H. (2016). Particular object retrieval with integral max-pooling of cnn activations.

[Wheeler and Calder, 2007] Wheeler, D. C. and Calder, C. A. (2007). An assessment of coefficient accuracy in linear regression models with spatially varying coefficients. *Journal of Geographical Systems*, 9:145–166.

[Wilbur and Sirotkin, 1992] Wilbur, W. J. and Sirotkin, K. (1992). The automatic identification of stop words. *Journal of Information Science*, 18(1):45–55.

[Wu et al., 2019] Wu, L., Wang, D., and Evans, J. A. (2019). Large teams develop and small teams disrupt science and technology. *Nature*, 566(7744):378–382.

[Yousef et al., 2022] Yousef, A. A., Mahmoud, S. F., Tamer, M., Yassin, A., Fida, A., and Abdulrahman, H. (2022). A machine learning model to predict citation counts of scientific papers in otology field. *BioMed Research International*.

[Zhou et al., 2020] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

[Zhou, 2021] Zhou, Z. (2021). *Machine Learning*. Springer.

Appendices

A Project Management

Risk/Event	**Probability Score**	**Impact**	**Impact Score**	**Strategy/Prevention**	**Further Actions**
Supervisor Unavailability	1	Potential delay in more complex sectors	3	Research in library or online	Work on other tasks in the meantime
Theory too complicated	2	Potential delay in report writing or understanding	3	Discuss concepts with peers, dedicate more time	Contact supervisor
Lack of Coding experience	4	Potential delay in code	4	Dedicate more time to coding and learning about machine learning	Find alternatives that are familiar to the author
File corruption	1	Potential delay in all technical aspects	5	Use of cloud-based software and backups	Depending on the moment of the loss, changing goals might be required
Citation network not good enough	2	Hindering of the machine learning models' capabilities	4	Gather enough data to ensure model training will be smooth	Leave enough time to find backup datasets
Other academic projects	5	Freeze of work	4	Excel in time management	Revise the projects' goals
Illness	4	Delay caused by unavailability	5	Focusing on tasks as soon as possible to prevent last-minute changes	When impacted significantly, contact special considerations

Table 2: The second semester's risk assessment table

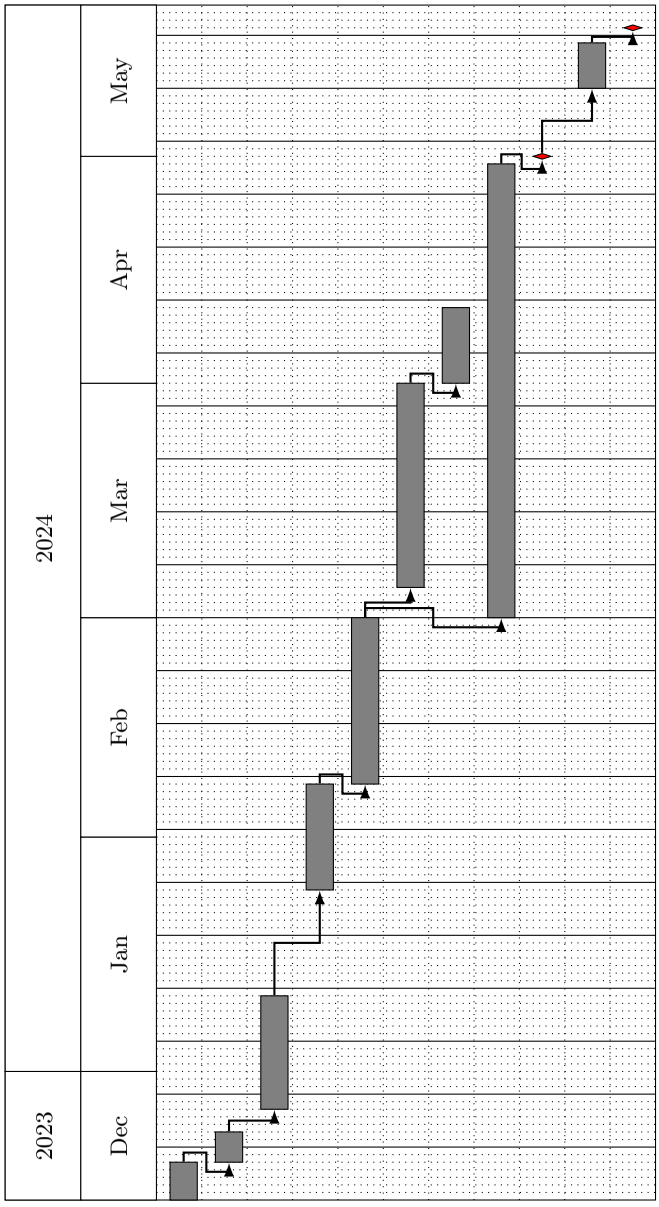


Figure 6: This is the originally planned Gantt chart for the final report submission, spanning from December 15th, 2023 to May 17th, 2024.

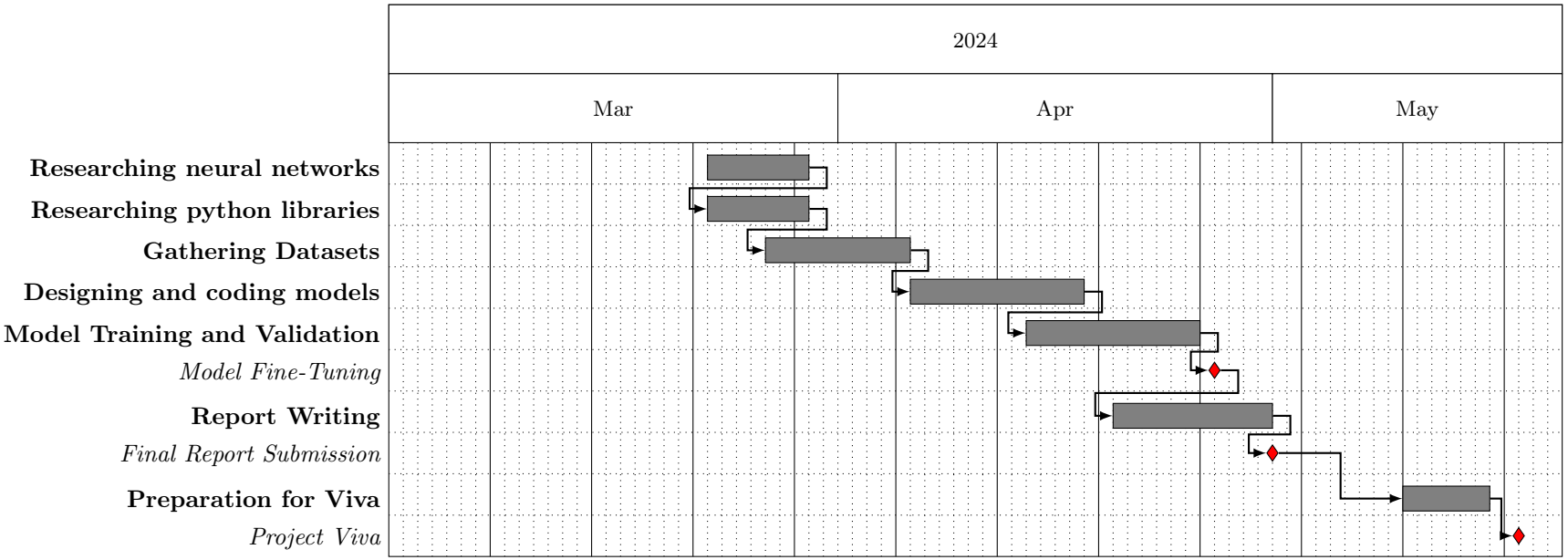


Figure 7: This is the finalised Gantt chart for the final report submission, spanning from March 1st, 2024 to May 20th, 2024.

B Project Brief

Since there was a complete change of topic in late October, the project brief is completely redundant, for this purpose, it has been omitted.

C Source Code and Data

The project is run in an environment with Python 3.11.5 where the NumPy, PyTorch, Sklearn, pandas, matplotlib, plotly, keras, nltk, and tensorflow libraries are installed.

The final code implementation has the following directory structure: