Domain analysis of feature implementations between Classic and Deep NLP models.

William Green

School of Computing

Final Year Project PJE / PJS 40

July 19, 2021

Abstract

This study investigates the relationship between theory and implementation amalgamations within Natural Language Processing models, researching how the use of older techniques can be brought to life with modern fundamentals for feature extraction of textual classification methods on Student Feedback; this project researched several NLP methods such as Bag-Of-Words, POS-tagging and Word2Vec categoried as Classical, Modern, and Contemporary approaches. To demonstrate the project hypothesis, two indented models were created to display performance differences based on open-source datasets.

This project concluded . . . This project faced one major limitation, being dataset sample size of student feedback, however, this did not affect the results of the model itself.

Keywords— natural language processing, machine-learning

Table of Contents

List of Tables

List of Figures

Acknowledgements

This ones for me.

Chapter 1

Introduction

This report focuses on the theoretical differences of how Natural Language Processing (NLP) models are implemented and subsequently how performance is affected with certain technical abilities, a key aspect of this report will demonstrate the question "does adding new to old bring enhancements?". This chapter will cover the technical context of this project and report, the aims, objectives and introduce the domain in which this project lies. Natural language processing is subtopic topic covering and interconnecting computational theory, artificial intelligence/ machine learning and linguistics.

To consolidate the theoretical findings throughout technical chapters, this report will include two variants of a traditional NLP model that represents how a specific NLP technique is implemented. The aims of this project are to produce two machine learning models in which outline if and how traditional NLP techniques can be enhanced using modern theoretically driven techniques; the fundamental ideology of this project is to explore amalgamating NLP concepts and techniques to seek performance increases for dataset dependant models, the specific domain for this project is NLP in academia with the dataset being focused on Student Feedback Surveys. As seen in Chapter 2, we can expand our specific intention for this project and dataset to produce a novel NLP model on how to predict Student Feedback using the same techniques.

1.1 Background and Context

Natural Language Processing has been relatively overlooked during the boom of machine learning, its fundamentals have not changed as there has not been a reason to progress at the same rate as other areas in "Artificial Intelligence", by adapting well—known theory, it is possible to progress the computational abilities of NLP. The background of this project is to bring modern approaches to older implementations with the justifications being student feedback surveys as the domain.

This project will be looking at areas from: Text and Speech Processing, Morphological Analysis, Syntactic Analysis and Lexical Semantics to give breakdowns of how they could function together to provide a potentially more efficient model.

1.2 Project Aims and Objectives

To successfully validate my project statement, there are several underlying implications that project aims must establish; the aims of this report are to compare the theory behind classical models and machine learning models of NLP and its subcategories of linguistics, such as grammar and text classification. This report will be corroborated using programming artifacts shown throughout, they will feature two NLP models, one of which demonstrates classical implementations on a given dataset and the other demonstrates an adapted form of a classical implementation with additional ML theory and techniques.

These aims will be achieved by meeting the following objectives:

- Contrasting and comparing types of NLP techniques in a specific domain
- Research and provide results for adding ML techniques to a traditional method
- Compare older methods to modern methods
- Speed advantages or disadvantages when combining different methods and implementations

- Using research from this disco comparing POS tagging and word2vec with ML implementations for text classification and sentiment analysis using the results

By meeting these objectives, this project will have highlighted a novel approach (explored in Section 5) to text classification and minimising computational costs whilst having no detrimental effects to accuracy.

1.3 Deliverable

Project Deliverable: compilable Python object.

1.4 Constraints

The biggest constraint for such a problem—related project is one of time, including time management; the approach of this project is to reflect on current theory to understand how to implement a novel solution to a specific application's domain, as implementations of Text—Classification are heavily theory dependant where each topic has a broad overview, it seems time will be of essence. A secondary constraint to this project may be inherited from a programmatic approach, whereby language features, libraries, or framework versions may have conflict.

1.5 Risk Assessment and Constraint

Risks

Chapter 2

Literature Review

The focus of this chapter is to analyse and breakdown current research and literature concerning the use of NLP models within a domain and their implementations, specifically looking into the area of academia. The domain for this research will be text classification on Student Feedback forms; NLP has many interconnecting domains, in which implementations can heavily affect performance and the returned results. As the theory behind NLP grows, it is vital to use the least computationally cost—effective methods in which this section will be looking at merging newer techniques with older models to potentially improve our understanding of NLP models.

2.1 What is NLP?

Fundamentally, Natural Language Processing is an area of Computer Science which enables computer systems to access and understand human linguistics (Eisenstein, 2019), expanding, NLP is theoretically driven computation for the purpose of evaluating, interpreting, and depicting naturally occurring transcripts to a certain level of detail. Differing depths of linguistics are used for analysis in—order to return a desired human—like range of processing for a particular application (needs the cite).

Over the last 20 years, NLP has become an integral topic of Computer Science as it combines computational linguistics with a popular buzzword Artificial Intelligence or more accurately Machine Learning (Ongsulee, 2017), these terms have been generalized as this area of computing heavily relies on theory from different departments. Human and machine communication mediums both have similarities, to which we can model our understandings on, syntax is an intrinsic value to both communications, and it is used to label every component of a language and its sets of rules (Jain et al., 2018).

The value of NLP is in its ability to remove ambiguity in linguistic forms, this explicitness results in a clear data—driven numeric structure for numerous types of applications (sas.com, 2021). The returned data structures are a result of some form of input, NLP algorithms can handle speech, text, or images where of the appropriate architecture. The properties and potentials of NLP are now used for commercial spaces and for public interest, several areas including:

- Machine Translation
- Speech Functions
- Dialog Interfaces
- Text Analysis
- Natural Language Generation
- Writing Assistance

The list above outlines six key areas of NLP that are used in commercial spaces and the majority appeals to the academic space; the listed six areas are a high—level overview at how different theoretical approaches combine in—order to perform a given task (Dale, 2019) for the purpose of this literature review, we will be looking at the most relevant areas in which aid this domain.

2.2 Uses of NLP in Academic Institutions

As NLP expands, so do its domains; a more recent use of NLP is within academic institutions. During the last term of a module, it is common for universities to collect data regarding its practices and teaching etiquette. Universities will utilize both formal and informal techniques for elicitation, typically a printed hand—copy or via an online questionnaire, thereafter student feedback is analysed to provide institutions with a gauge on how to improve students' satisfaction, module content, structure, and teaching methods. Information elicitation is completed in the form of a survey, with two main question approaches, being "program—wide" and "module—specific" to target flexibility of opinion vs factual coverage (Beran et al., 2007; Keane & Labhrainn, 2005).

During the academic year of 2019/2020, education became distanced due to COVID—19 and accordingly E—learning soared as academic institutes were forced to adapt their teaching mediums and operations to become temporality online only (Burgess & Sievertsen, 2020), this resulted in a frequent uptake of online feedback. This surge of sudden online academia has resulted in rapid development of Massive Open Online Courses (MOOC), these E—learning platforms enable student feedback on an extensive scale with reputable data to develop and train NLP models (Wang et al., 2021).

The data gathered is used to give insight if users are satisfied with their academic consumption, NLP methods can be applied to student feedback to give the academic institution an idea if its students are validating the unified theory of acceptance and use of technology (UTAUT) model (Kayali & Alaaraj, 2020). Large data sets such as a class of 200 students would be tedious and time consuming for a lecturer to manually analyse individual feedback; combining aspects of staff roles and deep learning would utilize the computational power required for sizeable datasets by minimizing the required engineering (LeCun et al., 2015).

Manual thematic analysis of a dataset to formulate codes and themes also allows for human error, poor judgement as interpretation is subjective, and themes may be overlooked (Belotto, 2018). As more aspects of data—driven interactions between clientele can be applied to NLP, the development of an all—purpose, accurate and, secure method to automating the elicitation of necessary linguistic aspects from an input source is increasingly imperative (Sindhu et al., 2019).

A generalised approach for analysing student feedback is with the use of Word Embedding, popular implementations are Word2Vec, GloVe (Pennington et al., 2014), and FastText (Edalati, 2020).

2.3 Applicable Methods for Text Classification on Student Feedback

The intended outcome is to evaluate student feedback surveys which will be targeted towards enhancing the level of learning and engagement by students; to achieve the desired results, the following techniques would be most suitable if appropriately applied in this context.

2.3.1 Sentiment Analysis and Opinion Mining

The desired outcome is to derive subjective material from students' input, this can be insights, discussions or opinions which automatically review the polarity (negative, neutral, or positive) of information regarding the academic facilities (Kandhro et al., 2019). Sentiment Analysis has three levels of scope which can be reduced to emulate different levels of comprehension:

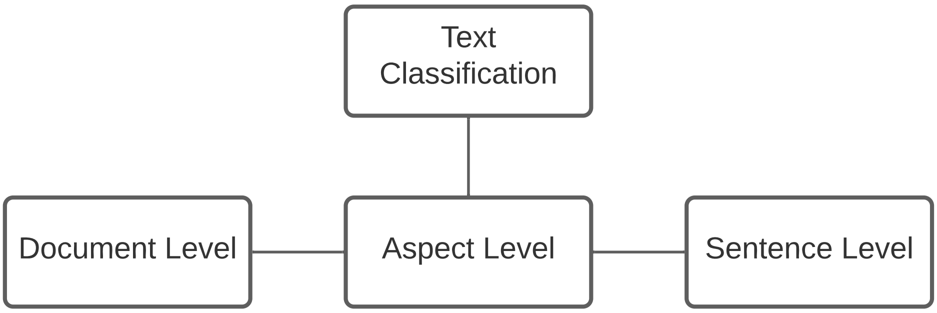


Figure 2.1: Taxonomical overview of Text Classification Levels

Kastrati, Imran, et al. (2020), established an aspect—orientated opinion—mining model; student feedback was in English (Natural Language) whereby three unique NLP techniques were applied to the dataset to produce three represented perceptions of the same dataset. The NLP techniques applied were: Word Embedding, Term Frequency (TF) and Term Frequency—Inverse Document Frequency (TF—IDF).

Kastrati, Imran, et al. (2020), trained their models using already explored classifiers, Decision Tree, Naïve Bayes, Support Vector Machines and Boosted Graphs on a 1—Directional Convolutional Neural Network (1D—CNN). Kastrati, Imran, et al. (2020), found traditional aspects of Machine Learning implementations yielded greater results than that of the sole use of a 1D—CNN.

Kastrati, Arifaj, et al. (2020) "Aspect—Based Opinion Mining of Students' Reviews" (2020) to produce a weakly supervised framework aimed at training deep learning models with little to no human interaction. Their proposed framework analyses a desired sentiment at document level and yielded an overall F1 score of 86.13% accuracy; they also tested their framework at the aspect level for sentiment analysis to which yielded an F1 score of 82.1%. The logic of this paper is applicable to this project as the framework is appropriate as little human involvement will occur when training and saw a similar F1 scoring.

2.3.2 Topic Labelling

This technique is used in conjunction with text mining to automatically handle significant and reoccurring themes and topics within students' feedback with automatic creation of category labels per survey and section. Latent Dirichlet Allocation (LDA) is a suitable model which can achieve the above, it uses Latent Dirichlet Distributions to group topics as a multinomial distribution of words and is able to associate words based on the probability distribution of the set (Unankard & Nadee, 2019).

2.3.3 Language Detection

This technique of NLP deals with determining which natural language (NL) is provided as the input source, this approach could be particularly beneficial to lecturers if the student and lecturer do not share the same first language; a classifier could be challenged by identifying an incorrect NL due to lexical and syntactical resemblance. These challenges could return pedagogical ambiguities within feedback (Heift & Hegelheimer, 2017), if a lecturer misinterprets student feedback due to a misunderstood lexical, an unintended impression will be represented.

This can be aided by converting NL structures in to a first—order logic (FOL) object to which these mathematical models will return the highest matching lexical in a percentage (Perikos et al., 2017). Therefore, minimising miscommunication.

2.3.4 Intent Detection

This technique is used to programmatically classify implied intent within an input, based on a certain ambition or outcome, usually a verbal adjective. Every student feedback survey has the same interaction purpose, to improve their quality of learning, this specific domain can be modelled to automatically categorise each intentional improvement specification. Intent classification can also be used to provide real—time feedback to lecturers opposed to standardised question—answer dialog systems (Jensen et al., 2020).

A common implementation of intent detection is with the use of a rule—based model, these systems use predefined constraints as intents with the hypothesis that occurring utterances conform with the predefined set of rules. These rules will be disputed when a novel utterance is parsed, as students have different methods of expressing their views, novel utterances will be frequent and with the appearance of utterances without a predefined label will increase, this problem exists under Zero—shot learning with CNNs (Xia et al., 2018).

However, textual classification is still an area of development and is not well—understood with regards to the most appropriate implementation, algorithm, and paradigm combination (Thangaraj & Sivakami, 2018). An inherent objective of this paper will be looking at the amalgamations of theory to return the most effective and efficient results driven implementation for domain specific feedback classification.

2.4 Related Research

This section will predominantly concentrate on considerable studies of similar nature, these studies conduct research spotlighting technical performance of differing textual and contextual classification methods using different datasets and NLP models. Analysing previous literature will guide further development as they will give support towards vital features that are a prerequisite for essential system requirements, to the contrary, previous literature may outline features that are not of importance and potentially expose gaps in current research.

2.4.1 Combination Classifiers

The completed work in "a comparative study of classifier combination applied to NLP tasks" by Enriquez et al. (2013), was viewed as a comprehensive comparison and overview of diverging NLP implementations and combining methods for exercised NLP workloads. Enriquez and colleagues' findings suggested lesser explored NLP models and classifiers yielded higher performance opposed to well—known implementations, for example, the combination of "stacking" anchors and "cascading" input layers for Part—of—Speech tagging returned results exceeding expectations (Enriquez et al., 2013).

The fundamental concepts Enriquez et al. (2013) encountered are applicable to modern development of combination classifiers; when developing a novel combination model, there are two compulsory criteria that must be met to be successful: heterogeneity of the chosen classifiers, this ensures a computational mistake will only be met once and will be provided a differing perception to an encountered error; veracity of the chosen classifiers, each classifier must reduce the occurrence of inaccuracies over another selected classifier.

An additional audit should be performed to verify the certainty of the desired classifiers will work together: statistical, are the chosen classifiers best suited? Given the problem based on previous resources; computational, accounting for time and space complexity, is there a potential to reach computational limits? That affect the desired result; representational, has the classifier been previously research to understand the objective task? Considering the criteria above, Enriquez et al. (2013) chose to investigate:

- Voting
- Bayesian Merging
- Behaviour Knowledge Space
- Bagging
- Stacking
- Feature Sub—spacing
- Cascading

The above NLP methods and techniques were applied to 9 different corpuses to train models for Part—of—Speech Tagging.

Since the work of Enriquez et al. (2013), NLP methods have advanced, more up—to—date findings reviewed in the paper "Prediction of Sentiment Analysis on Educational Data based on Deep Learning Approach" by (Sultana et al., 2018) centralises eight classifiers for performance inspection in which they are put against each other for speed, accuracy, and computational cost. This paper includes an open—sourced educational dataset from Kiteboard 360 which is provided to the individual classifiers, the classifiers tested were:

- Support—Vector Machine (SVM)
- Multi—layer Perception (MLP)
- Decision Tree
- K—star (K*)
- Bayes Net

- Simple Logistics
- Multi—Class
- Random Forest

The dataset was parsed to each classifier to create a trained model, the results were then investigated and corroborated with dummy data; if the returned object was valid, it was evaluated by metrics. Scoring against metric such as returned accuracy, RMSE, specificity, sensitivity, F1 percentage and Receiver Operating Characteristics (ROC) curve area to compare performance to conclude the most valuable model for a given dataset (Sultana et al., 2018). According to Sultana and colleagues, SVM and MLP implementations are perceived as the two surpassing models in comparison for applying NLP techniques to student feedback.

2.4.2 Constructing a New Combination Classifier

When attempting to create a novel approach to a widely researched area, challenges will be faced due to the theoretical capacity of current implementations; coverage on current methods must be known to be able include adaptions for a successful improvement. The review work of "Text Classification Algorithms: A Survey" by (Kowsari et al., 2019) provides in—depth insight for the construction and expansion of already implemented algorithms.

Kowsari et al. (2019) summarised the construction of text classification algorithms in real—world applications share four key aspects, in which can be dismantled into "feature extraction", "dimension reductions", classifier selection", and "evaluations"; phase 3 of Kowsari et al. (2019) process is complemented by Enriquez et al. (2013) findings on how to choose an appropriate classifier.

Kowsari et al. (2019) first addressed that the scope of the classifier must be identified, according to the scope levels of Sentiment Analysis in Figure 1. Phase 1 being feature extraction handles the source input, namely unstructured datasets that must be converted to an acceptable object for a classification model, this includes cleaning and formatting of the object.

Phase 2 being dimensionality reduction will handle the acceptable object to check for high costing computational executions, this allows a classification model to make use of low costing functions without decreasing accuracy, it also allows for pre—processing to take place rather than using inexpensive classification models; the aim of phase 2 is reduce the time and space complexity of a classification method.

Raunak et al. (2019) proposed a novel approach on how to handle datasets where dimensionality is a computational issue, Raunak demonstrated the use of pre—trained word embedding models for all depths of a document with emphasis on reduced space complexity. The proposed model uses the combination of the Parasitism—Predation Algorithm with Principal Component Analysis as a post—processing layer to filter irrelevant lexicons.

Phase 3 is simply identifying the most appropriate classification pipeline. Phase 4 is evaluation, there are many ways to evaluate how a classification model performs but the most important are speed and accuracy Kowsari et al. (2019).

2.4.3 Identifying Research Gaps and Including Novelty

The purpose of reviewing existing literature is to understand how current research and findings are being used to expand a topic's theory; the aim of this section is to compile accepted theories, identify common themes from background knowledge, and to distinguish potential gaps in existing literature to provoke the creation of a novel NLP idea and methodological schematic.

This section identifies two suggestions based on concurrent themes in this report; one is suggestive to the theoretical workings of this paper and the other is suggestive to domain specific feature applications.

Many comparative studies overlook how the interaction between simplistic aspects of traditional approaches can be benefitted by incorporating machine learning aspects such as (Convolutional or Recurrent) NNs can affect classification performance. As NLP grows, the comparison between POS—Tagging and evolved versions of Word2Vec have also been overlooked, studies such as "Recent Trends in Deep Learning Based Natural Language" by (Young et al., 2018) directly compare linear implementations of statistical analysis but do not look at non—linear Word2Vec advances such as SPvec (Zhang et al., 2020), this will be expanded on in section 8.1.

Suleiman and Awajan (2019) state that Word2Vec is an efficient model for word embedding, however think it can be improved with an extension, their proposed model explores how POS—Tagging could enhance the probabilistic values of results returned by Word2Vec models as POS—Tagging calculates a higher vector between feature semantics. Whilst covering Word2Vec implementations with POS—Tagging, they did not cover fundamental analysis of how POS—Tagging effects Word2Vec's facets such as POS—Tagging with CBOW vs Skip gram algorithms; item 1's "enhancement" could refer to the use of character n—grams to ensure the importance of Word2Vec word—order. this will be expanded on in section 8.1.

According to Sultana et al., (2018) applied SVMs yielded greater results when combined with a textual classification technique and was the building block for item two; the justified domain being student feedback can be promoted to predicting student feedback based on existing results and Text Frequency Analysis (Alqurashi, 2019).

Alqurashi (2019) proposed a new framework with four key factors to measure student satisfaction; 167 students completed a designed survey targeted at course interaction and perceived learning. Using a 5—point score system, the results indicated that student learning interaction had little effect on the prediction model, with score of 0.1%. Alqurashi's findings are important as it gives insight to which labels should be parsed based on importance to train a new predication model for topic labelling, suggested in section 2.3 which in turn validates the findings of (Unankard & Nadee, 2019) for topic detection.

Chapter 3

Methodology and Project Management

When managing the development of a project, there are several approaches one can take when planning the development lifecycle (Shylesh, 2017); the three main approaches for any project are sequential (linear), incremental, and iterative (non-linear) phases (Akinsola et al., 2020), which methodology is best suited depends on the nature of the application. The nature of this project involved one developer with evolving requirements, adaptable features, and unforeseen management issues, therefore the following methodologies were explored and appropriately chosen.

The planning and development carried throughout this project were based on an agile methodology model designed for specific and individual use. This section discusses how to differentiate between the most appropriate and applicable methodology, to which will establish the final selected software development lifecycle model (SDLC). The scope for this project can be displayed as such:

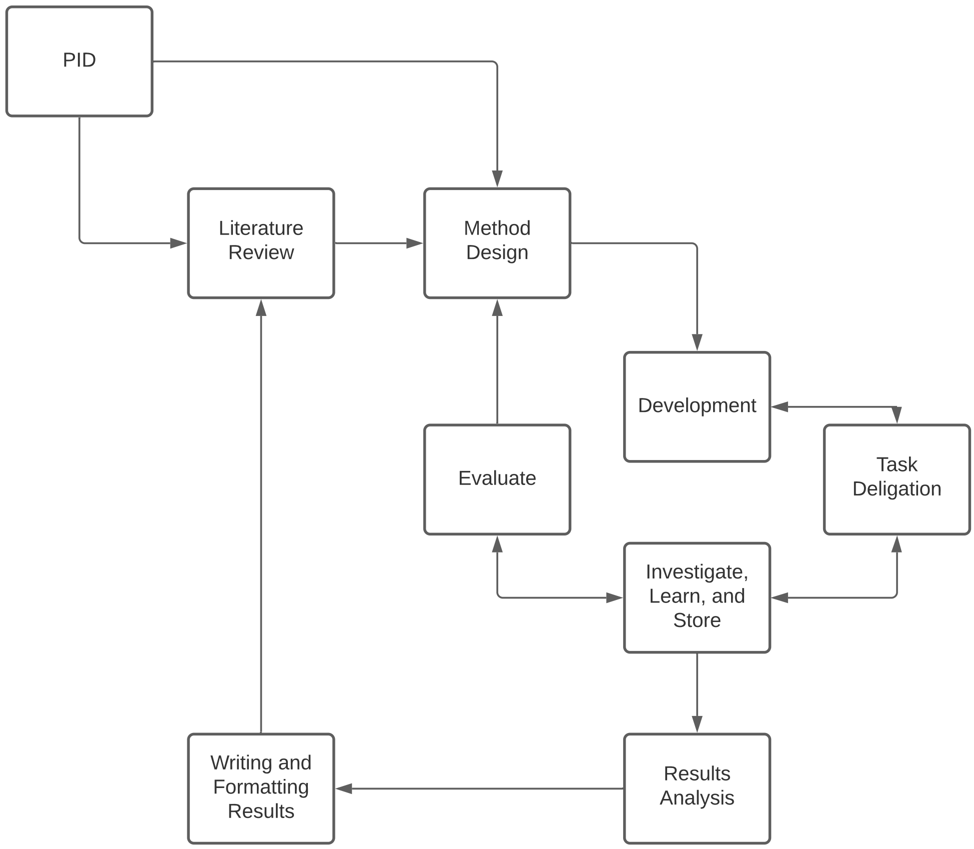


Figure 3.1: Overview of my project from start to finish

As seen in Figure 3.1, this project follows an iterative process with included circular motion for data validation and section corroboration. Due to this nature, it seems a hybrid SDLC Model is best suited for this project; the apparent combination of methodologies allows for aspects from both strict Test-Driven Development and Rapid Application Development without the inclusion of their inherent disadvantages.

Whilst the original commencement plan for this project was entirely constructed on the Waterfall Model as depicted in the supplied Gantt Chart within the PID document, this however, was not a sufficient process mainly due to time constraints, thus the Waterfall Model structure was partly ignored to allow for improved efficiency requirements and minor tweaks to previously stated tasks. These tweaks are outlined in the final chosen (adapted) model.

3.1 Methodologies

This project uses two SDLC's, one for the project in its entirety and one for development.

3.1.1 Data Mining

This project's primary objective is text classification which focuses on predictive aspects of NLP, data mining and analytics are inherently used in machine-learning and natural language processing as for a model to be predictive there must be a history of data to be analysed, this project uses openly sourced student feedback from Kaggle to achieve clean and rich data without breaching ethical concerns.

Big datasets are becoming widely used for research and data-mining techniques aid development as they ensure the correct dataset is being used, appropriate data must be used for the applied techniques and data manipulation because inappropriate data could lead to inaccurate or misleading results. It is essential that the use of data is appropriate for the proposed machine learning model, in the scope of this project it is student feedback being mined and analysed through a predictive model.

3.1.2 Data Analytics

Data analytics is a necessary topic for NLP to achieve the desired outcome; within this project, the outline goal is to be able identify lexical trends by analysis a given dataset to answer predict questions and potentially speculate a topical conclusion, i.e., a certain student is content in their feedback. This data-driven decisions and outcomes are only possible from analysing existing datasets and their inherent meaning(s); data analytics is a broad area within machine learning and this project concentrates on descriptive analysis and predictive analysis. The data analytics being performed on the chosen datasets are predominately for pattern recognition and accuracy improvements.

3.1.3 Developmental Flowchart

The elected methodology for this project is a bespoke hybrid model that includes features from sections 3.2.1, 3.2.2, and 3.2.3 to reap the benefits from each stated methodology whilst minimising the drawbacks. The proposed model is a specific draft construction with the focus on machine learning projects, it is appropriately titled Machine Learning Development Lifecycle and is its own SDLC. The proposed methodology can be displayed as a flowchart:

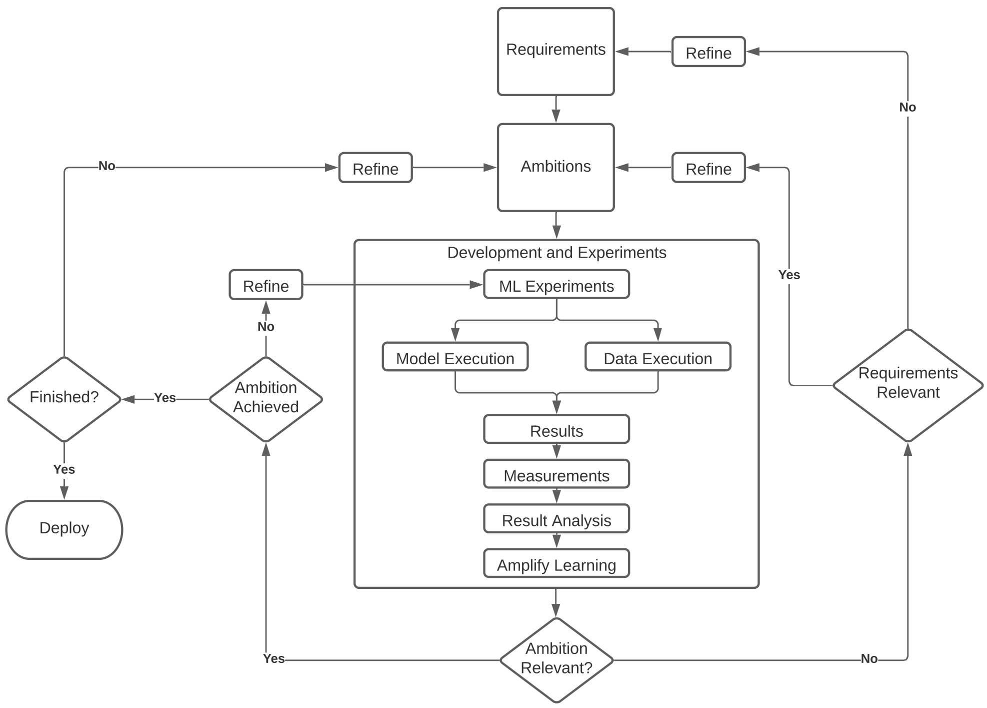


Figure 3.2: Workflow for the development of my project.

The proposed methodology includes an Agile and Iterative core, the project requirements are directly transposed into the ML ambitions, the ML ambitions are obtained with the guidance of its experiments. This approach allows for deferred commitment with scope and requirements, code quality and coverage whilst achieving a quick artifact delivery (Assaf, 2021).

3.1.4 Elected SDLC

This project strictly follows the Cross Industry Process for Data Mining model, widely known as CRISP-DM; when executing the development of a machine learning based project, there are several prefacing steps such as planning, organisation, and implementation, currently there is no standard model to efficiently carry out the development of such a project and this is where the CRISP—DM model is useful. It aims to intersect personal skills and knowledge into an effective and effective process (Wirth & Hipp, 2000). The proposed workflow can be displayed within a model such that:

Figure 3.3: The CRISP-DM Software Development Lifecycle (Jenson, 2012).

This methodology can be described as a hierarchical process model as there is an apparent level of abstraction, being described as: phases, generic tasks, specialised tasks, and process instances (Wirth & Hipp, 2000). This process is beneficial as it can be applied to this project, each section is broken down into its respected hierarchy and labelled appropriately which is then handled in development, it allows for stable agile development as the developer can refine requirements or datasets as the project goes on, however acknowledging potential unforeseen aspects.

The process model can be deconstructed into six specific categories or phases (Figure 3.3):

- ***Business Understanding***: rather than business understanding, the context for this project is the project scope itself, the developer needs to understand the context of the problem.
- ***Data Understanding***: understanding the initial data collected to identify data quality and detect potential insights.
- ***Data Preparation***: once the initial data is collected and analysed, it will need to be prepared to construct a finalised usable dataset for the model to be parsed.
- ***Modelling***: this phase deals with how the data is parsed into the model after applying the intended ML techniques, this is very similar to preparation phase.
- ***Evaluation***: once the desired models have been constructed and the data has been parsed, it will need a quality check to ensure analysis is correct before deployment.
- ***Deployment***: the requirements have been met by the developer and the model yield appropriate results, the model can be deployed to an appropriate environment.

As data mining is not a standard domain which can produce varied results depending on how the project is structured and outlined, the need for a standard framework was apparent for this project. There is not reject reasoning behind this selected SDLC as the approach aims to improve accuracy, efficiency, and effectiveness of data mining applications.

3.2 Project Management

3.2.1 Development Management

The source code to this project was decided to be monitored via a GIT repository stored on GitHub; the use of version control for a machine-learning based project is especially helpful as you can backtrack certain functions that outperform changes without affecting the entire stack. The choice to use GIT opposed to a local project had several factors, such as:

- Version control
- Track bugs
- Back-Up complete
- Branches for different features
- Testing purposes
- Source Code sharing
  - Developer to supervisor
  - Developer to developer systems

GitHub was the chosen hosting platform due to industry standard and familiarity to both the developer and project supervisor, however, other GIT based hosting platforms such as GitLab or BitBucket do exist that satisfy the same project requirements.

3.2.2 Task Management

Delegation of project tasks evolved over the course of completion; mentally keeping track of project TODOs became mentally taxing, thus a formal system for task delegation was implemented. As this project was completed by a sole developer, that immediately ruled out the use of a SCRUM based project board as team roles were not necessary, therefore, the "easy-to-adopt" KANBAN method was implemented, which by itself is also justification for a LEAN based SDLC model. KANBAN is a solution in which eases aspects of project design, management, improvement flow and situational knowledge and awareness by visualising (a simplified) "TODO", "DOING", and "DONE" categories. This in turn balances work demands compared to work capacity.

This project uses two KANBAN applications, separated by general delegation and development delegation, to oversee metrics such as developer velocity, lead and time cycle, and actionable agile metrics, a Trello Board was implemented that included all tasks related to this project and report. In addition, a GitKraken Board was used for development related tasks such as feature ideas and bugs, these two boards were synced for a complete overview of tasks on Trello to capture the project's work-in-progress (WIP) limits.

3.2.3 Time Management

Within the Project-initiation Document (PID) in Appendix A, a Gantt Chart is supplied detailing the expected timeline this project will follow. However, it was subject to change and adaptions due to unforeseen circumstances, the below is a Gantt Chart detailing an accurate time-frame which was created nearer project end:

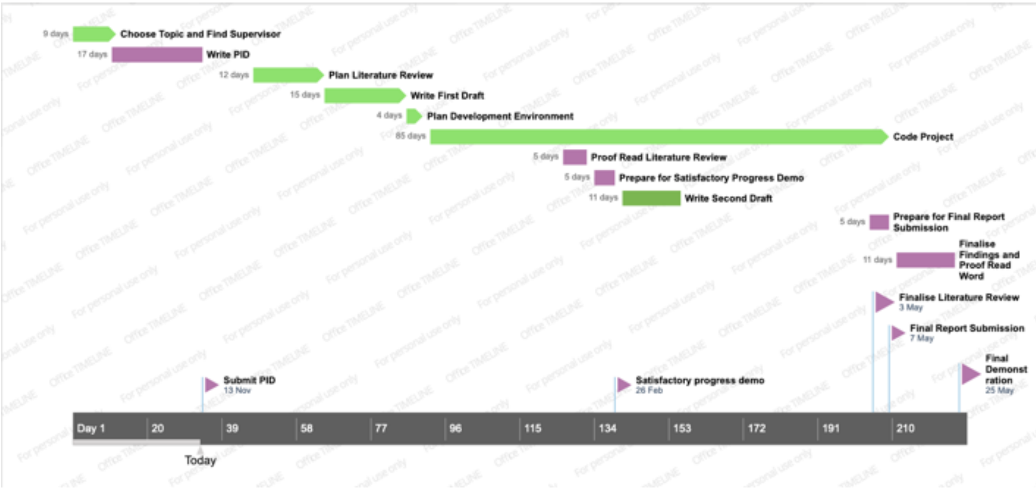


Figure 3.4: Project Gantt Chart.

Chapter 4

Requirements and Planning

This chapter outlines the desired requirements which are necessary for this project to work and to have met the criteria proposed, it also includes additional requirements such as potential requirements and unnecessary but topical requirements. In section 3.1.3, the model flowchart displays requirements must be detailed as part of the SDLC for project ambitions and solution scope; all requirements listed are relevant to a machine learning NLP focused model, seen as a process orientated requirement.

4.1 Requirement Elicitation

Requirement elicitation for this project started by defining its scope, planning a potential solution and desired outcome to further have a discussion with the project supervisor, to gain requirement insight. The project's functionality and capabilities were outlined in-order for the functional and non-functional requirements to be identified.

As no human parties were involved, other than the developer and project supervisor, no external input was involved. Therefore, the requirement elicitation process primarily focused on academic research and existing applications of similar intent. Existing text classification machine learning models provided insight to potential requirements; the research covered in the literature review (section

2) under existing applications highlighted key areas of interest. This project benefitted from deriving requirements seen from a process perspective opposed to traditional methods such as a questionnaire, for example a procedural step within the machine-learning process i.e., model must clean data.

4.1.1 Requirement Research of Current Models

This project's system and functional requirements can be broken down into a simplified structure, such that the procoess for text classification if as follows:

1. Parse Data
2. Pre-processing of Data
   (a) Tokenization
   (b) Vectorization
3. Text pre-processing
4. Clean data
   (a) Remove empty data, useless punctuation, and unnecessary stop words
   (b) Stem the words
5. Feature Engineering and Extraction
6. Feed clean dataset into model
7. Train model
8. Tune hyper-parameters
   (a) Number of layers in model and units per layer
   (b) Dropout rate
   (c) Learning rate
   (d) Kernel size
   (e) Embedding
9. Evaluate model

4.2 Requirement Specification

This project makes use of the MoSCoW prioritisation technique whereby requirements are defined as "Must Have", "Should Have", "Could have", and "Won't Have at this Time". The four categories of MoSCoW can be translated to Core, Base, Additional, and Future Work. This project acknowledges the disadvantages of the MoSCoW technique; however, its simplicity outweighs the disadvantages in this environment.

4.3 Requirement Constraints

- Issues locating appropriate openly sourced datasets of existing student feedback.
- Accuracy of model given located datasets.
- Components of the model may not communicate well with other aspects.
- Programming and language concern.
- Logistical issues.

Defining potential constraints of this project aided identifying its functional and non-functional requirements as there was an insight as to what may work or not work when in the development phase, these requirements are listed below with the appropriate level of prioritisation.

4.4 Functional Requirements

- ***Must Have***
  - The model must correctly parse a given dataset such that the correctness of the original dataset is intact.
  - The model must pre-process the dataset into a given method: tokens/vectors.
  - The model must pre-process the text within the dataset into specific categories.

- The model must clean the text such that cleaning involves the removal of empty fields within a CSV file, any useless or incorrect punctuation, and unnecessary stop words.
- The model must take the clean data and categories stem words.
- The model must produce a machine learning implementation that learns and is trained on sample data that is then extrapolated into a useable asset.
- The model must classify text.

• ***Should Have***

  - The model solution should be agnostic towards data types, data sensors, vendor (mostly universities or colleges) and data creation date.
  - The model should report analysis in real-time with graphs or training data within the CLI, this includes any abnormalities which might need deeper analysis to be useful data.
  - The model must execute statistical analysis on the yielded information which is generated for future examination.

• ***Could Have***

  - The model could provide an analytical solution in which communicates with processing features.
  - The model could create an interactable alert so the user can decide on how to proceed.
  - The model may predict future student evaluations.
  - The model could save outputted data to a local text file
  - The model could have param-arguments for it's runtime, most likely in the form of CLI arguments, this allows the model to execute commands in a certain order, i.e —parse to select the input dataset.

• ***Won't Have***

  - The model won't have a GUI (can be developed for future work).

4.5 Non—Functional Requirements

- ***Must Have***
  - The model must provide and produce an accurate F-Score.
  - The model must be maintainable within its set scope, overdeveloping or under developing can lead to bugs or broken links such as outdated APIs.
- ***Should Have***
  - The model may yield predictions or results limited to the scope of a set asset.
  - The model may yield results across several datasets.
  - The model must yield maximum theoretical performance for its implementation, this includes:
    * Correct potential true positives
    * Correct potential false positives
    * Account for and correct false negatives
    * Recall of data and specifics
  - The model should yield optimal precision of data and classification.
  - The model should run within an acceptable time-frame for a given machine, e.g., testing must not run over 24hrs for an appropriate dataset.
  - The should must be useable for a lay person who wants to classify text (school admin).
- ***Could Have***
  - The model may be scalable for multiple datasets or machines.
- ***Won't Have***
  - GUI

4.6 Challenges in Requirements

The biggest challenge this project faced is within the defined must-have functional requirements, axiomatically being able execute functions related to the scope and very definition of this deliverable; classification functionality has been the hardest challenge to implement.

4.7 Cost Prediction

This project will not have any costs associated with or throughout the development. However, future development may include renting server space for better spec machines to run training of this model.

Chapter 5

Project Design - New

This chapter outlines the design behind each component of the text classification model and the respected process each component may entail, several components will execute more than one step to achieve the desired result. The design is reflected within the final model and each component is broken down to display the functionality and theory within this project. In section 4.4, it is detailed there won't be a GUI for the interaction and thus the design section relates to the inner workings of the model itself, that being: system architecture, logistics and theory.

5.1 Classical vs Modern

As originally intended, this project would have demonstrated two differing implementations of the same concept, one being of a classical nature implemented as a machine learning model and the other being a modern variation of the same approach, as previously mentioned this project experienced time management issues due to unexpected problems, to which resulted in only focusing on a traditional implementation within a modern model in attempt at a novel approach.

Classical NLP implementations can be described as:

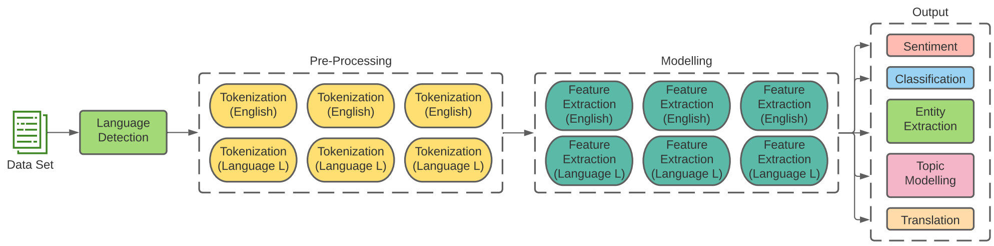


Figure 5.1: Classical NLP pipeline for text classification.

Classical or "traditional" methods for corpus classification include: N-Grams, Hidden Markov Models using Markov Chains, Part-Of-Speech Tagging and Bag-Of-Words. Machine learning concepts can be classed as a "black-box" of functionality as the user does not necessarily see what is being executed within the hidden layers, a high-level abstraction for this project can be generalised into the following diagram:

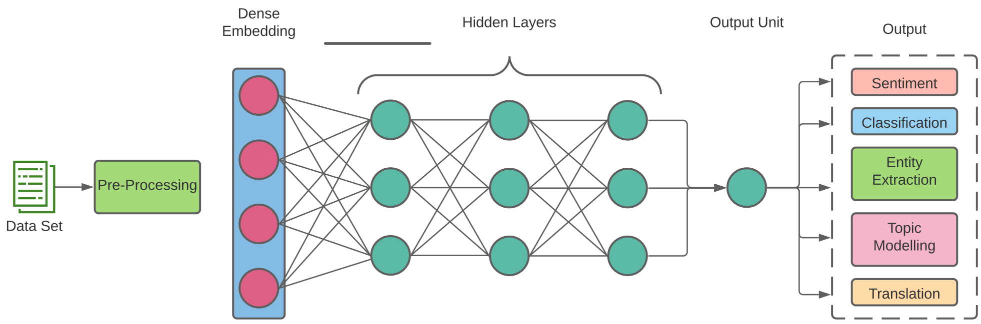


Figure 5.2: Machine Learning Model for an NLP pipeline for text classification.

5.1.1 Model Approach — Traditional

The traditional implementation would have been designed based on word-embeddings within a 2D space where word vector values would be used alongside the Bag-Of-Words approach. The combination of Bag-Of-Words with a machine learning model to calculate a word's vector based on its TF-IDF value would have been the start, such that:

$$tf(t,d) = \frac{f_{t,d}}{\sum t \in_d f_{t',d}} \tag{5.1}$$

Where the TF-IDF value is how often a lexeme occurs in a given corpus and would have also used the inverse TF-IDF value as the project model covers multiple datasets, such that:

$$idf(t,D) = \log \frac{N}{|d \in D : t \in d|} \tag{5.2}$$

Where the inverse TF-IDF value is represents how rare a word is in a given corpus.

5.1.2 Limitations of Bag-Of-Words

As there is a specific domain for this project, it is important to focus on the limitations for potential methods, CBOW models have demonstrated high training accuracy and is a relatively simple model to implement, it is naturally flexible as it can be trained on different datasets for a specific context; however, it does have disadvantages for classification and text prediction. Universities host students from different backgrounds and levels of education which can imply issues when training datasets, these issues can impact: the

- ***Vocabulary:*** Students may have varying output to describe the same context, this can create confusion for training.
- ***Frequency:*** The frequency of a word may influence its power in a dataset.
- ***Context:*** If students describe the same situation in a different way, some words may lose semantic meaning or be interpreted incorrectly, for example isolating neologisms, synonyms, colloquialism, polysemes, or semantic change.

5.1.3 Model Approach — Modern

Modern or "contemporary" methods for corpus classification include: . . . The modern implementation would have been based around Word2Vec implemented in a machine learning model for Sentiment Analysis which can be seen as a subcategory of text classification. The word2vec implementation would produce vector values for identified key words and plot them as such:

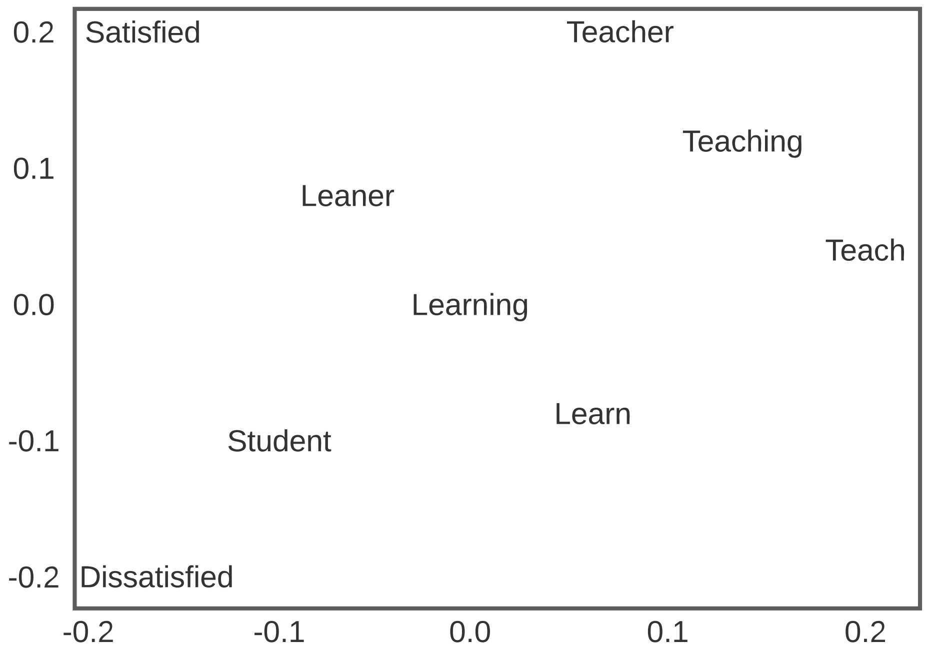


Figure 5.3: Example of student satisfaction vector graph in 2d space.

5.1.4 Word2Vec Skip-Gram

Skip-gram rather than Continuous Bag-Of-Words (CBOW [which is the modern implementation of Bag-Of-Words]) as it yields better results with large datasets - such as student feedback - skip gram can also be context aware as it converts neighboring lexemes to vectors.

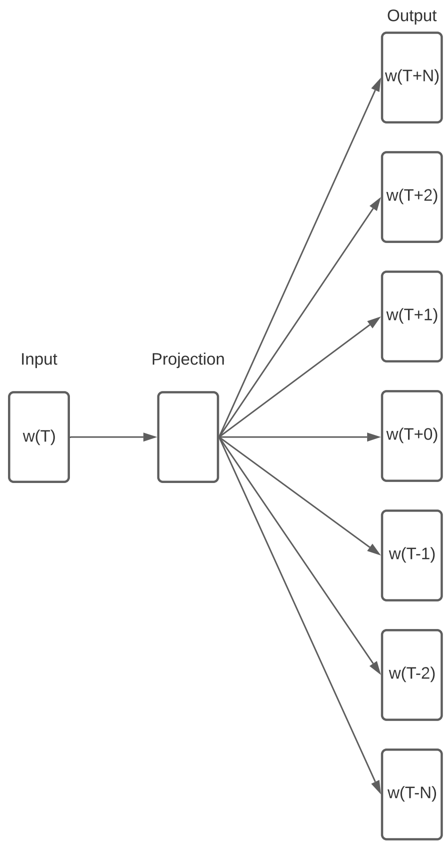


Figure 5.4: Input flow of Skip-Gram model.

The skip-gram model can be seen as the inverse of the bag-of-words model as it attempts to vectorize neighboring words first to identify corpus context, whereas, bag-of-words takes each lexeme first, produces a vector sum and then categorises each word.

The Skip-Gram's network architecture is similar to the diagram displayed in Figure 5.2, the diagram above is another level of abstraction as to how this project will make use of machine learning fundamentals.

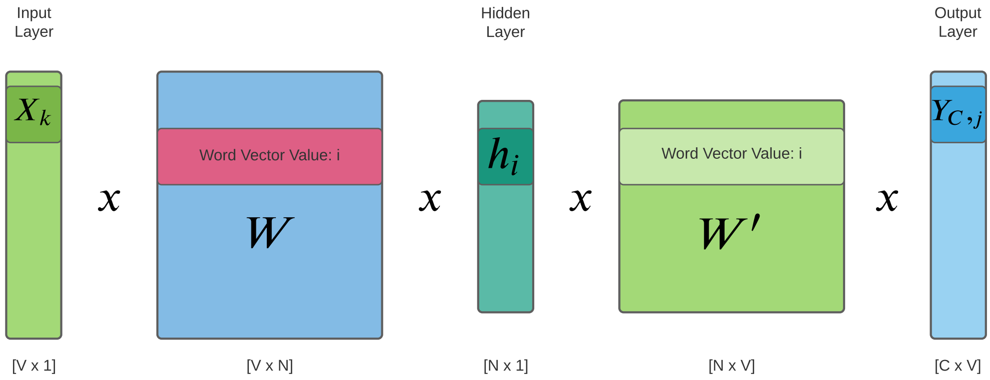


Figure 5.5: Skip-Gram implementation for Word2Vec network architecture.

5.1.5 One-Hot Encoding

Encoding each significant lexeme in a given dataset is helpful for the model to distinguish the level of importance, outlined as context, the diagram shown above in Figure 5.4 takes a 1D row vector as the input layer which allows for the model to acceptably one-hot encode each lexeme unit as a numeric value.

$$"Teach" = \begin{bmatrix} 0, & 0, & 0, & 0, & 1, & 0 \end{bmatrix} \qquad "Teaching" = \begin{bmatrix} 0, & 0, & 0, & 0, & 1, & 0 \end{bmatrix}$$

This equates to differing versions of the same word (root + free or bound morpheme [affix]) to have the same influence when being parsed forward within the training model.

5.1.6 Forward Propagation

Once the corpus text is encoded as an acceptable 1D matrix, it can then be parsed to the first hidden layer's node:

$$h = x^T W \tag{5.3}$$

'x' represents the row vector, 'h' can be taken as the column height ($[x]^T$ of the vector 'W' where 'h' is the 'k_{th}' column.

$$h = W^T_{(k,:)} := v^T_{w_I} \tag{5.4}$$

As each lexeme unit parses through the hidden layer nodes, the exit value needs to be calculated:

$$u_c = W'^T h = W'^T W^T x \tag{5.5}$$

For simplicity, u_c will transpose each unit through the model in its entirety, however, the value(s) of u_c cause performancy and memory issues in the model. Thus, ***softmax*** will be used to slice the vector value of u_c to [0,1].

$$y_c = Softmax(u)$$

$$p(w_{c,j} = w_{O,c}|w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{j'})} \tag{5.6}$$

'x' represents the center word $w(T)$ of a vector where W' will yield the softmax value of u_c, this ensures the vector of $w(T)$ will have exact values of input for 'x', where:

$$u_{c,j} = u_j = {v'_{w_j}}^T \cdot h \qquad c = 1 \ldots |C| \tag{5.7}$$

v'_{w_j} represents the exit vector for the j^{th} lexeme unit in w_j (the corpus vocabulary).

5.1.7 Backward Propagation

As stated below Equation 5.5, the transposition of each unit u_c is simple where node weight is not considered. To account for the weight of the model matrices

W and W', the Stochastic Gradient Descent method is applied, which in turn will optimise the backward propagation of node errors; when training the model, errors are bound to occur, to which a loss function is needed to calculate each layers efficiency.

$$\begin{aligned} Error(E) &= -\log \mathbb{P}(w_{O,1}, \ldots, w_{O,c}|w_I) \\ &= -\log \prod_{c=1}^{C} \frac{exp(u_{c,j_c^*})}{\sum_{j'=1}^{V} exp(u_{j'})} \\ &= -\sum_{c=1}^{C} u_{j_c^*} + C \cdot \log \sum_{j'=1}^{V} exp(u_{j'}) \end{aligned} \tag{5.8}$$

j_c^* represents the exit vector column's index within the vector's vocabulary. Once loss is calculated, the model can apply the ***chain rule*** for weight error classification within W and W'. This is achieved by obtaining the partial derivative of Error(E) of each exit node $u_{c,j}$.

$$\frac{\partial E}{\partial u_{c,j}} = y_{c,j} - t_{c,j} := e_{c,j} \tag{5.9}$$

$t_{c,j}$ represents vector u's ***Ground Truth***; $[t_{c,j}]$'s state of hypothesis can be represented as the following definition:

$$EI_j = \sum_{c=1}^{C} e_{c,j} = \sum_{c=1}^{C} (y_{c,j} - t_{c,j}) = \frac{\partial E}{\partial u_j} \tag{5.10}$$

EI_j represents the ***Row-Wise Sum*** as a column vector in attempt to account for word context errors for $w(T)$; ***backpropagation*** can then occur by obtaining the partial derivative of Error(E) within matrix W'.

$$\begin{aligned} \frac{\partial E}{\partial w'_{ij}} &= \sum_{c=1}^{C} \frac{\partial E}{\partial u_{c,j}} \cdot \frac{\partial u_{c,j}}{\partial w'_{i,j}} \\ &= \sum_{c=1}^{C} (y_{c,j} - t_{c,j}) \\ &= EI_j \cdot h_i \end{aligned} \tag{5.11}$$

When backpropagation occurs, the Stochastic Gradient Descent is redefined in terms of the matrix W':

$$w_{i,j}'^{(new)} = w_{i,j}'^{(old)} - \eta \cdot EI_j \cdot h_i \tag{5.12}$$

η represents the model's training (learning). Once the learning rate has been established, the model can update the distribution of errors between the model layers, particularly between unit input and the hidden layer whereby the partial derivative is weighted against an error from a hidden layer. This can be calculated with:

$$\begin{aligned} \frac{\partial E}{\partial h_i} &= \sum_{j=1}^{V} \frac{\partial E}{\partial u_j} \cdot \frac{\partial u_j}{\partial h_i} \\ &= \sum_{j=1}^{V} EI_j \cdot w_{ij}' \end{aligned} \tag{5.13}$$

As errors are calculated between the input layer and the hidden layer accounting for error weight, it is possible to calculate the weighted loss of matrix W by taking the partial derivatives of Error(E) against the partial derivative of an index within matrix W, as such:

$$\begin{aligned} \frac{\partial E}{\partial W_{ki}} &= \frac{\partial E}{\partial h_i} \cdot \frac{\partial h_i}{\partial w_{ki}} \\ &= \sum_{j=1}^{V} EI_j \cdot w_{ij}' \cdot x_k \end{aligned} \tag{5.14}$$

The weight of the Stochastic Gradient Descent is redefined in terms of the matrix W:

$$w_{i,j}^{(new)} = w_{i,j}^{(old)} - \eta \cdot \sum_{j=1}^{V} EI_j \cdot w_{ij}' \cdot x_j \tag{5.15}$$

The fundamental functionality has been outlined in theory but in practice will be enough to train the given Skip-Gram Network in Figure 5.5.

5.2 Planning the ML Model Design

Initial prototyping of the machine learning model for a new amalgamation of NLP techniques helped to indicate what the best route of development could be, the planning stage piggybacked off existing work flowchart diagrams in-order to apply the most appropriate method and technique combination. The sequence flowchart is as follows:

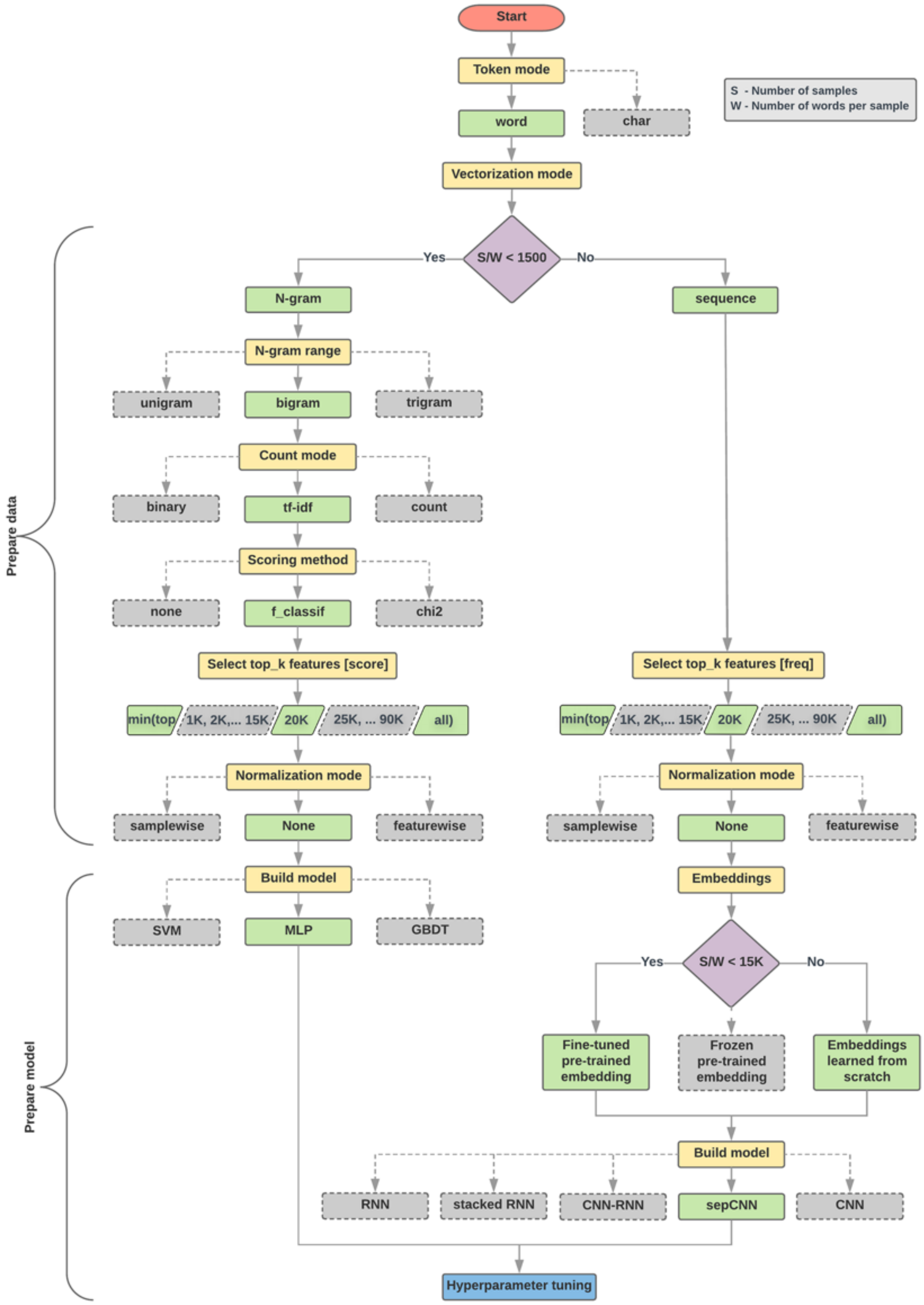


Figure 5.6: Text Classification Flowchart (Google, 2021).

5.3 Supervision

The development of the project model is based on a supervised approach due to the datasets located, it was most appropriate to use a supervised approach due to the datasets having no labels or lexical categories to train the model on; the model has user input to account for missing labels on data which have been manually and algorithmically added. The supervision for this project can be represented as the following diagram:

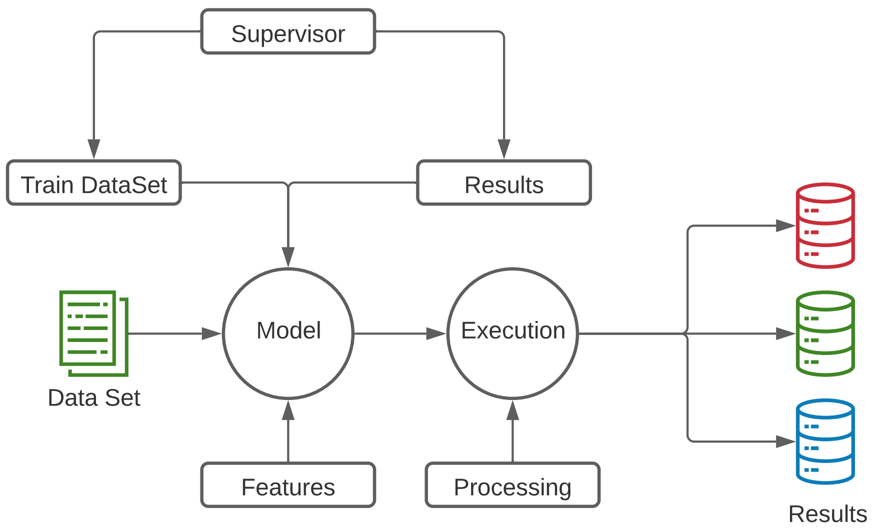


Figure 5.7: Sequence control for Supervised learning.

5.4 Pipeline Design

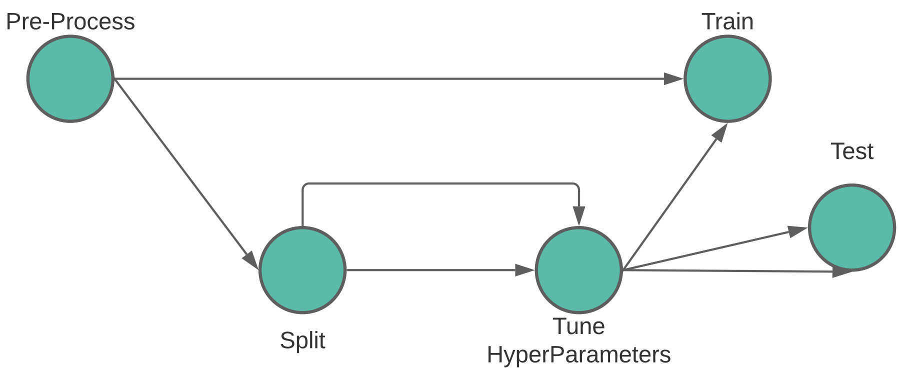


Figure 5.8: Pipeline for model classification.

There are five main steps for a text classification pipeline:

1. ***Pre-processing***: prepare the raw dataset to be trained.
2. ***Splitting***: split the processed dataset to be trained and validated.
3. ***Tuning***: identity valuable parameters within trained data.
4. ***Training***: train the current iteration of the model with updated hyper-parameters.
5. ***Testing***: test and collect statistics for analysis to make further predictions.

5.5 Data Prep, Pre-processing, and Analytics

Preparing and pre-processing of the parsed data enables meaning to be allocated to the processed classification information, bypassing the implementation of a pre-processing step will ruin the chances of yielding valid returned data and will most likely detract meaning from any results throughout data analysis. The preperation and pre—preprocessing of data will be completed with the use of NLP libraries such as ‘pandas’ and ‘sklearn’.

5.5.1 Data Organisation and Binding

The initial steps for data organisation will be to tidy the datasets procured from Kaggle as they held a lot of useless information and or columns with little weight. The first organisation step is to merge multiple CSV files and their respected columns into an easily readable format. The removal of unnecessary columns will aid the preprocessing as the dataset will be smaller.

5.5.2 Pre-processing

The following methods of preprocessing will be applied to a given dataset:

- ***Punctuation:*** removal of all unnecessary punctuation, where punctuation includes the regular expression: `["[.!?\\-]"]`.
- ***Missing Values:*** word features with a percentage higher than the defined threshold for missing values will be sliced from the dataset as missing features will cause unreliable training data.
- ***Collinear features:*** features in which are highly correlated with another lexical unit will be removed as these features can decrease overall performance on training data, this is due to higher variance of the similar features and lower interpretability of the model.

Chapter 6

Implementation and Testing

This chapter focuses on the implementation stage of this project and discusses the development of the artefact's codebase; the development of this project is dissected into its respected categories, such as each significant chunks of code and overall testing. As stated in section 1, the project will form an executable deliverable which is expected to meet the requirements outlined in section 4.4 and section 4.5 which forms the Minimum Viable Product (MVP) until the final production release. The release of the final deliverable will handled in a staged environment to be assessed.

6.1 Linking to Methodology

This project was developed in an agile manner with continuous development as seen in Figure 3.3, it had one major iteration to which adaptations built into the existing codebase; if test features were needed, a separate git branch was created as insurance the existing (working) code was not affected. The implementation phase is broken down into two key sections as this allows the developer to assess where or not the methodology is functioning as desired with their chosen tech-stack.

6.2 Elected Programming Environment

6.2.1 Development Environment

This project has made use of popular development environments with their own intended purpose:

- ***PyCharm:*** for general development.
- ***Jupyter NoteBooks:*** for data organisation and visualization.
- ***Kaggle NoteBooks*** for open-source testing and comparison.

6.2.2 Language Choice and Justification

There are more appropriate languages to consider for this project as the nature is how differing theory implementations affect performance and accuracy; C++ was considered at the beginning of this project due to its performance and execution speeds, popular machine learning libraries are written in C++ such as TensorFlow and ported to Python with the use of Cython. However, the learning curve for writing machine learning in C++ is a lot steeper than in Python to which Python was the chosen language for ease of use and development time.

As the artefact relies on programming knowledge, it was essential to pick a language the developer is comfortable writing in as learning a language for a specific use may have required dedicating too much time, especially with time-management overhead. This was beneficial as the developer has previous experience with Python and is relatively fluent.

As stated above, many a data mining & analytics and machine learning libraries are written for or in Python which also increases the ease of use for project development with specific traits, such as NLP with ML. This project's interests are heavily backed with open-source development of Python libraries which ensure the imports are relatively low costing and the codebases work as efficiently as possible for a given task. This makes Python a well suited language for machine learning projects and is almost the goto language for data-science related tasks.

6.2.3 Language libraries and Justification

- ***NLTK:*** Natural Language ToolKit for Python
- ***Pandas:*** Popular Python Library for data-science, particularly, data manipulation and formatting.
- ***TensorFlow:***
- ***PyTorch***

6.3 Project Structure

6.4 Data Preparation

6.4.1 Data Preprocessing

Withinn subsection 5.5.2

Punctuation

Missing Values

Collinear Features

6.4.2 Data Presentation

6.4.3 Data Parsing

6.5 Implementation — Traditional

```
class POSTagger:
    def __init__(self, dataset, training_data):
        self.dataset = []
        self.training_data = []

    @classmethod
    def train(dataset)
        return dataset
```

6.6 Implementation — Contemporary

```
class Word2Vec_SkipGramModel:
    def __init__(self, dataset, training_data)
        self.dataset = []
        self.training_data = []

    @classmethod
    def train(dataset)
        return dataset
```

6.7 Data Analytics

6.8 Issues with Solutions

6.8.1 Issues and Known Bugs

6.9 Project Testing

Chapter 7

Evaluation and Testing

This chapter critically evaluates the final artefact and its development process against the previously defined constraints, functions, and methodology outlined within chapters: one, three, four, and the Project Initiation Document displayed within Appendix A.

7.1 Aims and Objectives Evaluation

Aims and Objectives

Aim and Objective	***Description***	***Satisfied - Yes/ No***
21	22	24
31	32	34
41	42	44

Table 7.1: Evaluation table for Aims and Objectives

7.2 Methodology

Methodology

7.2.1 Project Plan

Project Planning Evaluation

7.3 Requirement Evaluation

Requirement Evaluation

7.3.1 Functional Evaluation

Requirement	***Desired Outcome***	***Details***	***Satisfied - Yes/ No***
21	22	23	24
31	32	33	34
41	42	43	44

Table 7.2: Evaluation table for Functional Requirements

7.3.2 Non-Functional Evaluation

Requirement	***Desired Outcome***	***Details***	***Satisfied - Yes/ No***
21	22	23	24
31	32	33	34
41	42	43	44

Table 7.3: Evaluation table for Non-Functional Requirements

7.4 Artefact Evaluation

Artefact Evaluation

7.4.1 Testing

Chapter 8

Future Work

This chapter focuses on possible amendments for this project, be it design or structural alterations for potential ideas to be constructed. The development carried out throughout this project has seen breakpoints which have led to new implementational ideas based on this project's scope, entirely novel models which are a result of research (discussed in chapter 2). This chapter breaks down those ideas into their respected backgrounds and outlines future work and life for this project.

8.1 Derived from Literature Review

As discussed in section 2.4.3, gaps can be identified within the research conducted, this section will focus on furthering the statements to explain how the literature review exposed less explored theoretical concepts and their counterpart implementation.

- (Adaptation) — ***Compare POS-Tagging and Enhanced versions of Word2Vec with Machine Learning Implementations for Sentiment Analysis and Text Classification***. Previous comparative studies focus on traditional versions of Word2Vec implementations, whereby it is often to see similar group-set of NLP and ML techniques being used in the results, this is a small change but could have a big impact for comparative studies and domain analysis. As ML concepts are now able to make use of more computational power, it is possible to phase out the testing of traditional techniques in favour for their ML adaptation (fundamentals do not change).

- (Novel Implementation) — ***Investigate the use of Machine Learning with Support-Vector Machines (SVM) and use Neural Networks or Deep Learning for Ensemble Learning applied Sentiment Analysis on Student Feedback.***

8.2 Derived from Artefact

8.2.1 Traditional Model

N.Y.C

8.2.2 Contemporary Model

Phrase Generation

$$\boldsymbol{score}(w_a, w_b) = \frac{\boldsymbol{count}(w_a w_b) - \delta}{\boldsymbol{count}(w_a) \times \boldsymbol{count}(w_b)} \tag{8.1}$$

Subsampling

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{8.2}$$

Negative Sampling

$$\log p(w|w_i) = \log \sigma({v'}_w^T v_{w_I}) + \sum_{i=k}^{K} E_{w_i P_n(w)} \left[\log \sigma(-{v'}_w^T v_{w_I})\right] \tag{8.3}$$

8.3 Derived from Observations

When developing and training this project's artefact, ethics were taken into consideration and it was decided to only make use of open-source (predefined) datasets, this decision limited the search for useable data and as a result, this model made use of two datasets from higher academic institutions. It would be beneficial to have access to more data or datasets for higher accuracy when training.

Expanding the model to an outside host, this project was developed within an isolated environment (the developer's personal environment) as the model was easier to contain and maintain external variables. This resulted in limiting the nature of the model and its potential scope as resources were limited, however, developing in this state did minimise risk and lower the fault tolerance of the text-classification model. It would be beneficial to containerise this model and execute training on a more capable machine such as a higher core server.

The original intention for this project included the planning, design, and development of multiple NLP machine-learning models to put against each other to see how different implementations of theory may affect performance when trained on different datasets and NLP domains, however, due to unforeseen circumstances, this project experienced several difficulties with management and overall development. If this project were to have further development, it would be within reason to explore deeper theoretical combinations as discussed in section 8.1. The project's scope and limitations would not differ as there is a pretrained and predefined model to use as a reference point.

Chapter 9

Conclusion

Even though the mordern approach was not implemented, the mathematical foundations it is built on suggests it is a viable model as an alternative amalgamation with a potential accuracy of 96% and a loss value of 40.65.

Appendices

Appendix A

Project Initiation Document

School of Computing
Project Initiation Document

William Green

Domain analysis of feature implementations between Classic and Deep NLP models.

PJE40

1. Basic details

Student name:	William Green
Draft project title:	Domain analysis of feature implementations between Classic and Deep NLP models.
Course:	BSc Computer Science - C0056S
Project supervisor:	Dr. Alaa Mohasseb
Client organisation:	N/A
Client contact name:	N/A

2. Degree suitability

Please describe how your project satisfies the criteria for your current course. For example, if you are a Software Engineering student, please explain why your project is suitable for a Software Engineering degree.

In each section, please write your text below ours in regular (non-italic) font.

The proposed project will be suitable for a BSc Hons Computer Science degree because it will include current areas of interest and research, I have chosen to base my project on Natural Language Processing which will identify different model implementations. For this project, I will be including classical models, machine learning and programming skills acquired throughout my time at University.

When choosing my project, I sought to find a challenging but manageable idea which includes opportunities to learn various aspects of Computer Science Theory, specifically how I can view mathematical models and representations of current Computer Science Theories. I am taking on a new area of interest, one that has not been formally taught to me and am taking full advantage of the resources offered dedicated to my degree. My degree teaches me how to process information and create solutions based on know-how and intuition with an efficient result; this project will test the past 2 years of my undergrad and present new philosophies of data-science.

Throughout my project, I will be mainly programming in Python, in which I will be implementing (Convolutional) Neural Networks to train a machine learning model to apply transformations on a dataset. “Computer science is the study of algorithmic processes and computational machines”, I aim to look at several algorithmic implementations and features to best decide the efficiency between classic models and a newer if not new implementation of how I can compute information.

3. Outline of the project environment and problem to be solved

*For engineering **projects without a client**:*	*For projects **with a client**:*	*For theoretical or **study** projects:*
What is the problem that you will investigate? *Why is it worth working on?*	*Who is the client?* *What do they do?* *What is their problem?* *Why does it need to be solved?*	*Who is the intended readership/audience?* *What is the contextual significance of this topic?* *What are the research questions you are seeking to answer?*

My project focuses on the efficiency of different NLP models and how they interact with their respected datasets; there are many domains within NLP which have their subsequent areas of interest, I will be looking for variable changes as to how classical approaches differ to deep-learning implementations of the same domain. In this instance, can the use of deep learning affect the performance results of text classification using ensemble learning.

Research suggests there are inconsistencies in newer implementations as buzzwords are being attached to different problems, it seems this idea is novel as it is based upon current research which has not applied these adjacently applied these techniques. It is worth the time and effort as it is new and may have potential to give valuable insight when designing or merging algorithmic approaches.

4. Project aim and objectives

What is the overall aim of the project?
What are the objectives that will lead to you meeting that aim?

This project seeks to outline any performance differences between classical NLP models and the use of deep learning to existing approaches; ideally, this project aims to profile and focus on any form of performance gains by applying CNNs to classical models such as the bag-of-words concept.

I will achieve this by creating three to five different NLP models with their deep learning counterpart, this will allow me to have enough variation in my results to ensure there is no overlap and that the results are accurate.

5. Project deliverables

For an engineering project, what information system artefacts will be developed? What documents will be produced? This always includes your project report but could also include supporting documentation for your client such as requirement and design specifications, test strategies, user guides, that are useful outside of the project report.

For a study project, are there anticipated outcomes besides the report, for example datasets or recommendations to external bodies?

This project will have several direct and related deliverables with a detailed final report, my report will include all, and any python source code developed throughout this dissertation, this will most likely be handled by screengrabs and version control software in-order to display sequential updates in the form of commits. I intend to produce and deliver three to five deep learning models each possessing different characteristics.

Title	**Type**	**Description**
Dissertation Report	PDF/ Report	A concise overview of the project.
Source Code	.py files	Python code written during report.
Machine Learning Networks	.py files and mathematical models	Evidence to support my report.
Results	Graphs and Tables	All outcomes of this project.

6. Project constraints

What constraints are there on your solution to the problem? For example, you could not test a medical system on real patients.

This project will likely come with constraints common to most projects.

Constraint	**Description**
Sample Size	The chosen data set may not enough entries to produce valuable results.
Lack of Reliable Data	My research and or results may not produce reliable/ consistent data.
Lack of Prior Research	As this idea is seemingly novel, there may not be enough supporting material to use as guidance for improvement.
Lack of Technical Ability	Oneself may not be able to produce the required program.
Self-Reported Data	The quality of the results may be questionable.
Time	We can only devote so much time to a dissertation, it may not be enough to achieve a level of satisfaction or desired outcome.

7. Project approach

How will you go about doing your project? What background research do you need to do? For an engineering project, how will you establish your requirements? For a study project - can you refine your larger research area into research questions that you can meaningfully answer? What skills do you require and how are you going to acquire those that you do not already have? What methodologies are you going to use?

I will conduct preliminary research of a wide-scale approach, using existing research to gain knowledge on the individual concepts and interconnectivity between them, I plan to use this research to support my hypothesis and adapt any future work.

I will then conduct specifically targeted research to find possible gaps in existing research to support why my idea may be a viable option to fill potential gaps.

Once I have ironed out the theoretical concepts I will be using, I will start to look at language specifics, in this case Python Libraries and or Frameworks I can use to aid the development of this project's program. I am aware of libraries such as: NumPy, SciKit-Learn, NLTK and a higher-level abstraction service WordNet API for python.

I will approach my project problem with a logical methodology, in this case it is most effective to start with the Top-Down approach as it implies the problem can be compartmentalized into smaller modules and tackled separately until there is a full hierarchical module. Starting with a base idea, I can have a high-level approach and engineer lower-level aspects as I get deeper into the problem set.

8. Literature review plan

What are the starting points for your research? (e.g. specific books or papers in journals, existing reports or documents, online resources, existing systems)

For my literature review, I will start by looking at the combined words of Andrea Ferrario and Mara Naegelin for their paper titled "The Art of Natural Language Processing: Classical, Modern and Contemporary Approaches to Text Document Classification" as it looks at preprocessing of a dataset and classical bag-of models compares with NNs for text classification. In addition, I will be reading "Combining Machine Learning and Natural Language Processing to Assess Literary Text Comprehension" by Balyan and McCarthy due to its nature of applying machine learning to an ensemble transformer. These are starting points for my research.

9. Facilities and resources

What computing/IT facilities will you use/require?
What other facilities/resources will you use/require?
Are there constraints on their availability? If funds are required to acquire them, have these been allocated? Will they be available in time?
For example, you might need a specialist lab or equipment at the university, which might be in use in teaching and by other project students. Your own computer and free software, or software you already have, do not normally need to be mentioned.

For this project, I will not require the use of university facilities or any external resources, I do not plan for this project to incur costs.

10. Log of risks

What risks will you encounter when doing your project? What backup plans do you have if identified things go wrong?
What is your plan for reviewing risks? Remember that risk probabilities, and hence priorities, will change over the course of the project, so this section should be maintained.
Use a table like below.

Description	Impact	Likelihood	Mitigation	First indicator
COVID-19 outbreak means I cannot get into a lab for usability testing	*Severe*	*Likely*	*Get in while I can, prioritise lab tasks in time.* *Make an alternate test plan that does not need the lab.*	*University informs that lab closure is likely*

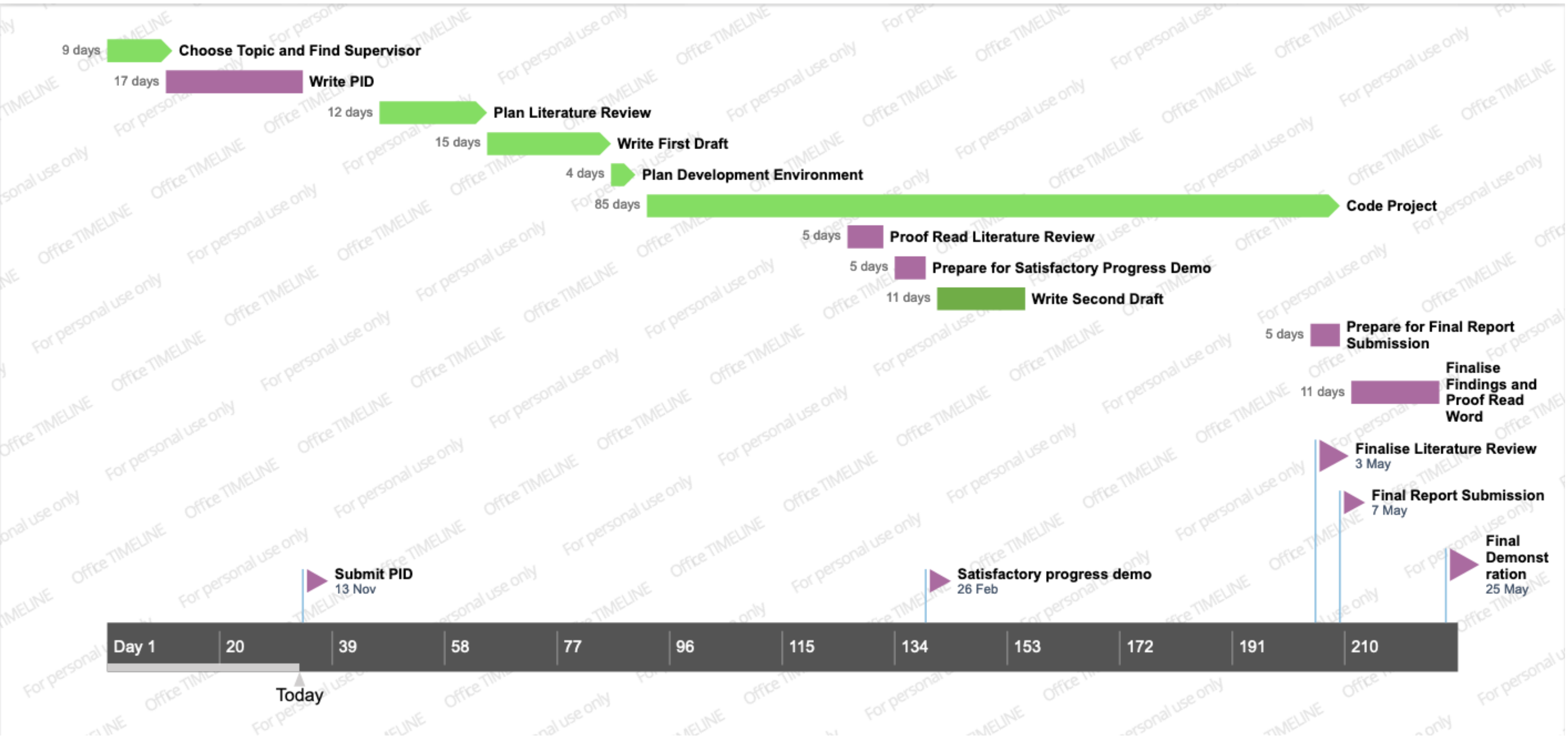
Description	**Impact**	**Likelihood**	**Mitigation**	**First indicator**
Time Issues	Severe	Likely	Create a detailed plan for time management allowing for unforeseen circumstances.	Falling outside of the created plan.
Learning Curve	Severe	Likely	Prepare research and give enough time to learn new material.	Struggling to conceptualize new material and implement theoretical aspects.

Requirements are not well defined	Severe	Unlikely	Thorough requirement elicitation	Project scope changes.
Unplanned Work relating to knowledge gaps	Moderate	Likely	Rely on current programming skills and logic overcome academic challenges	Struggling to complete or implement theoretical aspects.
dependency issues	Low - Moderate	Likely	Virtual Environments for Python	Dependencies do not work.

11. Project plan

What do you need to do to create the artefact / do the primary research and write the report? Walk through your proposed approach and break it down into tasks.
When are you planning to perform these tasks? When do you need access to other people or resources? Usually a Gantt chart is a good way of presenting the plan.
Note that plans can change over the course of the project, so this plan should be maintained.

Project overview Gantt Chart



v 2020-09

Appendix B

Ethics Review

Certificate of Ethics Review

Project Title: Domain analysis of feature implementations between Classic and Deep NLP models.

Name: WILL GREEN **User ID:** 853829 **Application Date:** 12-Nov-2020 23:24 **ER Number:** ETHIC-2020-1528

You must download your referral certificate, print a copy and keep it as a record of this review.

The FEC representative for the School of Computing is Carl Adams

It is your responsibility to follow the University Code of Practice on Ethical Standards and any Department/School or professional guidelines in the conduct of your study including relevant guidelines regarding health and safety of researchers including the following:

- University Policy
- Safety on Geological Fieldwork

It is also your responsibility to follow University guidance on Data Protection Policy:

- General guidance for all data protection issues
- University Data Protection Policy

Which school/department do you belong to?: **SOC**
What is your primary role at the University?: **Undergraduate Student**
What is the name of the member of staff who is responsible for supervising your project?: **Dr Alaa Mohasseb**
Is the study likely to involve human subjects (observation) or participants?: **No**
Are there risks of significant damage to physical and/or ecological environmental features?: **No**
Are there risks of significant damage to features of historical or cultural heritage (e.g. impacts of study techniques, taking of samples)?: **No**
Does the project involve animals in any way?: **No**
Could the research outputs potentially be harmful to third parties?: **No**
Could your research/artefact be adapted and be misused?: **No**
Does your project or project deliverable have any security implications?: **No**
Please read and confirm that you agree with the following statements: **Confirmed**
Please read and confirm that you agree with the following statements: **Confirmed**
Please read and confirm that you agree with the following statements: **Confirmed**

Supervisor Review

As supervisor, I will ensure that this work will be conducted in an ethical manner in line with the University Ethics Policy.

Supervisor signature: **Date:**

Bibliography

Akinsola, J. E., Ogunbanwo, A. S., Okesola, O. J., Odun–Ayo, I. J., Ayegbusi, F. D., & Adebiyi, A. A. (2020). Comparative analysis of software development life cycle models (sdlc), 310–322.

Alqurashi, E. (2019). Predicting student satisfaction and perceived learning within online learning environments. *Distance Education*, *40*(1), 133–148.

Assaf, P. (2021). *Towards a development methodology for machine learning - part 1*. https://assaf-pinhasi.medium.com/towards-a-development-methodology-for-machine-learning-part-i-f1050a0bc607

Belotto, M. J. (2018). Data analysis methods for qualitative research: Managing the challenges of coding, interrater reliability, and thematic analysis. *Qualitative Report*, *23*(11).

Beran, T., Violato, C., & Kline, D. (2007). What's the" use" of student ratings of instruction for administrators? one university's experience. *Canadian Journal of Higher Education*, *37*(1), 27–43.

Burgess, S., & Sievertsen, H. H. (2020). Schools, skills, and learning: The impact of covid–19 on education. *VoxEu. org*, *1*(2).

Dale, R. (2019). Nlp commercialisation in the last 25 years. *Natural Language Engineering*, *25*(3), 419–426.

Edalati, M. (2020). The potential of machine learning and nlp for handling students' feedback (a short survey). *arXiv preprint arXiv:2011.05806*.

Eisenstein, J. (2019). *Introduction to natural language processing*. MIT press.

Enriquez, F., Cruz, F. L., Ortega, F. J., Vallejo, C. G., & Troyano, J. A. (2013). A comparative study of classifier combination applied to nlp tasks. *Information Fusion*, *14*(3), 255–267.

Google. (2021). Text classification flowchart. https://developers.google.com/machine-learning/guides/text-classification/images/TextClassificationFlowchart.png

Heift, T., & Hegelheimer, V. (2017). Computer–assisted corrective feedback and language learning. *Corrective feedback in second language teaching and learning*, 51–65.

Jain, A., Kulkarni, G., & Shah, V. (2018). Natural language processing. *International Journal of Computer Sciences and Engineering*, *6*(1), 161–167.

Jensen, E., Dale, M., Donnelly, P. J., Stone, C., Kelly, S., Godley, A., & D'Mello, S. K. (2020). Toward automated feedback on teacher discourse to enhance teacher learning. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–13.

Jenson, K. (2012). Crisp–dm process diagram.png. https://commons.wikimedia.org/wiki/File:CRISP--DM_Process_Diagram.png

Kandhro, I. A., Chhajro, M. A., Kumar, K., Lashari, H. N., & Khan, U. (2019). Student feedback sentiment analysis model using various machine learning schemes: A review. *Indian Journal of Science and Technology*, *12*(14).

Kastrati, Z., Arifaj, B., Lubishtani, A., Gashi, F., & Nishliu, E. (2020). Aspect–based opinion mining of students' reviews on online courses. *Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence*, 510–514.

Kastrati, Z., Imran, A. S., & Kurti, A. (2020). Weakly supervised framework for aspect–based sentiment analysis on students' reviews of moocs. *IEEE Access*, *8*, 106799–106810.

Kayali, M., & Alaaraj, S. (2020). Adoption of cloud based e–learning in developing countries: A combination a of doi, tam and utaut. *Int. J. Contemp. Manag. Inf. Technol*, *1*(1), 1–7.

Keane, E., & Labhrainn, I. (2005). Obtaining student feedback on teaching & course quality. *Brie ing paper*, *2*, 1–19.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Ongsulee, P. (2017). Artificial intelligence, machine learning and deep learning. *2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE)*, 1–6.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Perikos, I., Grivokostopoulou, F., & Hatzilygeroudis, I. (2017). Assistance and feedback mechanism in an intelligent tutoring system for teaching conversion of natural language into logic. *International Journal of Artificial Intelligence in Education*, *27*(3), 475–514.

Raunak, V., Gupta, V., & Metze, F. (2019). Effective dimensionality reduction for word embeddings. *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP–2019)*, 235–243.

sas.com. (2021). *What is natural language processing (nlp)?* https://www.sas.com/en_gb/insights/analytics/what--is--natural--language--processing--nlp.html

Shylesh, S. (2017). A study of software development life cycle process models, 534–541.

Sindhu, I., Daudpota, S. M., Badar, K., Bakhtyar, M., Baber, J., & Nurunnabi, M. (2019). Aspect–based opinion mining on student's feedback for faculty teaching performance evaluation. *IEEE Access*, *7*, 108729–108741.

Suleiman, D., & Awajan, A. A. (2019). Using part of speech tagging for improving word2vec model. *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)*, 1–7.

Sultana, J., Sultana, N., Yadav, K., & AlFayez, F. (2018). Prediction of sentiment analysis on educational data based on deep learning approach. *2018 21st Saudi Computer Society National Computer Conference (NCC)*, 1–5.

Thangaraj, M., & Sivakami, M. (2018). Text classification techniques: A literature review. *Interdisciplinary Journal of Information, Knowledge & Management*, *13*.

Unankard, S., & Nadee, W. (2019). Topic detection for online course feedback using lda. *International Symposium on Emerging Technologies for Education*, 133–142.

Wang, G., Khan, M. S., & Khan, M. K. (2021). Predicting user perceived satisfaction and reuse intentions toward massive open online courses (moocs) in the covid–19 pandemic: An application of the utaut model and quality factors. *International Journal of Research in Business and Social Science*, *10*(2), 1–11.

Wirth, R., & Hipp, J. (2000). Crisp–dm: Towards a standard process model for data mining. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, *1*.

Xia, C., Zhang, C., Yan, X., Chang, Y., & Yu, P. S. (2018). Zero–shot user intent detection via capsule neural networks. *arXiv preprint arXiv:1809.00385*.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, *13*(3), 55–75.

Zhang, Y.-.-F., Wang, X., Kaushik, A. C., Chu, Y., Shan, X., Zhao, M.-.-Z., Xu, Q., & Wei, D.-.-Q. (2020). Spvec: A word2vec–inspired feature representation method for drug–target interaction prediction. *Frontiers in chemistry*, *7*, 895.