ATHABASCA UNIVERSITY

IDENTIFYING STUDENT DIFFICULTY AND FRUSTRATION FROM DISCUSSION FORUM POSTINGS

BY

STEVEN C. HARRIS

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE IN INFORMATION SYSTEMS

SCHOOL OF COMPUTING AND INFORMATION SYSTEMS
FACULTY OF SCIENCE AND TECHNOLOGY

ATHABASCA UNIVERSITY
MAY, 2018

Approval of Thesis

The undersigned certify that they have read the thesis entitled

IDENTIFYING STUDENT DIFFICULTY AND FRUSTRATION FROM DISCUSSION FORUM POSTINGS

Submitted by

Steven Harris

In partial fulfillment of the requirements for the degree of

Master of Science in Information Systems

The thesis examination committee certifies that the thesis and the oral examination is approved

Supervisor:
Dr. Vivekanandan Kumar
Athabasca University

Committee Members:
Dr. Shawn Fraser
Athabasca University

Dr. Kinshuk
University of North Texas

External Examiner:
Dr Cindy Xin
Simon Fraser University

July 3, 2018

Abstract

This work applies natural language processing techniques, like those used in sentiment analysis, to the data generated by students in a digital online learning environment to detect confused or frustrated students and alert instructors so that time-sensitive educational support can be provided.

Utilizing a data set of 9,141 discussion posts collected from an Introduction to Java Programming course, seven types of classifiers were tested, including Support Vector Machine (SVM), Naive Bayes, and Random Forest algorithms; it was determined that the optimum results for the data set was an SVM classifier using a non-linear Gaussian kernel, combined with a custom dictionary and noun phrase POS frequency count for feature vector identification and the determination of a relevance probability.

The resulting application, TutorAlert, produced a promising F1 score of 0.79 and an accuracy of 0.83. Further, agreement values of 88% were achieved during inter-rater reliability testing between the classifier and human judges.

Keywords: Sentiment analysis, e-learning, opinion mining, natural language processing

Acknowledgements

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Vivekanandan Kumar, for his advice, comments, and unwavering support. Without his calm, supportive demeanour, and academic guidance this work would not have been possible. I was truly honoured to share this journey with him.

Thanks also to the Researchers and fellow Graduate Students at Athabasca University School of Computing and Information Systems - Diane Mitchnick, Isabelle Guillot, Rahim Virani, David Boulanger, Jeremie Seanosky, Colin Pinnel, Geeta Paulmani, and Stella Lee. I learned so much from all of you and looking back our times together seemed to go by very quickly. I wish you all the best in your future endeavours and look forward to staying in touch.

Finally, I am extremely grateful for the support of my family, and especially to my wife Janine and my sons Simon and Cole for their patience, understanding and support in creating the space and time to conduct my research.

The research work described in this paper was supported in part though the Alberta Innovates Graduate Student Scholarship program.

Table of Contents

List of Tables

List of Figures

Chapter I - INTRODUCTION

The past decade has seen a marked increase in the establishment of online learning environments by academic institutions, private training organizations, and businesses offering a wide array of education and training options to a broad audience of potential students. Everything from corporate training and certification to compete academic degree programs are now available through digital classrooms at a range of scales and pricing options. This growth, however, which allows for near immediate satisfaction of learning needs, has often come at the cost of a reduced student to instructor interaction than would otherwise been encountered in a traditional classroom. Even where instructor to student ratios have remained constant, the breadth of options and materials that are available to the students fuel an expectation of almost real-time response geared towards student convenience and the necessary wait involved in receiving any direct response learning assistance from an actual human tutor abruptly puts the whole learning process on hold.

Indeed, when students encounter issues in an online learning environment, they typically have a limited number of options. These options may include sending a direct message to the tutor through some form of online messaging or help desk function, or to ask for advice or assistance from fellow students in a shared course discussion group. Delays in either option can lead to increases in overall frustration and student dissatisfaction [1], especially if there is perceived to be too long a delay in receiving a response. This may even be compounded in shared discussion groups where multiple students experiencing the same issue will join the fray, turning questions into a series of complaints before the issue is even formally addressed by the

course instructor. In short, the quicker an instructor can identify and address relevant learning issues or problems in online discussion groups, the better.

In addition, the identification and aggregate reporting of topics and concepts that were consistently associated with student frustration and confusion over multiple sections of the same course material could aid in the determination of weak or problematic course material over time. Thus, tracking this sort of information will offer a unique perspective to aid in the improvement of the overall learning experience of the digital course.

Just as society has seen the growth in digital online learning environments, there has also been a dramatic increase in the overall use of online text-based platforms for public communication or storing public information. These platforms range from social media tools like Twitter and Facebook, to online product review sites, public discussion forums, and other similar digital sources. The drive to better understand and measure these discussions on the part of businesses, policy decision makers, and academics, has fueled considerable growth in research techniques around sentiment analysis, or opinion mining, as well as other big data reporting functions. Indeed, a substantial amount of work has gone into developing the natural language processing tools and related machine learning techniques that attempt to identify the polarity of audience sentiment around relevant entities and their component attributes. Examples of studies in this area include work on determining sentiment in online movie reviews or consumer product reviews, as well as attempts to identify negative opinions in active social streaming tools, such as Twitter [2].

In theory, the same natural language processing techniques used to build classifiers in opinion mining and sentiment analysis may also be leveraged to develop other types of custom classifiers, including an algorithm that can classify student frustration and confusion, and immediately send an alert to the course instructor, notifying them of the issue when appropriate. The instructor would then be better equipped to prioritize their time in developing responses, and the overall experience for the students could be dramatically improved. Further, if the alert system was developed in such a way as to recognize keywords and phrases at the root of the frustration and store those over time, the resulting data might be a valuable set of learning analytics metrics to assist course developers in assessing content, and specifically identifying weak content over time.

Research Question

The overarching research question in this thesis work is to determine whether the analysis and classification of the text-based student interactions in a digital learning environment could be used to detect student frustration or learning difficulties through the application of techniques such as those used in sentiment analysis and other natural language processing applications. Further, to investigate whether an algorithm could be developed that would not only alert instructors to address immediate individual student issues, but to label the data in a meaningful way so that the accumulated classified data would be used to identify content deficiencies over time.

Or, a more applied version of the research question: can data from the students' use of Moodle Classroom Discussion Forum posts be used to alert instructors when students write about their confusion or frustration? And, if so, can those aspects of the coursework that are causing greater difficulty to the students be identified over the long term?

There are a number of good reasons for exploring this avenue of research. For instance, as mentioned above, learning environments continue to grow in popularity globally at a considerable rate, as discussed in greater detail in Chapter 3, and education providers are continuously facing pressure to contend with increased class sizes. Automated systems that work to identify potential problems and assist instructors in prioritizing their responses to student discussions may free up valuable instruction time spent scanning through student posts that do not require assistance - not unlike the efficiencies found in a good email spam filter.

Further, as already been mentioned above, research suggests a relationship between student satisfaction and the speed in which potential issues are addressed [3]., so as post-secondary institutions continue to experience budgetary constraints, an automated system that allows for more efficient use of staff time may be beneficial to the greater organization.

With regards to a unique academic contribution, the detection of confused or frustrated students explored in this thesis is unique and presents some interesting classification challenges due to the nature of the language and phrases used in a normal academic conversation. Posts around learning difficulties do not tend to have the same obviously negative indicators that might be found in movie review or angry Twitter exchanges. In other words, the posts do not necessarily fall into clear positive or negative sentiment labels.

For example, to a human reader a student comment, *"(d)id you read this week's material? The algorithms are interestingly complex*", would not appear to indicate confusion or frustration, while another student post, *"(d)id anyone understand this week's lecture? The algorithms are way too complex for me"* may indicate the need for a response. Even though a human reading it can see the difference, the language used by the later student does not appear overtly negative and it is entirely likely that it may not have been identified as problematic even with a very well-trained sentiment analysis classifier. In that sense, the classification of student difficulty as represented in student discussion posts more closely resembles the detection of email spam than it does positive, negative, or neutral sentiment. This also suggests the use of a coarse binary classifier labelling of *responseYes* or *responseNo*, over a more granular sentiment analysis approach that might break out content into a broader range, or increased number, of emotions or classes. Granular classification algorithms, and other forms machine learning classification like deep learning and neural networks, also require far more training data, than is typically generated in individual online courses, and their domain sensitivity makes effectively training them on individual course subjects quite difficult. This is discussed further in Chapter 3.

The other related consideration in addressing the research question, is the determination of relevance. Ideally, an instructor should only receive an alert highlighting student difficulty or confusion for topics that are directly relevant to the course material. Unrelated discussions, such as complaints about the weather, should be ignored. The goal here is to have the algorithm review each alert candidate generated from the preprocessing process against a custom

dictionary and assign a relevance probability that would then be used to decide whether to forward an alert to the course instructor.

Thesis Overview

This thesis is organized into five chapters. The first chapter serves as the introduction, lays out the overarching problem, and presents the research question which will be explored in the remainder of the document.

Chapter 2 sets out the research background in greater detail by taking a closer look at online learning platforms, and then discusses related previous work around natural language processing and sentiment analysis. It goes on to detail specific text preprocessing and classification techniques that are used in this growing area of machine learning research, and that were considered for this project.

Chapter 3 examines the methodology used in the development of the TutorAlert classier and defines the parameters of the comparative classifier research tests contained in this work, and details some of the technology used in the process. As well, this chapter focuses on some of the challenges that are unique to this work, and how this approach differs from more general sentiment analysis research.

Chapter 4 covers the overall experiment results of the research work. It includes discussion on the accuracy of the different classifiers used in the attempts to identify relevant student difficulties, both in terms of the actual training data, as well as in comparison with inter-rater reliability testing performed with experienced online course instructors.

Finally, Chapter 5 contains the thesis conclusion with a special focus on the finalization of the overall algorithm suggested by the research results. There is also a discussion on proposed future research directions, including a few interesting development opportunities that presented themselves during the experiment, but were out of scope for this current research.

Chapter II - BACKGROUND

Digital learning environments can refer to a broad range of online education platforms used in the delivery and administration of course material and training programs. Indeed, they are becoming a ubiquitous part of the learning experience and are even utilized as supplements in traditional classroom-based learning institutions in addition to completely self-contained online applications.

This chapter starts off by taking an academic look at digital learning environments, as well as their growth and use trends to date. It then moves on to a discussion regarding natural language processing and speaks to some of the current approaches used to build classifiers for similar data, including sentiment analysis and opinion mining techniques. Finally, upon establishing the background, it breaks out the individual techniques that will be utilized in this research and covers each in their own section.

Digital learning environments

Generally, digital learning environments have a set of common features including the provisions of online lessons and supplementary educational material, the management of student assignments and grading, as well as a platform for instructor-to-student and student-to-student communication and collaboration. They are found in a wide range of applications from corporate training, to supplementing a classroom-based course, and the past decade has seen a marked increase in the use of these systems in post-secondary education to support remote, continuing education, or fully online education programs. In fact, an estimated 25%

of higher education students in India, 30% of post-secondary students in the United States, and 40% of distance learning students in Turkey are enrolled in some form of online education managed through a digital learning environment [4]. Growth is especially pronounced in India and China who have both doubled their student base over the same time period [5], and increasingly, online learning completely ignores borders with one-quarter of Australian post-secondary students claiming foreign citizenship [6].

Researchers tend to divide digital learning environments into different types, depending on their scale, feature set, and use case. Two of the most common forms of online learning are the Massive Open Online Course (MOOCs) platforms, such as Udemy, Udacity, Coursera, and Sandford Online; and open university-style courses offered on platforms like Moodle or Blackboard. Though as student class sizes in Open University courses continue to grow, while MOOCs simultaneously target smaller paid classes through the offering of recognized diplomas for students [7], the distinction between the two is becoming less obvious.

With the growth in the provision of online learning systems comes a great deal of student and course-related data that can be used to track the success of both individual students as well as the effectiveness of the overall course material. These data points can include personal information about the students, interactions with course content and assignments, and overall course grades. Figure 2-1 shows an example of a Blackboard course discussion between multiple students in the course discussion forums, where students and instructors can discuss their progress, or identify problems they are experiencing with the learning material or assignments.

Of course, as the use of online learning environments continues to increase, ensuring that instructors can monitor and react to student needs in a timely fashion is a growing concern and research suggests that instructor reaction time may have a significant effect on both student satisfaction and student success rates [1].

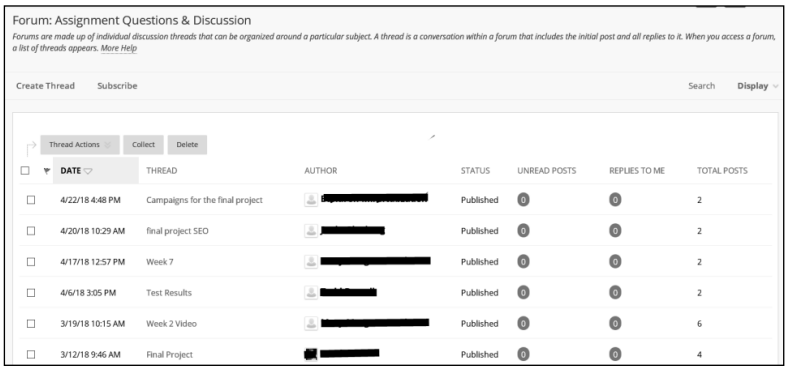

Figure 2-1: Example of a Digital Classroom Discussion Forum on Blackboard.

Current online learning systems already collect a wealth of metrics and logs pertaining to student interaction with the system. Everything from reading, writing, assignment work, tests, and even communication with students and instructors is available for data mining by educators and/or the administrative staff [8]. Further, while there are several automated systems that can assist with grading or managing structured data, such as multiple-choice assignments or quizzes, the data found in the discussion forums of online learning environments poses a challenge because of its diverse, unstructured nature.

For instance, as already mentioned, student discussions around unrelated topics like the weather or current events may show signs of frustration or confusion but have nothing to do

with the course material. Currently, instructors are still forced to go through and read all comments as quickly as possible whether they require a response or not, because there is no way to prioritize messages that indicate the need for immediate assistance against those posts that are in no way time-sensitive in nature.

It also must be noted that while there are sentiment analysis features available for most current online classrooms, including Moodle [9], and there is a body of research dedicated to obtaining sentiment from related discussion groups and forms [10], these efforts, are dedicated to determining student sentiment or positive/negative opinions around various topics after having studied them, or about the overall course as a whole, and do not specifically identify immediate individual student confusion. Nor do they highlight learning topics directly related to, or that may have contributed to immediate confusion to students for later reporting and insights, beyond looking at student grades or physical interaction with the digital platform.

Of course, a classifier-based alert system will never be expected to be the sole comprehensive information source available to instructors and course planners to determine problematic content. A metric showing what topics were most responsible for time-sensitive instructor interventions, however, when combined with the rest of the aggregate physical interaction and academic success data currently collect by the online learning system, could be a unique and useful metric when instructors and course creators are looking for ways to maximize the effectiveness of the learning content over time.

Natural Language Processing

Natural Language Processing (NLP) falls within the fields of artificial intelligence and computational linguistics and is mainly concerned with a range of techniques and processes that allow for the analysis and representation of human language. As a discipline, NLP has been explored since at least the 1950s, but it was in the 1980s when statistically-based machine learning algorithms such as decision trees and Hidden Markov-based Part-Of-Speech research began to be applied to NLP problems, aided by rapidly increasing computer processing power [11], [12] that allowed for much larger data sets to be worked with.

There has been considerable academic and commercial work in NLP over the past decade as the growth in social media, web sites, blogs, product review sites, and other big data sources have created a wealth of largely unstructured information of interest to both businesses and researchers alike. Some of the largest technology companies in the world, including Google and IBM, have invested heavily in projects ranging from web search engine technology, to product review analysis, human-computer interfaces, etc. And while all this attention has certainly encouraged significant strides in technology, it is important to point out that most NLP systems - even the more popular and seemingly advanced ones such as IBM's Watson or Apple's Siri - do not actually have deep understanding of the language they are working with, but instead view much of text analysis as largely a pattern matching exercise. And while there are expert systems that do contain detailed domain knowledge, the work required to develop those custom knowledge centres for general cross-domain use is still very labour and computationally-intensive [9], [13].

Previous research in natural language processing that is directly relevant to this thesis work includes Wang and Cardie's research into developing a dispute identification classifier for Wikipedia Talk discussion pages. Every Wikipedia entry has an associated Wikipedia Talk page where the volunteer editors discuss the content and justify any updates or changes made to the related Wikipedia entry. When disputes get too heated between editors, Wikipedia staff must consider stepping in and engaging in dispute resolution between the editors, prior to any further entry changes. Given the size of Wikipedia, however, it is very difficult to manually detect every disputed entry.

Wang and Cardie tested several NLP classifiers to determine the best method of identifying situations where the content on a Wikipedia page was under dispute based on the comments of the editors - labelling each page with either DISPUTE or NON-DISPUTE. Like student discussion posts, the editor discussion data contains opinionated text or dialogue-specific keywords that are often not specifically negative, which can complicate classification, so they included category detection to their algorithm and used that as an input to train the classifier to adjust for topics where dispute-like discussion was common but may not have indicated an actual problem with the content entry, such as religion or politics. Their research experiments with support vector machine (SVM), logical regression, and isotonic conditional random field (CRM) as candidate classifiers, and on their particular data set, a CRM-based classifier with part-of-speech tagging and category identification performed best with an accuracy of 0.80 and an F1 score 0.78 [14].

Kim and Kang's 2010 research to identify unresolved discussions in student online forums was a rules-based attempt to classify student discussion posts into the categories: questions,

answers, issues, acknowledgements, and corrections. Interestingly, their model would probably be transferrable across domains because many of their rules were based on standard language conventions; for example, the presence of a question mark was used as a strong indicator that the post was a question. Their other categories, however proved more difficult to distinguish, so their highest performing SVM classifier only reached an accuracy of 0.65 in recognizing issues, and 0.57 in correctly labelling responses and acknowledgements [15]. Their work does exemplify the value of at least some rules-based functionality, though, as they were able to detect student questions with 0.95 accuracy and identify answers to specific questions with 0.87 accuracy.

In 2013 a Research Team that included one of the foremost writers on opinion mining, Bing Liu from the Department of Computer Science at the University of Illinois at Chicago, looked at the problem of determining intent within online discussion group posts. Specifically, they were developing an algorithm to identify explicit expressions of purchasing intentions with in the content of a discussion post. Through this work they developed the Co-Class algorithm, which combines the Information Gain (IG) method of feature selection, using entropy to differentiate the categories through the presence or absence of feature labels, with two naïve Bayes (NB) classifiers (hence the term Co-Class), each one trained in a different domain and run repeatedly until they stabilize, and the strongest indicator is selected. They were able to demonstrate this method of category identification paired with NB classifiers produced more accurate cross domain classifiers when looking at purchase intent across four independent domain-specific discussion forums [16] than single classifiers. While the research in this thesis focuses on a single domain with topic categorization, and therefore is more directly relevant

to their initial individual classifier development, prior to pairing them, Co-Class is an interesting approach that is also discussed with future research directions in Chapter 5.

Sentiment Analysis

Sentiment analysis, or opinion mining research, is a branch of NLP that, along with increased academic focus, has seen a significant amount of interest from business and political groups over the past decade, fuelled by the growth of readily-available data analysis tools and the wealth of textual data provided by social media, e-commerce, and digital marketing platforms. In the business world companies and organizations are starting to use these techniques to drive product, marketing or public relation decisions, while political and public organizations are using it to help inform policy decisions. It has even been researched as a method of predicting financial markets or to identify misleading online product reviews [17].

Sentiment Analysis is included here because the techniques relevant to determining sentiment polarity are very closely related to the processes used in this research work, so an understanding of relevant work in this area will be beneficial. And while sentiment analysis may be a relatively straight forward classification problem for direct opinions on a specific entity, the process can quickly become complicated when dealing with more complex entries such as comparative opinions (e.g.: *Samsung phones are better than LG phones*), or when identifying individual attributes of an entity (e.g.: *this phone has a great camera but a terrible battery life*) [18].

There are a number of decisions that must be considered when setting out to do a sentiment analysis problem. One of the first is whether to engage in binary or multi-class classification. Binary classification looks at sentiment as either being positive or negative, while multiclass can include a neutral class, or may even attempt to label corpuses across a spectrum of extremes. Though it is somewhat dependent on the target content being labelled, sentiment analysis algorithms are typically much more likely to obtain a higher degree of accuracy using binary classification, because of the substantially increased sensitivity and training time necessary to develop a multi-class model [19].

Another choice to make when setting out to engage in sentiment analysis is the granularity at which the classification will take place. For instance, sentiment labels can be applied at the document level, the sentence level, or even the attribute level, with increases in the complexity accompanying each move towards great granularity. A professional movie review might be an acceptable target for a document-level sentiment analysis if only the top-level aggregate sentiment is of interest. If, however, understanding the Reviewer's opinions on the Director, the Actors, etc., a sentence-level or even attribute-level classifier may be more useful.

The level of chosen granularity is also heavily dependent on the data being evaluated. Twitter tweets can be a good candidate for sentiment-level sentiment analysis because they are rarely longer than a sentence, but can be challenging to extract individual entities from, while on the other hand product reviews might be easer to work with at a more granular level because the entities are generally prominent easier to extract [20].

Once the number of classes and the level of granularity have been chosen, the next decision is that of the specific technique to employ in labelling the data. There is a wide variety of options at this point, though they can be divided between Lexicon and Machine Learning-based approaches, as shown in Figure 2-2.

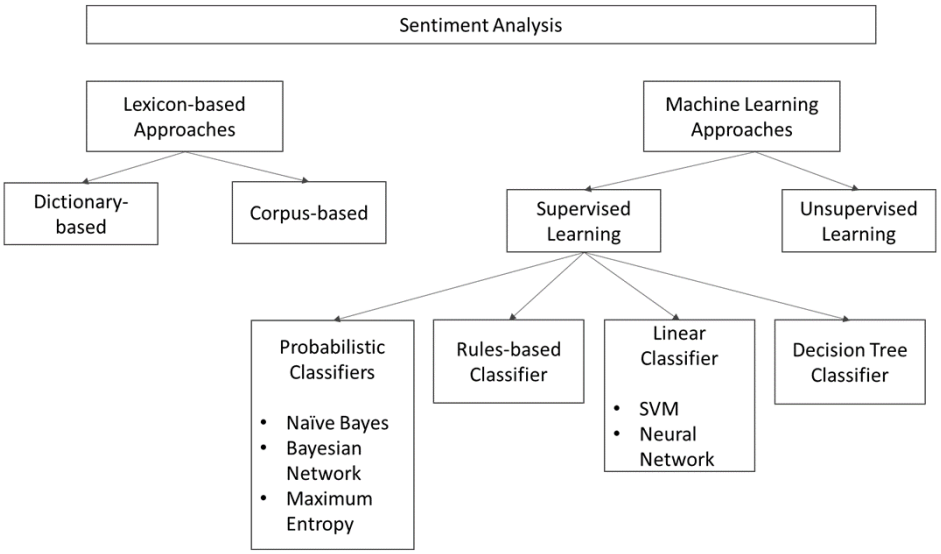

Figure 2-2: Approaches to Sentiment Analysis [20]

Lexicon-based approaches utilize dictionaries and predefined opinion word lists and lexicons to assign positive or negative sentiment. The text is converted into tokens, or tokenized, and then each token is matched to the dictionary and, if matched, is added to the total score [21]. The ultimate classification of the document depends on the total score it ultimately achieves. Lexicon-based approaches yield quite accurate results, but they do not scale well if their tagged corpus or dictionary is growing to accommodate new situations [2]. The research in this thesis uses lexicon-based term frequency-inverse document frequency (tf-idf) to provide a relevance

score to the discussion posts it processes, checking each post against the custom dictionary created by pulling keywords out of learning material related to the course. At its base level, tf-idf is just counting the keyword that appear in the custom dictionary and scoring them by simply counting how many appear in the target post. This will be discussed in greater detail later in the chapter.

Machine learning approaches to sentiment analysis, on the other hand, are subdivided into supervised and unsupervised learning. Supervised learning generally takes the path of data collection, preprocessing the text for the classifiers to better work with the data, training the classifier on a subset of the data, the actual classification task, and then plotting results. The focus on the creation of the TutorAlert classifier will utilize a mix of supervised learning algorithms that have shown promise with similar application in the past. These classifiers include naïve Bayes, Maximum Entropy, SVM, J48 decision tree, and others that we will detail in upcoming sections.

Figure 2-3: The Standard Steps for Supervised Machine Learning Classification

There are even hybrid approaches, such as Yang et al.'s model that combines dictionary and classification approaches in an attempt to leverage the strengths of both, which have seen promising results – though as with most sentiment analysis research, it is heavily dependent on the makeup and character of the data [21].

Stemming, Lemmatization, Part-Of-Speech, and Tokenization

As stated above, before engaging in an NLP project such as sentiment analysis or language classification, there are some pre-processing options which normally must be considered and tested. This is the data preparation that takes the raw text data and prepares it for the classifier [22].

Stemming or lemmatization can be used to normalize words and reduce similar words to a common base form, grouping them together and reducing the overall dimensionality of the data. Part-of-speech (POS) tagging can add an additional level of context to text data by identifying nouns, verbs, adjectives, etc., based on both the word's definition and relationship with the adjacent related words in a phrase or sentence, which will be discussed in the next section.

While the goals of stemming and lemmatization are similar, their approach is quite different, as is their effects on data. Stemming is a heuristic process that reduces similar words to a single base word, which itself may or may not be an actual word, by chopping off the end. So, the words *study*, *studying*, *studies*, and *studious* might all reduce to the stem *studi.* Stemming algorithms have been used extensively in information retrieval and search engines

due to their comparative speed and ability to identify potentially synonymous material [23]. As well, most stemming algorithms can be tuned to exclude certain terms that should not be altered and allow for custom rules that specifically identify words with considerably different roots, like “skis” and “sky”.

Lemmatization, on the other hand, groups various forms of a word together into its single lemma, or base form. This is a slower, more resource demanding process that requires the use of an external dictionary, trading off the speed of stemming for improved accuracy. For example, the words “car”, “care”, and “caring” might all be reduced to the word “car” in stemming, lemmatization would likely recognise the difference, provided the words existed in the dictionary being used. This reliance on the dictionary can also cause challenges with slang, abbreviations, and non-standard terminology [24].

There is the danger, of course, that employing too aggressive a lemmatization or stemming tool will cause too many words to be shortened and may affect accuracy and meaning; while using a solution that is too liberal can overly spread out the weight of results that should be combined into a single meaning.

The use of all these preparation techniques is by no means required to a sentiment analysis task, and there have been plenty of academic studies that do not employ more than one or two of these natural language processing techniques. They are, however, considered to be an effective way to reduce the level of dimensionality and thereby improve the overall training of the algorithm. In fact, during the research and testing phase of TutorAlert development, these pre-processing techniques had a marked effect on the end results, which will be

covered in more depth in Chapter 4, in the section regarding the preparation of the data for testing.

Part of Speech (POS) Tagging

Part of speech (POS) tagging is a natural language processing technique which assigns a tag to each word in a piece of text, classifying it to a specific morphological type such as verb, noun, etc. POS taggers can be useful for extracting subject keywords or features from a text, as these are normally nouns or noun phrases [24].

There are different types of POS taggers, generally divided between rules-based solutions and stochastic-based solutions that assign tags based on probabilities. Either of these can be implemented as supervised or unsupervised, though unsupervised taggers tend to be implemented as a part of a greater NLP system that provides learning feedback, while supervised taggers tend to be trained on a previously-tagged corpus.

The most commonly used taggers for NLP research tend to be the Hidden Markov Model (HMM) TreeTagger, the Maximum Entropy (ME or MaxEnt)-based Stanford POS tagger, and Python's NLTK toolkit, which includes both HMM, Maximum Entropy taggers; as well as a rules-based Transformation taggers. Comparative academic research on POS taggers consistently recommends the use of either the TreeTagger or Stanford POS for English-based research work [25] [26]. For the research performed as part of this thesis, therefore, the Stanford POS was chosen, due to it's compatibility with other test software, and the consistently solid performance at tagging nouns and verbs, as demonstrated by Tian & Lo

[25], where they demonstrated an accuracy of 92.7% on noun identification, and Abebe & Tonella [26] who saw 81.8% accuracy improvement over unprocessed natural sentences.

Common Natural Language Processing Classifiers

Once deciding on the preparation treatment and feature selection of the data, the next major step is to identify the type of classifier that will be the most successful on the data set in question. Here, there are several choices, that have been well-tested in applications that suggest they may also be useful in determining student frustration or confusion. The most common machine learning algorithms used in sentiment analysis and related research are either naive Bayes, SVM, Decision Tree, or Maximum Entropy learning algorithms. In fact, SVM and naive Bayes have long been cornerstones in email spam research, which is arguably a more mature, well-tested area in natural language processing than sentiment analysis, so they are promising places to start [27], [28]. For the research here, and generally when looking at new natural language processing applications, it is important to test against each of these because different classifiers tend work better with some data sets than others.

Naïve Bayes and Multinomial Naïve Bayes

Naive Bayes is a family of algorithms based on Bayes' Theorem that has been popular in natural language processing applications including sentiment analysis, largely because of the simplicity in both the training and classifying stages. It is a probabilistic classifier that identifies patterns by assigning class labels to problem instances, represented as vectors of feature values, where that value is assumed to be independent of the value of any other feature.

For example, if the classifier is told that a fruit that is round, red, and about 3 inches in diameter, then each of these three features are treated independent of one another, but whenever they exist together they contribute to the probability that the fruit will be classified as an apple [29].

Bayes theorem provides a way of calculating posterior probability P(c|x), the posterior probability of class c given predictor x, from the prior probability of the class P(c) , the prior probability of the predictor P(x), and the probability of the predictor given the class P(x|c) as shown in the equation seen in Figure 2-4.

$$P(c|x)=\frac{P(x|c)P(c)}{P(x)}$$

Figure 2-4: Bayes theorem

When used in NLP, the per-processed data along with extracted features are provided as input to the classifier and, once the training is complete, it can apply labels to the set based on the learned pattern – the polarity of sentiment analysis, for example.

This research also included testing on a Multinomial Naïve Bayes classifier – the main difference being that with Naïve Bayes there is no assumption made on the nature of the distribution of the data, while Multinomial Naïve Bayes assumes a multinomial distribution. This change has proven effective in a number of research studies involving NLP, depending on the nature of the data, and was therefore worth testing here [21].

Support Vector Machines

Support Vector Machines (SVM) are a family of supervised machine learning algorithms that are used in both regressions and classification problems, and they are often seen in academic research around sentiment analysis and similar language classification problems. Using this algorithm, every data item is plotted as a point in n-dimensional space, with the value of each feature being the value of the plotted coordinate. From there, classification is performed by determining the hyper-plane that based differentiates the two classes, as seen in Figure 2-5 [30].

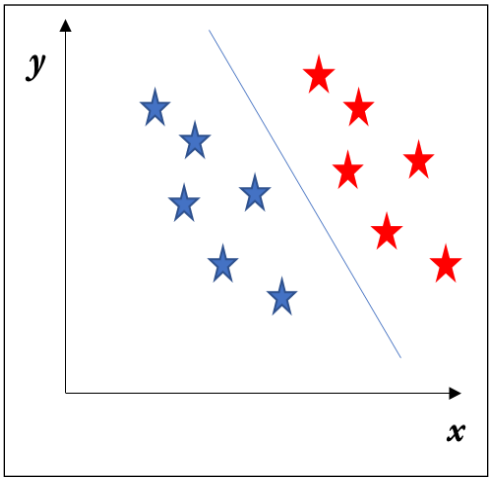

Figure 2-5: Visualization of a Support Vector Machine

Obviously, the best separation is achieved where the hyperplane has a good separation between the data points because, while most SVM implementations can handle outliers well, too many of them can introduce noise and negatively affect the classifier's accuracy.

An additional consideration when implementing SVM is the kernel function to use. Kernel functions, or kernel tricks, are methods used by SVM to convert data that is not separate in one dimension and transform it to a higher dimension that is separable by transforming the data based upon the labels or outputs defined in the problem [30].

Kernels are generally separated into linear and Gaussian, or RBF kernels. Linear kernels are fast but perform best when the data is well separated. Gaussian or RBF kernels are much more computationally heavy, but tend to have a better predictive ability, especially where the plotted data is more scattered [31]. For the research contained in this thesis, both a linear and RBF kernel have been used for comparison purposes.

Decision Tree

A decision tree appears like a flowchart where each individual node represents a test on an attribute and each branch represents an outcome of the test, and each leaf, a class. Used to generate rules for the prediction, and also known as C4.5, decision tree classifiers are a good classifier choice for situations that require dealing with a significant amount of data. The words and phrases from a training corpus, for example, become represented in the leaf nodes of the decision tree [30].

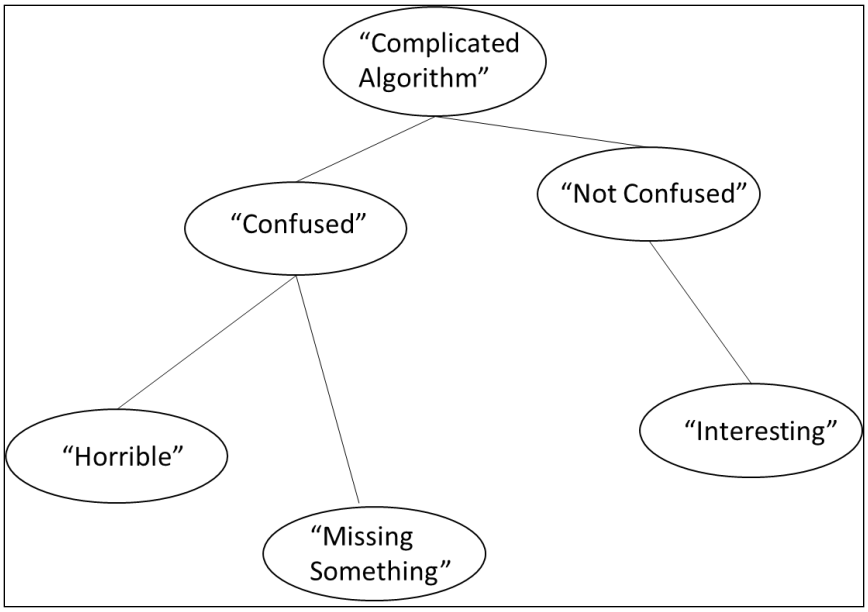

Figure 2-6: Example Decision Tree for language classification.

As an example, in Figure 2-6 above, words and bigrams are labelled with their respective classifications, indicating whether a student in question is confused about subject. So in this model, the phrase "I find complicated algorithms more interesting" would be labelled as "Not Confused", where the phrase, "I find complicated algorithms horrible, I am missing something", would be labelled as "Confused".

Maximum Entropy

Maximum Entropy is a probability-based classification technique, which has been commonly used in natural language processing applications. The main concept behind ME classifiers is that of feature-based classification, where the features are used to find a distribution over the different classes, or the probability of any data point belonging to a particular class, using

logistical regression. Unlike Naïve Bayes, ME makes no assumptions about independence of features, so it can more easily handle bigrams, trigrams, and even phases without worrying about feature overlap.

So, for example, for every discussion post (w) to be categorized in a class (c), an ME classifier would define a joint feature $f(c,w) = N$, where N is the number of times w occurs in a training document already labelled as class (c) [32]. In practice sentiment analysis classifiers based on ME have been used very effectively in some very prominent applications – the Stanford Sentiment Analysis Engine being one popular example. That said, as with the Stanford Sentiment Analysis Engine, they tend to be extremely tied to the domain they are trained in because their classifications are based on weights between correlated features, and they are more sensitive to poor or unstructured training data than other type of classifier [33].

Feature Extraction and Selection

With the main machine learning classifiers covered it would now be worthwhile to turn attention to the topic of feature selection and extraction. Typically, natural language processing, including work with sentiment analysis, is more useful when the object or focus of the opinion can be identified. This is a challenging problem, however, especially when considering automated feature extraction from unstructured data such as discussion forum posts or open-ended product reviews. Indeed, many current sentiment analysis classifiers in use in business today, such as Sysomos, Crimson Hexagon, and Netbase, operate largely on a higher document-or sentence-level classification, because of the difficulty in identifying the features upon which the individual sentiments are based [34]. So, for example, if a review for

a specific digital camera contained a comment like: "the Poloron 5000's lens was fine, but the battery life was very poor," and the classifier was working on sentence-level sentiment analysis, there is a danger that the lens, battery life, and the whole Poloron 5000 itself would be assigned negative sentiment, even if that wasn't technically the intent of the comment.

Feature extraction and selection are very active areas of natural language processing research, and there have been several techniques explored in the automatic identification and selection of domain features. One common method of feature extraction is to attempt to handle the problem in pre-processing using custom parsers which use Hidden Markov models to identify features during POS tagging – the GENIA tagger which is specifically designed for biomedical text tagging is one example of this approach [35]. A separate, though related approach is the use of a custom dictionaries as part of the lemmatization process. Lemmatization replaces words with their base, or lemma, using a dictionary database which can be customized for the specific domain in which the classifier will operate. The Bio Lemmatizer for biomedical texts [36], or Concept Net, which is specialized for technology product reviews [37], are two such examples built on the common, general purpose Word Net Lemmatizer [38]. This was a direction initially considered for the research in this thesis, but the manual effort required in creating a custom dictionary database for every possible educational application would dramatically affect the ultimate portability of the model, so less manually intensive efforts were used instead.

Once the potential features have been identified, or extracted, feature selection attempts to identify the most relevant attributes to increase classifier and model accuracy. Current approaches to feature selection can be divided into four main types – heuristic, statistical,

clustering-based, and hybrid [39]. Heuristic models operate largely on POS data and attempt to select important features based on their grammatical position in the content, and have quite high accuracy, though this is very dependent on the accuracy of the POS tagging [24]. Clustering-based feature selection uses machine learning algorithms to group similar features into groups, the benefit being that there is little required in the way of configuration – though clustering tends to favour large features, and minor features can be difficult to identify and can be missed [39].

Statistical feature selection techniques have employed several methods, including information gain, and decision tree models; and while these approaches have seen some success [38], [27], they can also be very computationally expensive, which has encouraged many researchers to explore hybrid models that combine the strengths of more than one approach [24].

Building on these prior methods, for this thesis research, a custom dictionary is created of keywords and phrases taken directly from relevant course material and compared against a stop list to remove common or irrelevant terms. The resulting list is tokenized and provides a weighting factor for the td-ift that can be by the classifier to help determine whether a comment is applicable to the course material simply through the presence of applicable keywords. So, when a classified post appears as a candidate for an alert to the instructor, the topic and relevance probability determine whether an alert is forwarded.

Further, instead of integrating feature selection and clustering within the classification algorithm, TutorAlert treats that as an independent data mining issue and allows for reporting

on topic identified within the alerts, as well as clustering and trends over time, without further complicating the classification process.

Chapter III - METHODOLOGY

The intention of the TutorAlert project is to develop a natural language processing classifier that can identify student confusion or frustration in the discussion forums of an online learning environment and alert the instructors to the situation as quickly as possible. A secondary goal is the identification of course content that is consistently causing confusion and frustration so that it can be improved over time. Previous efforts in this area, have largely focused on the learner's level of physical interaction with the learning environment [41], or have performed standard sentiment analysis to gauge overall student satisfaction with the tools or course material [42], rather than using natural language processing to identify students who are having difficulty with specific topics based on their posts to the discussion forums.

Classifying discussion posts of an online learning environment raises a few interesting challenges, though. First, there are the standard difficulties of handling free-form text and the imprecise nature of the English language. Second, there is the necessity to identify the subjects of the text and attempt to match them with some learning concept or course feature to better inform the instructor of the nature of the difficulty. Third the use of lexicon-based solutions becomes complicated because most general sentiment dictionaries or word lists that identify language as positive or negative do not necessarily apply. For instance, it is very possible that a student could request assistance in a manner that was very positive in language and sentiment but may still require an equally prompt response from the instructor.

Another consideration inherent to classifying posts from a digital classroom discussion forum is the breadth of relevant subjects that a solution will need to account for. As a rule, sentiment analysis classifiers and processes do not transfer readily from one domain to the other, and

similarly the language used to identify problems with an English course will likely be markedly different than that of a Computer Science course. Any existing solution considering this area would need to tackle this issue using something like Pan et al.'s Spectral Feature Alignment algorithm to create a baseline relationship between the two domains [43], or use develop the aforementioned custom word list as part of an entropy based classifier as suggested by Deshmukh and Tripathy [44].

It is important to note here, too, that even though this thesis work covers only the single domain of an Introductory Java Programming Course, and the ability to move across domains is not an absolute requirement at this stage of research, for the purposes of ongoing flexibility and usefulness of the project, the ability to move to other subjects with as little effort as possible, remains a desirable design goal. This will also factor into the final decisions regarding data preprocessing in this Chapter, and algorithm decisions discussed later.

Development Platform and Tools

The initial development and testing of TutorAlert has been done on a blend of Java and Python 3. As has already been mentioned, the Java-based work centered on the Stanford University NLP Parser and POS Tagger [25] and the University of Waikato's Waikato Environment for Knowledge Analysis (Weka) [48]. These tools were chosen largely due to their prevalence in academic NLP research, and the fact that they contain main optimizations and parameters that would be extremely time-consuming and complex to put together from scratch. Additionally,

the fact that these software tools are open source, and licensed under the various versions of the GNU General Public Licence as also helpful in integrating them in the TutorAlert workflow.

For the tf-idf process to help identify relevance, the Python 3-based scikit was utilized, because Weka does not yet allow for custom dictionaries to be used in their version of tf-idf. Scikit is licenced under the BSD license [49].

Additionally, the Python 3-based Natural Language Tool Kit (NLTK), which is distributed under the Apache 3.0 licence, and was used for testing various relevance weights, stemming, some POS work, and confirming Weka classifier results [47].

The web components of TutorAlert are contained within Google' cloud-based app engine, and the data is stored on the Google App Engine Datastore.

Data Preparation

The raw data for this thesis study consisted of 9,141 individual forum posts and messages taken over 6 sections of the Comp 268 course and collected in the Moodle online education environment logs. Once processed, with any blank lines removed, the data set was 71,175 lines long. Initial parsing and pre-processing included anonymizing the data to remove student

numbers, names, and identifiers, as required by the University Ethics Approval found in Appendix A.

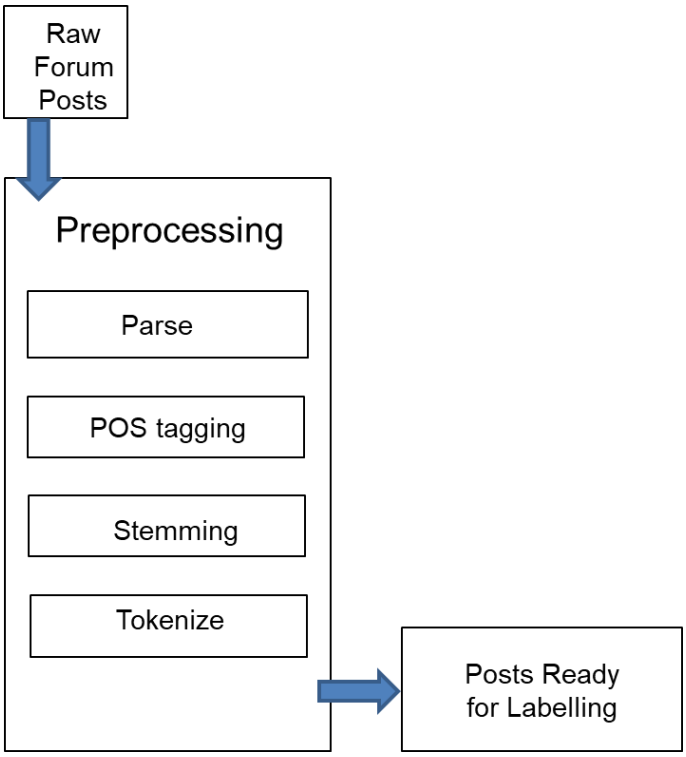

Figure 3-1 TutorAlert logical flow, highlighting preprocessing tasks.

Posts by the instructors were also manually identified and removed from the raw data, but student posts that elicited a response from an instructor were labeled as such, for initial training purposes. Additional training came from manually labelling data that would be used in the training sections. In total, there were 1,786 posts that were seen to have required time-sensitive instructor intervention, with the remainder being deemed as not time-sensitive in nature. For

training purposes, 1000 posts requiring assistance, and 1000 posts that did not require assistance were included in the training set, in an attempt to ensure a well-balanced classifier.

The raw data set entering preprocessing is demonstrated in Figure 3-2 below, with an anonymized Student ID number, the time and date of the original post, the post text, and whether there was an instructor response to the post.

The Student IDs, while anonymous, are unique to the individual student, thus allowing for the identification of students having multiple posts about different topics contained in the course material. The only data point not shown is the unique database ID assigned to each individual post.

STUDENT_ID:	U126583
DATESTAMP:	Sunday, 5 May 2013, 02:57 PM MDT
TEXT:	Hello everyone I have a question regarding question 25 in assignment 1. I think I am over thinking it but I wanted ask. I can only really think of two lines of code. Do the math calculation. Format number and display in console. Is it really that simple? Cheers
ALERT	responseYes

Figure 3-2: Sample training post after initial preprocessing

At this point the data is ready to enter the preprocessing functions shown in Figure 3-1. The first step of this process is to apply the Part of Speech tagging, and then to append the POS tags to each word in the post, as seen in Table 3-1 below. This step must happen prior to

stemming so that the POS tagger has the best chance of properly understanding and tagging each word properly. As discussed in Chapter 3, the main POS tagging solution that was decided upon for TutorAlert was the Stanford POS tagger – both because of its recommendations in the academic references cited previously, and the fact that the Stanford POS tagger works with Weka, the machine learning research tool, available from the University of Waikato, New Zealand, which contains several useful preprocessing and attribute testing functions [45]

Part of Speech tags have largely standardized around the Penn Treebank tag set and is also used by the Stanford tools as shown in Table 1.

Table 3-1: POS Tag definitions.

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun

Tag	Description
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Whdeterminer
WP	Whpronoun
WP$	Possessive whpronoun
WRB	Whadverb

Looking at Figure 3-1, it is easy to pick out the nouns that will be an important input to both the classifier and the relevance probabilities. Keywords such as “question”, “assignment”, “code”, “calculate”, are clear and it is probably little surprise that this was indeed a post that was actually responded to by the instructor in the training set.

STUDENT_ID:	U126583
DATESTAMP:	Sunday, 5 May 2013, 02:57 PM MDT
TEXT:	Hello/UH everyone/VB I/PRP have/VBP a/DT question/NN regarding/VBG question/NN 25/CD in/IN assignment/NN 1/CD ./. I/PRP think/VBP I/PRP am/VBP over/RB thinking/VBG it/PRP but/CC I/PRP wanted/VBD ask/VB ./. I/PRP can/MD only/RB really/RB think/VB of/IN two/CD lines/NNS of/IN code/NN ./. Do/VB the/DT math/NN calculation/NN ./. Format/VB number/NN and/CC display/NN in/IN console/NNP ./. Is/VBZ it/PRP really/RB that/DT simple/NN ?/. Cheers/NNP
ALERT	responseYes

Figure 3-3: Sample training post after POS tagging

After POS tags have been applied, it is now time to apply stemming to the data as seen in Figure 3-4 below. Both Weka and NLTK have a number of stemming options, though the Porters Stemmer, implemented as the Snowball Stemmer in Weka, is considered by a number of researchers to produce some of the best output compared to other stemmers, and even though it is more time consuming, the trade off is a significantly lower error rate in English, so it is very suitable for the purpose of this research [50].

STUDENT_ID:	U126583
DATESTAMP:	Sunday, 5 May 2013, 02:57 PM MDT
TEXT:	Hello everyon I have a question regard question 25 in assign 1 I think I am over think it but I want ask I can onli realli think of two line of code Do the math calcul Format number and displai in consol Is it realli that simpl Cheer
ALERT	responseYes

Figure 3-4: Sample training post after Porters Stemmer applied.

The stemmer has the ability to ignore terms contained in the custom dictionary, and that was tested in the initial work to determine if accuracy was increased by keeping keywords and n-grams that are most relevant, untouched. The difference between including and excluding the custom dictionary was not significant, though, and in fact there were a number of terms that should be included as keywords that were missed out by not being stemmed – coding and code, for example, so stemming was performed on the custom dictionary.

Finally, the POS tagging was performed, as can be seen in Figure 3-5 below. As a final step, the data is tokenized in preparation for the classifiers. Also seen in Figure 3-5 below, a number of the nouns are again highlighted to show the effect of the combination of the stemming and POS tagging.

STUDENT_ID:	U126583
DATESTAMP:	Sunday, 5 May 2013, 02:57 PM MDT
TEXT:	Hello/UH everyon/VB I/PRP have/VBP a/DT question/NN regard/VBG question/NN 25/CD in/IN assign/NN 1/CD ./. I/PRP think/VBP I/PRP am/VBP over/RB think/VBG it/PRP but/CC I/PRP want/VBD ask/VB ./. I/PRP can/MD only/RV realli/RB think/VB of/IN two/CD line/NN of/IN code/NN ./. Do/VB the/DT math/NN calcul/NN ./. Format/VB number/NN and/CC display/NN in/IN consol/NNP ./. Is/VBZ it/PRP realli/RB that/DT simpl/NN ?/. Cheer/NNP
ALERT	
	responseYes

Figure 3-5: Sample training post with both POS tagging and stemming applied, highlighting nouns.

Category List Creation

To classify the posts as being relevant to a specific topic, a custom list of categories is required as an input for labelling topics and determining overall relevance of the post to the course topics. This proved to be one of the more challenging aspects of the thesis work, and a prime area for further research, as discussed further in Chapter 5.

There was a great deal of experimenting with various methods of developing topic-matter ontologies from the course material and testing other methods of automatically generating hierarchical terminology lists, but these approaches were rejected for two main reasons. First, for TutorAlert it would dramatically reduce the chance of the algorithm being transportable to other subjects beyond our study data. Further, the development of custom ontologies or structured labels would markedly increase the technical complexity and sheer amount of work that would be involved in moving the tool to another subject.

Second, the focus of this research is on the actual classifier, and there is a risk that developing an intricate post labelling system might compound errors, or at least take the project significantly beyond the its current scope.

Instead, the list of categories was developed by using the Full Table of Contents, Programming Exercise Descriptions, and Glossary from the free online textbook, "Introduction to Programming Using Java, Seventh Edition", by David J. Eck [46]. Then, using Python's Natural Language Tool Kit's (NLTK) [47] stop word list, we removed any common conversational words such as "I", "your", "if", etc., so that only those words relevant to the subject matter remained.

Table 3-2: Summary of top custom category list.

Frequency and top words :

Word	Occurrences	Frequency	Rank
program	131	2.9%	1
class	87	1.9%	2
object	71	1.6%	3
ar	70	1.5%	4
java	69	1.5%	4
type	65	1.4%	5
us	60	1.3%	6
method	55	1.2%	7
variabl	54	1.2%	7
data	53	1.2%	7

The categorization of the posts happens after they have been preprocessed, with stemming POS tags already in place, the custom category list should also be in the same format to simply comparison, so stemming and POS tagging were also applied to the category list, and duplicates are removed. Because the category list is not in sentence form, however, and only a high-level categorizing of post as a measure of relevance is required, it is possible to generate the categories simply by applying the noun and verb POS tags to each of the word list.

```
1  program/NN,program/NNS,program/NNP,program/NNPS,program/VB,program/VBD,program/VBG
   ,program/VBN,program/VBP,program/VBZ,class/NN,class/NNS,classm/NNP,class/NNPS,clas
   s/VB,class/VBD,class/VBG,class/VBN,class/VBP,class/VBZ,object/NN,object/NNS,object
   /NNP,object/NNPS,object/VB,object/VBD,object/VBG,object/VBN,object/VBP,object/VBZ,
```

Figure 3-6: Partial category list with stemming and POS tagging applied.

The completed category list was then used as an input to the tf-idf process.

Experimental Setup

Once the preprocessing is complete, the test to determine the best classifier for the discussion post data can proceed. As previously mentioned, the Weka version of the classifiers will be used for the experiment itself, both for the ease of configuration, and because Weak allows for saving trained classifiers for later use, rather than forcing a retraining of the model every time, or having to create a custom persistent environment for training and testing.

Table 3-3: Statistics of the dataset.

Dataset	Type	Response Required	No Response Required	Total Posts
Discussion Forum Posts	Train	1000	1000	2000
	Test	786	6355	7141

As stated above, our training set consists of 1000 discussion posts requiring a response, and 1000 discussion posts that are not labelled as requiring a time-sensitive response from an instructor, for a total of 2000 labelled posts. Similarly, the remaining data set aside to test the classifiers is split between those requiring time-sensitive responses, and those that do not.

Also as stated above, the training data is an even split between those posts requiring responses. This is known as over sampling, and is undertaken so that avoid classifier bias because of a lack of data on either side of the class [51].

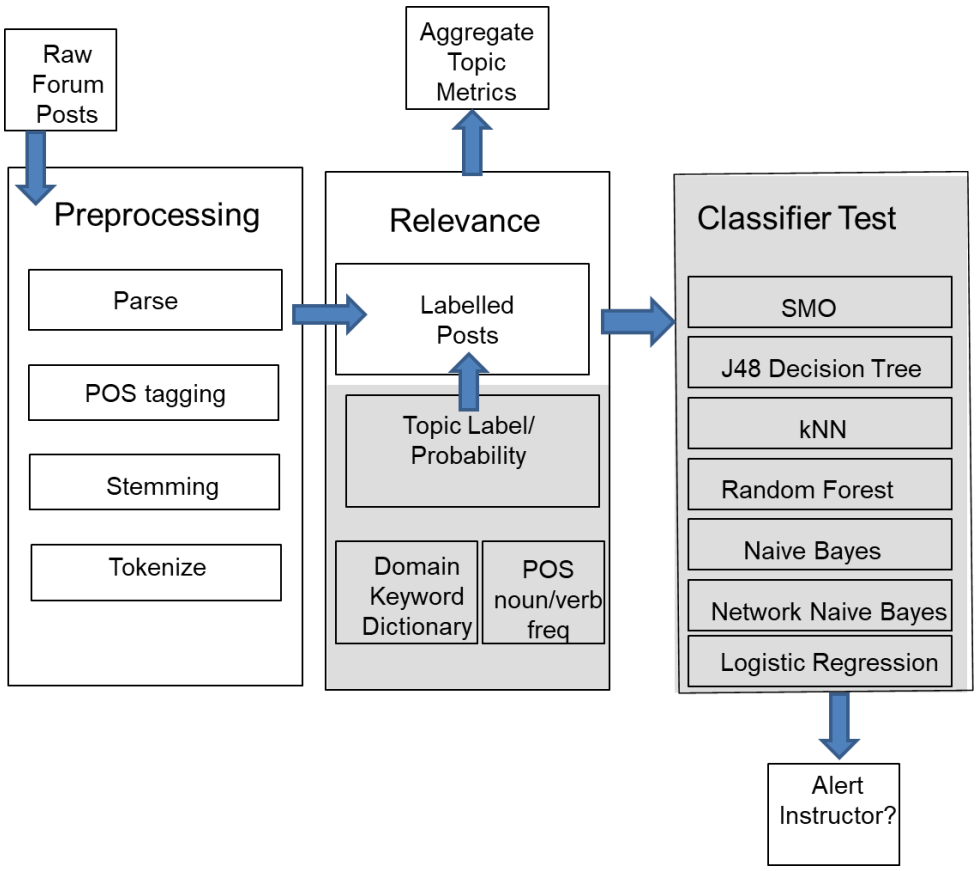

Figure 3-7: Category list with stemming and POS tagging applied.

The final stage of the experiment is to train each of the classifiers with the training data, and then expose them to the test data in order to determine their accuracy. The results of this phase will be covered in the next chapter, Chapter 4.

Chapter IV - RESULTS

A quick scan through current academic research and it soon become apparent that there have been a large number of different machine learning algorithms that have been tested with different data sets, and within different domains, with wildly varying results. Since there was not a great deal of academic research found that was specific to testing different classifiers on online education discussion posts, it is necessary to cast a wide net in determining which would be the best choice. As discussed in Chapter 3, the 9,141 discussion posts were split into training and test data for the purposes of testing a number of different classifiers build within the Weka environment.

Seven different classifiers – an implementation of a Support Vector Machine algorithm called Sequential minimal optimization (SMO), C4.5/J48 decision tree, k-nearest neighbor, Random Forest, Naive Bayes, Network Naive Bayes, and Logistic Regression – were created and trained in order to run the experiment.

For each classifier we computed the average accuracy, max accuracy, and F-score, or F-measure which is the harmonic mean of recall and precision, and provides an indication of classifier accuracy on a scale between 0 at the low end, and 1 at the high end [52].

Looking at the results in Table 4-1, the SVM algorithm obtained both the highest accuracy and F-score measures, followed by Random Forrest and Multinomial Naive Bayes.

Table 4-1: Overall Classifier Results

Classifier	***Avg Accuracy***	***Max Accuracy***	***Avg F***
SMO	81.86%	84.82%	0.820
J48 Decision Tree	73.25%	77.60%	0.728
Lazy-IBK (kNN)	70.88%	75.52%	0.708
Random Forest	79.79%	82.75%	0.798
Naive Bayes	69.82%	71.33%	0.688
Multinomial Naive Bayes	78.90%	80.14%	0.783
Logistic Regression	74.64%	76.62%	0.746

While a high-level NLP classifier that can alert instructors to potential student problems at a post-level is certainly useful, one of our research goals was to also provide feedback to instructors around course material that may be problematic, or at least consistently causing confusion to students on some level. To these ends, we have developed a report that identifies course-related keywords from the syllabus and runs a similar analysis just on posts containing those concepts.

Inter-Rater Reliability Testing

Working with our classifier and achieving a satisfactory level of accuracy is certainly promising, but to truly test the utility of a natural language processing tool like TutorAlert, the gold standard to test the resulting algorithm against actual instructors through the use of inter-rater reliability testing [53].

To accomplish this test, 200 of the student discussion posts from the data were loaded into a question bank where three independent on-line course college instructors were asked to determine whether they, on seeing a similar post, would intervene in an urgent, time-sensitive fashion to the post being displayed.

Table 4-2: Inter-rater Reliability Test Results

Instructors	***% Agree***	***Cohen's Kappa***	***N (200) disagrees***
Instructor 1	91%	0.79	18
Instructor 2	74%	0.58	52
Instructor 3	81%	0.65	38

As one can see in Table 4-2, the results are very positive, ranging from 74% agreement to 91% agreement. It is important to note, though, that in the two cases with the lowest agreement, the instructors from the inter-rater reliability test identified a greater number of posts that they felt required a time-sensitive response, than the classifier recognized. And while this may be ascribed to personal opinions of what constitutes a time-sensitive issue, ensuring for some level of instructor customization of the system could help address this at an individual level – much like the training of an email spam filter.

Chapter V - CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This thesis research focused on the design and testing of a number of NLP classifiers in order to determine the most accurate candidate for the TutorAlert algorithm that will be used to identify frustration, confusion, and other learning complications in a digital online learning environment. Ultimately the intention is for TutorAlert to be implemented as a plugin or extension of an existing online learning environment, or be hosted in a cloud environment, to provide alerts to students having time-sensitive learning issues, as well as providing a unique set of metrics to assist in the determination of weak course material that is consistently causing problems.

Figure 5-1 below lays out the final algorithm details of TutorAlert, including the SVM classifier and the data store which retains the trained state of the algorithm so that it does not need to be retrained from scratch every time it starts up.

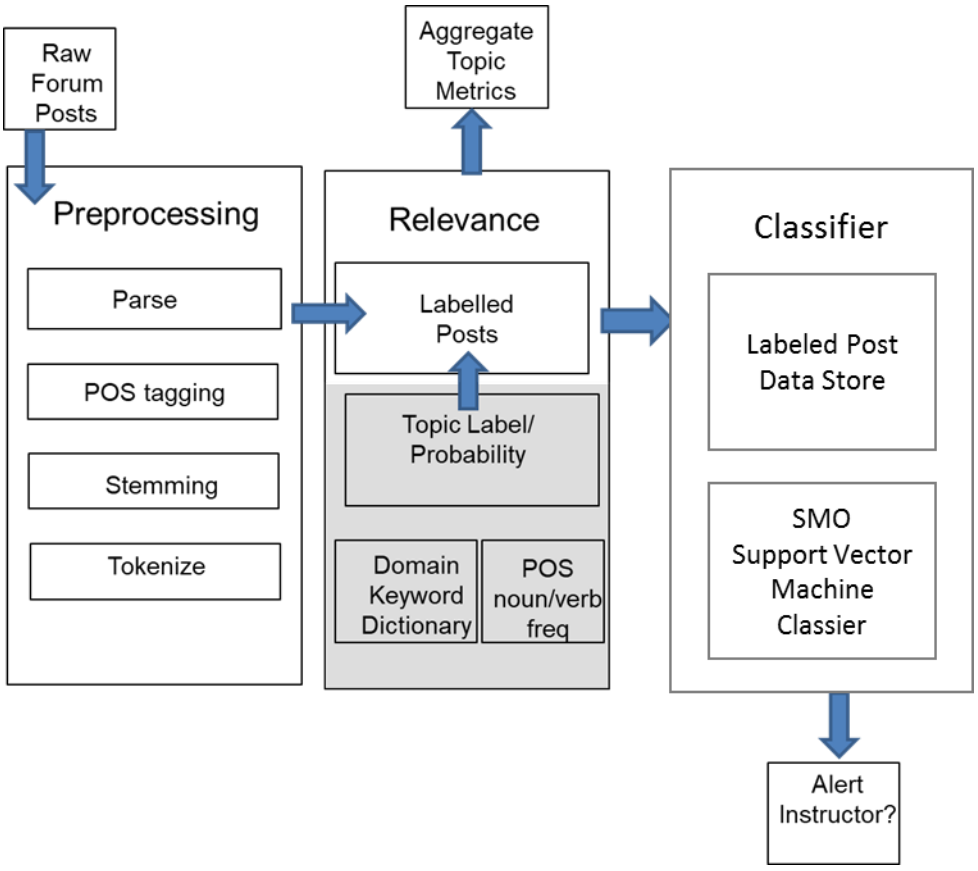

Figure 5-1: TutorAlert algorithm details.

Another thing that became apparent from the inter-rater reliability testing is that different instructors appear to harbour slightly different opinions on what constitutes a time sensitive discussion post. So even with TutorAlert trained to an acceptable default level, it would be beneficial to include a feedback loop that would continue to train each instance of the algorithm to the preferences of the individual instructor. For this function to be incorporated into TutorAlert two situations need to be considered – posts that are incorrectly labelled as frustration or confusion but are not (false positives), and those posts that are not labelled as indicating frustration or confusion but should be (false negatives). The former state is resolved in a

relatively straight forward fashion by including an “Ignore” button on each alert, that informs the classifier that the instructor does not view the issue as needing immediate action, as seen in Figure 5-2.

This function would, over time, allow instructors to continue to train and adjust the classifiers to suit their needs in much the same way many people train email spam features today by marking them as “Spam” or “Not Spam” in their various email folders.

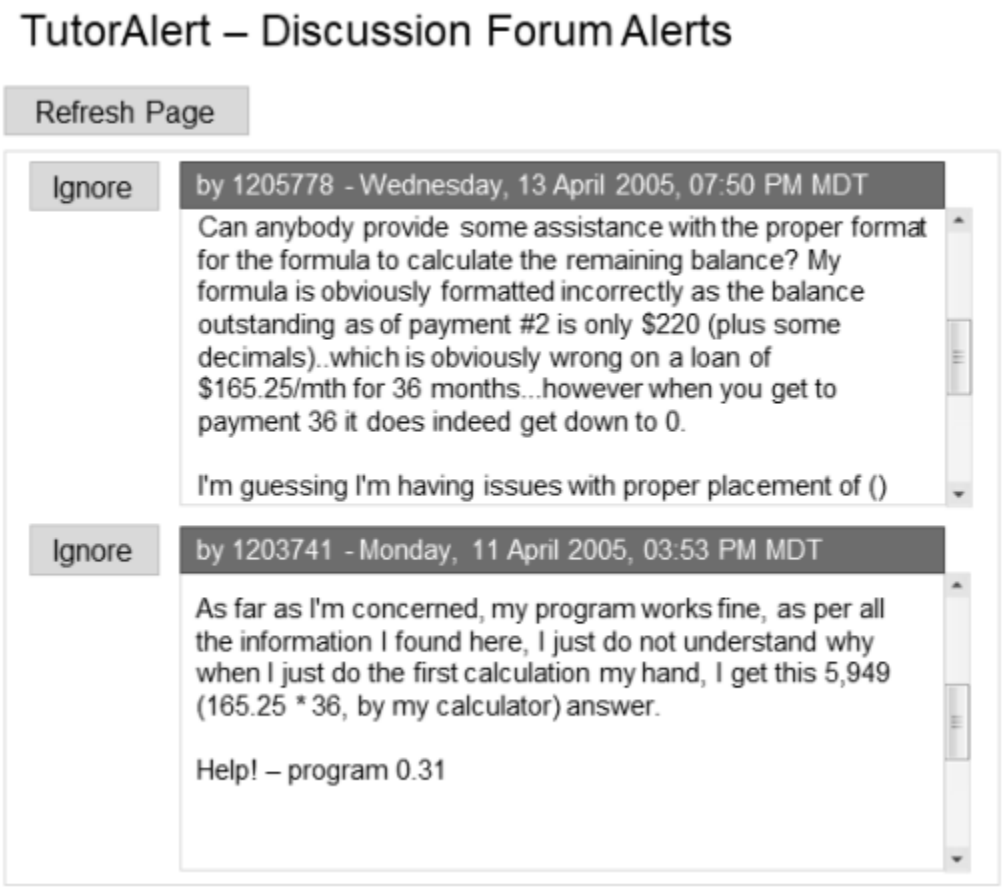

Figure 5-2: TutorAlert Web Interface showing alerts.

Designing a solution for the former situation where posts were mislabeled as not requiring a response is more difficult, and may require further integration in the digital classroom platform – be it Moodle, Blackboard, etc. Again, like an email spam function, this would require instructors to occasionally comb through posts classified as not requiring teacher intervention and allowing for some sort of correction action. For now, that remains outside of the scope of this research, but it will be an important next step in ensuring the ongoing utility of the project.

An additional fundamental goal was simplicity in design to allow for portability of TutorAlert to other domains or subject matter with relatively little training of development. Indeed, the use of a simplified custom dictionary pulled directly from course content is expected to aid in this goal. As mentioned previously, the nature of the binary classification should also ease the process of moving TutorAlert to other source subjects.

There are, of course, methods that could be employed that might allow for an increased number of dimensions or classifications states. Having access to additional data points, such as assignment grades, multimedia events, past course grades, parsing additional topic information from posts, or even considering interaction times with the digital classroom itself could conceivably improve TutorAlert's accuracy considerably in this area, though this would likely also require an increased amount of data for training purposes as well, as discussed in Chapter three.

In summary, while there are a number of compelling development and research directions that the TutorAlert could move towards, and indeed much of this will be discussed in the next

section on future directions, the overall accuracy of the classifier and the potential utility of the overall algorithm is very promising and definitely warrants further investigation.

Future Directions

Going forward, work will continue on TutorAlert on a number of fronts. First, further research and refinement of the keyword and attribute identification system might better determine specific course material or topics where numerous students typically have difficulty. The current solution is simple and serves the immediate purpose, but a more sophisticated system might be able to provide more valuable content and education metrics and learning analytics beyond merely estimating relevance.

Further testing of TutorAlert across different domain and subject materials is also a definite priority as that will be paramount in establishing its usefulness to educators and institution administrators moving forward.

Other research of interest would include investigating how the system is affected by gender or cultural differences, because overly polite or passive language could conceivably lead to an increase in false negatives, and, if so, might require some form of additional mitigation. In fact, further investigation into this area might also suggest ways to avoid students manipulating the TutorAlert system in order to receive more immediate help from instructors - in other words gaming the system to get to the front of the line.

Other possible future directions could be the incorporation of additional classifiers into TutuorAlert as a way of dynamically adding new dimensions or improving overall accuracy

across domains, without introducing unnecessary complication. This could range from a more traditional sentiment analysis classifier so that a post might be marked as both confusing and be causing negative sentiment; to other functions that attempt to determine learning style based on a student's greater interaction with specific content within the digital classroom. For instance are some students less prone to confusion with the presence of increased video content in the course material.

Currently, TutorAlert is operating on open source software that is best suited for prototyping machine learning applications. To grow beyond individual courses, to a department or university level, it would likely need to be deployed on a much larger platform. IBM's Watson is a good candidate for this, as it has enterprise equivalents of most of the functions through their natural language understanding libraries, and both Python and Java are available as programming language options.

Finally, work to expand the integrated functionality of TutorAlert through the creation of plug-ins or integration into popular online learning environments will also continue going forward. The growth in online education is only set to continue, and tools like TutorAlert can dramatically improve the overall learning experience for both instructors and students.

REFERENCES

[1] Ladyshewsky, R. K. (2013). Instructor Presence in Online Courses and Student Satisfaction. *International Journal for the Scholarship of Teaching & Learning* 7 (1), 1-23. http://doi.org/10.20429/ijsotl.2013.070113

[2] Liu, B., & Zhang, L. (2012). A survey of opinion mining and sentiment analysis. In *Mining Text Data*, 415-463. http://doi.org/10.1007/978-1-4614-3223-4_13

[3] Ellis, R. K. (2010). A Field Guide to Learning management systems. *International Anesthesiology Clinics*. http://doi.org/10.1097/AIA.0b013e3181e5c1d5

[4] Bannier, B. J. (2016). Global Trends in Transnational Education. *International Journal of Information and Education Technology 6* (1), 80-84. http://doi.org/10.7763/IJIET.2016.V6.663

[5] Simsek, A. (2013). Global trends in distance education. *International Conference on Communication, Media, Technology & Design*, 89. http://doi.org/10.1017/CBO9781107415324.004

[6] Bhandari, R., & Blumenthal, P. (2011). International students and global mobility in higher education: national trends and new directions. *International and development education*. Springer. http://doi.org/10.1007/s10734-011-9490-3d

[7] Daniel, J. (2012). Making Sense of MOOCs: Musings in a Maze of Myth, Paradox and Possibility. *Journal of Interactive Media in Education, 2012*(3). http://doi.org/10.5334/2012-18

[8] Romero, C., Ventura, S., & García, E. (2008). Data mining in course management systems: Moodle case study and tutorial. *Computers and Education*, *51*(1), 368-384. http://doi.org/10.1016/j.compedu.2007.05.016

[9] Maia, D., Belau, F., Rigo, S. J., Alves, I., Barbosa, J. L. V, & Hentges, A. (2016). A Module for Sentiment Analysis in Moodle. The *10th International Technology, Education and Development Conference,* 3404-3411. http://doi.org/10.21125/inted.2016.1802.

[10] Thoms, B., Eryilmaz, E., Mercado, G., Ramirez, B., & Rodriguez, J. (2017, January). Towards a Sentiment Analyzing Discussion-board. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

[11] He, Y., Lin, C., & Alani, H. (2011, June). Automatically extracting polarity-bearing topics for cross-domain sentiment classification. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1,* 123-131. http://doi.org/http://doi.acm.org/10.1145/1364782.1364798

[12] Zhu, Z., Dai, D., Ding, Y., Qian, J., & Li, S. (2013, February). Employing Emotion Keywords to Improve Cross-Domain Sentiment Classification. In *Chinese Lexical Semantics: 13th Workshop, CLSW 2012, Wuhan, China, July 6-8, 2012, Revised Selected Papers* (7717), 64-71. Springer. http://doi.org/10.1007/978-3-642-36337-5_8

[13] Wang, L., & Cardie, C. (2014). A Piece of My Mind: A Sentiment Analysis Approach for Online Dispute Detection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (2),* 693-699. arXiv preprint arXiv:1606.05704.

[14] Mao, Y., & Lebanon, G. (2007). Isotonic conditional random fields and local sentiment flow. *Advances in Neural Information Processing Systems*, 961-968. http://doi.org/10.1007/s10994-009-5139-1

[15] Kim, J., & Kang, J. H. (2014). Towards identifying unresolved discussions in student online forums. *Applied Intelligence (40),* 601-612. http://doi.org/10.1007/s10489-013-0481-1

[16] Chen, Z., Liu, B., Hsu, M., Castellanos, M., & Ghosh, R. (2013). Identifying Intention Posts in Discussion Forums. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 1041-1050.

[17] Koncz, P., & Paralic, J. (2011). An approach to feature selection for sentiment analysis. In *2011 15th IEEE International Conference on Intelligent Engineering Systems,* 357-362. http://doi.org/10.1109/INES.2011.5954773

[18] Liu, B. (2010). Sentiment analysis: A multifaceted problem. In *IEEE Intelligent Systems 25*(3), 76-80. http://doi.org/10.1109/MIS.2010.75

[19] Bouazizi, M., & Ohtsuki, T. (2016). Sentiment analysis: From binary to multi-class classification: A pattern-based approach for multi-class sentiment analysis in Twitter. In *2016*

IEEE International Conference on Communications, ICC 2016, 1-6. http://doi.org/10.1109/ICC.2016.7511392

[20] Liu, B. (2015). *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press. http://doi.org/10.1017/CBO9781139084789

[21] Kai, Y., Cai, Y., Dongping, H., Li, J., Zhou, Z., & Lei, X. (2017). An effective hybrid model for opinion mining and sentiment analysis. In *2017 IEEE International Conference on Big Data and Smart Computing, BigComp 2017*, 465-466. http://doi.org/10.1109/BIGCOMP.2017.7881759

[22] Ramos, J. (2003). Using TF-IDF to Determine Word Relevance in Document Queries. *Proceedings of the First Instructional Conference on Machine Learning, (242),* 133-142. http://doi.org/10.1.1.121.1424

[23] Moral, C., de Antonio, A., Imbert, R., & Ramírez, J. (2014). A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, *19*(1), n1. http://doi.org/10.9790/0661-17367680

[24] Asghar, M. Z., Khan, A., Ahmad, S., & Kundi, F. M. (2014). A review of feature extraction in sentiment analysis. *Journal of Basic and Applied Scientific Research*, *4*(3), 181-186.

[25] Jivani, A. G. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, *2*(6), 1930-1938.

[26] Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing - EMNLP '06,* 120–128. http://doi.org/10.3115/1610075.1610094

[27] Asghar, M. Z., Khan, A., Ahmad, S., & Kundi, F. M. (2014). A review of feature extraction in sentiment analysis. Journal of Basic and Applied Scientific Research, 4(3), 181-186.

[28] Wakchaure, S. L., Pawar, S. D., Ghuge, G. D., & Shinde, B. B. (2017). Overview of Anti-spam filtering Techniques. *International Research Journal of Engineering and Technology (IRJET), 4* (1), 429-434.

[29] Ray, S. (2015). 6 Easy Steps to Learn Naive Bayes Algorithm (with code in Python). *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2015/09/naive-bayes-explained/ Retrieved May 15, 2017.

[30] Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, 282-289. http://doi.org/10.1038/nprot.2006.61

[31] Al-Moslmi, T., Omar, N., Abdullah, S., & Albared, M. (2017). Approaches to Cross-Domain Sentiment Analysis: A Systematic Literature Review. *IEEE Access*, *5*, 16173-16192. http://doi.org/10.1109/ACCESS.2017.2690342

[32] Gupte, A., Joshi, S., Gadgul, P., & Kadam, A. (2014). Comparative Study of Classification Algorithms used in Sentiment Analysis. *(IJCSIT) International Journal of Computer Science and Information Technologies*. 5(5), 6261-6264.

[33] Go, A., Bhayani, R., & Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. *Processing* 1(12). http://doi.org/10.1016/j.sedgeo.2006.07.004

[34] Abirami, A. M., & Gayathri, V. (2017, January). A survey on sentiment analysis methods and approach. In *IEEE Advanced Computing (ICoAC), 2016 Eighth International Conference,* 72-76.

[35] Kim, J. D., Ohta, T., Tateisi, Y., & Tsujii, J. I. (2003). GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1), i180-i182. http://doi.org/10.1093/bioinformatics/btg1023

[36] Liu, H., & Singh, P. (2004). ConceptNet - a practical commonsense reasoning tool-kit. *BT Technology Journal*, *22*(4), 211-226. http://doi.org/10.1023/B:BTTJ.0000047600.45421.6d

[37] Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. John Wiley & Sons, Inc. http://doi.org/10.1139/h11-025

[38] Abbasi, A., France, S., Zhang, Z., & Chen, H. (2011). Selecting attributes for sentiment classification using feature relation networks. *IEEE Transactions on Knowledge and Data Engineering, 23*(3), 447-462.

[39] Ganesan, K., & Zhai, C. X. (2012). Opinion-based entity ranking. *Information Retrieval 15* (2), 116-150. http://doi.org/10.1007/s10791-011-9174-8

[40] Koncz, P., & Paralic, J. (2011, June). An approach to feature selection for sentiment analysis. In Intelligent Engineering Systems (INES), 2011 15th IEEE International Conference, 357-362. http://doi.org/10.1109/INES.2011.5954773

[41] Romero, C., López, M. I., Luna, J. M., & Ventura, S. (2013). Predicting students' final performance from participation in on-line discussion forums. *Computers & Education, 68,* 458-472. http://doi.org/10.1016/j.compedu.2013.06.009

[42] Wen, M., Yang, D., & Rose, C. (2014). Sentiment Analysis in MOOC Discussion Forums: What does it tell us? *In Educational Data Mining*, 146-150.

[43] Pan, S. J., Ni, X., Sun, J. T., Yang, Q., & Chen, Z. (2010). Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World Wide Web*, 751-760.

[44] Deshmukh, J. S., & Tripathy, A. K. (2018). Entropy based classifier for cross-domain opinion mining. Applied Computing and Informatics, 14(1), 55-64 http://dx.doi.org/10.1016/j.aci.2017.03.001.

[45] Eibe, F., Hall, M. A., Witten, I. H., & Pal, J. C. (2016). The WEKA workbench. Appendix for "Data Mining: Practical Machine Learning Tools and Techniques, 4th Edition.

[46] Eck, David J, *Introduction to Programming Using Java, Seventh Edition*. 2014. http://math.hws.edu/javanotes/index.html

[47] Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

[48] Eibe Frank, Mark A. Hall, and Ian H. Witten (2016). The WEKA Workbench. Online Appendix for *Data Mining: Practical Machine Learning Tools and Techniques, Fourth Edition*, 2016.

[49] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning* research, 12(Oct), 2825-2830.

[50] Jivani, A. G. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, *2*(6), 1930-1938.

[51] Tang, Lei & Liu, Huan. (2005). Bias analysis in text classification for highly skewed data. *Proceedings - IEEE International Conference on Data Mining*, 4, 1-9. http://doi.org/10.1109/ICDM.2005.34.

[52] Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies, 2* (1), 37-63. Powers, D. M. W. (2011). http://doi.org/10.1.1.214.9232

[53] Hallgren, K. A. (2012). Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in Quantitative Methods for Psychology*, *8* (1), 23. http://doi.org/10.20982/tqmp.08.1.p023

APPENDIX A – ETHICS APPROVALS

CERTIFICATION OF ETHICAL APPROVAL

The Athabasca University Research Ethics Board (AUREB) has reviewed and approved the research project noted below. The AUREB is constituted and operates in accordance with the current version of the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS) and Athabasca University Policy and Procedures.

Ethics File No.: **21867**

Principal Investigator:
Mr. Steven Harris, Graduate Student
Faculty of Science & Technology\School of Computing & Information Systems

Supervisor:
Dr. Vivekanandan (Vivek) Kumar (Supervisor)

Project Title:
Using Natural Language Processing to Identify Students Experiencing Academic Difficulty in a Digital Education Environment

Effective Date: August 19, 2015 **Expiry Date**: August 18, 2016

Restrictions:

Any modification or amendment to the approved research must be submitted to the AUREB for approval.

Ethical approval is valid *for a period of one year*. An annual request for renewal must be submitted and approved by the above expiry date if a project is ongoing beyond one year.

A Project Completion (Final) Report must be submitted when the research is complete *(i.e. all participant contact and data collection is concluded, no follow-up with participants is anticipated and findings have been made available/provided to participants (if applicable))* or the research is terminated.

Approved by: **Date:** August 19, 2015

Ali Akber-Dewan, Chair
School of Computing & Information Systems, Departmental Ethics Review Committee

---

Athabasca University Research Ethics Board
University Research Services, Research Centre
1 University Drive, Athabasca AB Canada T9S 3A3
E-mail rebsec@athabascau.ca
Telephone: 780.675.6718

Athabasca University
RESEARCH CENTRE

CERTIFICATION OF ETHICAL APPROVAL - RENEWAL

The Athabasca University Research Ethics Board (AUREB) has reviewed and approved the research project noted below. The AUREB is constituted and operates in accordance with the current version of the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS) and Athabasca University Policy and Procedures.

Ethics File No.: **21867**

Principal Investigator:
Mr. Steven Harris, Graduate Student
Faculty of Science & Technology\School of Computing & Information Systems

Supervisor:
Dr. Vivekanandan (Vivek) Kumar (Supervisor)

Project Title:
Using Natural Language Processing to Identify Students Experiencing Academic Difficulty in a Digital Education Environment

Effective Date: August 17, 2017 **Expiry Date**: August 16, 2018

Restrictions:

Any modification or amendment to the approved research must be submitted to the AUREB for approval.

Ethical approval is valid *for a period of one year*. An annual request for renewal must be submitted and approved by the above expiry date if a project is ongoing beyond one year.

A Project Completion (Final) Report must be submitted when the research is complete *(i.e. all participant contact and data collection is concluded, no follow-up with participants is anticipated and findings have been made available/provided to participants (if applicable))* or the research is terminated.

Approved by: **Date:** August 17, 2017

Joy Fraer, Chair
Athabasca University Research Ethics Board

Athabasca University Research Ethics Board
University Research Services, Research Centre
1 University Drive, Athabasca AB Canada T9S 3A3
E-mail rebsec@athabascau.ca
Telephone: 780.675.6718

APPENDIX B – TUTORALERT INTER-RATER RELIABILITY

The inter-rater reliability test for TutorAlert consisted of a bank of 250 student discussion posts from the test data, split evenly between posts that the algorithm registered an alert and the other half which did not. Three experienced college instructors participated in answering the questions, served online through Google forms, as demonstrated in Figure 0-1.

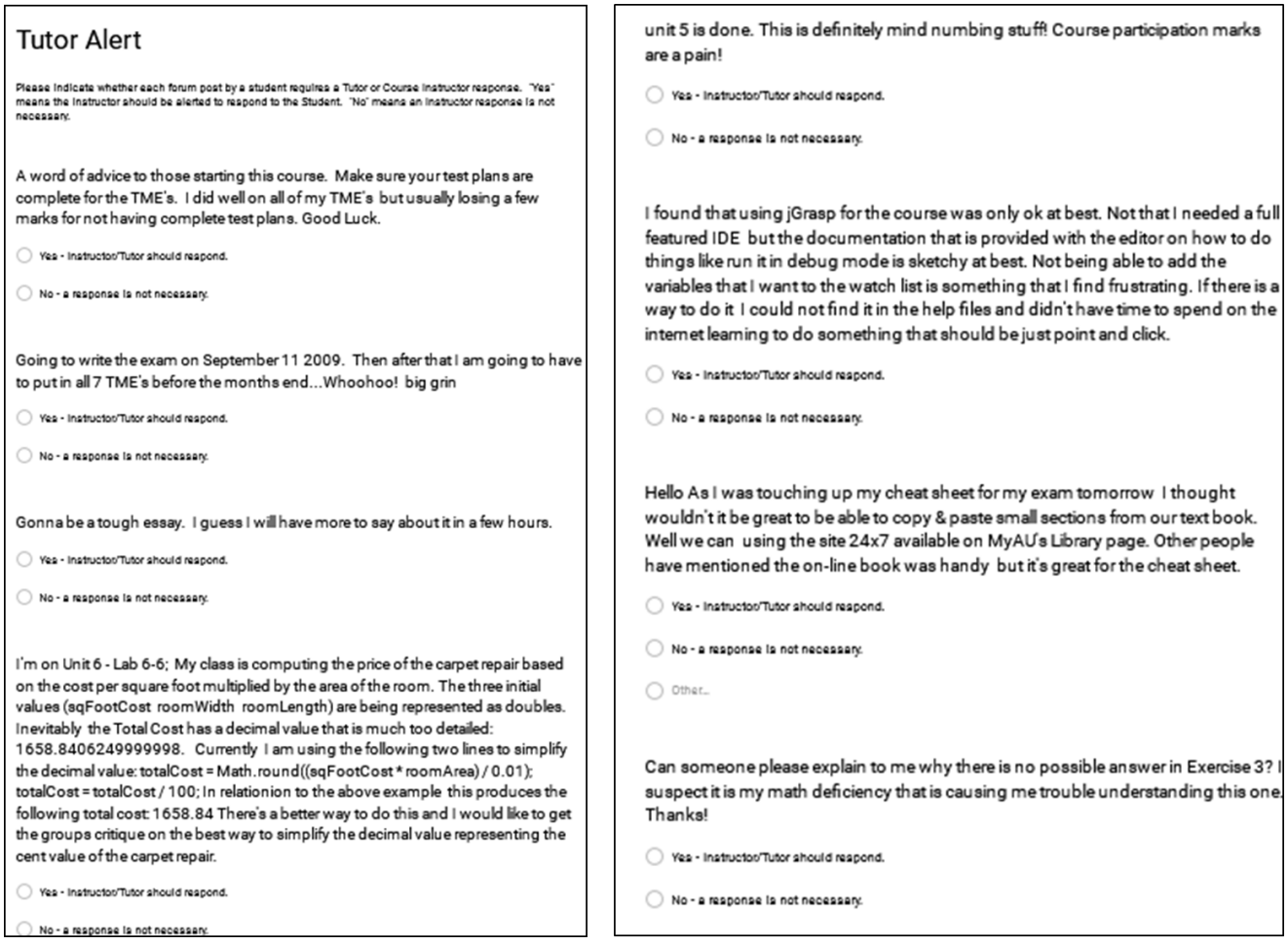

Figure 0-1: Inter-rater Reliability example.