DOCUMENT CLASSIFICATION

by

Shane K. Panter

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

May 2013

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Shane K. Panter

Thesis Title: Document Classification

Date of Final Oral Examination: 1 May 2013

The following individuals read and discussed the thesis submitted by student Shane K. Panter, and they evaluated his presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Tim Andersen, Ph.D.	Chair, Supervisory Committee
Amit Jain, Ph.D.	Member, Supervisory Committee
Teresa Cole , Ph.D	Member, Supervisory Committee

The final reading approval of the thesis was granted by Tim Andersen, Ph.D., Chair, Supervisory Committee. The thesis was approved for the Graduate College by John R. Pelton, Ph.D., Dean of the Graduate College.

Dedicated to Lori, Sara, and Talon Panter

ACKNOWLEDGMENTS

The author wishes to thank his family Lori, Sara, Talon, Wendy, Ashley, Kay, and Rosie for the support they have given him in completing this endeavor. Additionally, the support of the Computer Science faculty at Boise State University, specifically Tim Andersen, Amit Jain, and Teresa Cole, was invaluable. Luke Hindman helped keep me sane throughout the process by listening to my rants and raves over many cups of coffee. Thanks to my Mom for convincing me that I could in fact finish when I thought all was lost. This work was partially supported by DARPA grant number ST071-005 Open Source Information Geospatial Overlay (OSIGO) with the balance supported by Lori Panter.

ABSTRACT

We present an overview of the document classification process and present research conducted against the newly constructed SBIR-STTR corpus. Specifically, the current methods in use for annotation, corpus construction, feature construction, feature weighting, and classifier algorithms are surveyed. We introduce a new dataset derived from public data downloaded from `sbir.gov` and the Text Annotation Toolkit (TAT) [1] for use in classification research.

TAT is a collection of independent components packaged together into one open source software application. TAT was engineered to support the document classification process and work flow. Tracking of changes in a working corpus, saving data used in the training of classifiers to ensure reproducibility, and providing a mechanism for interacting with copyright protected corpora are all fundamental issues that TAT addresses. TAT is built using the robust Open IDE [35] framework that allows plug-in developers access to standard well tested libraries saving years of development time. The main goal of TAT is to minimize the labor intensive process of creating labelled data that can be used to train, test, and deploy machine learning models for automated text annotation. Additionally, TAT allows researchers an easy method to automatically reproduce prior results. The toolkit can facilitate the annotation of text using different machine learning packages as well as corpora with different metadata specifications.

[1]TAT is freely available for download from trac.boisestate.edu

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API – Application Programming Interface

TAT – Text Annotation Toolkit

SBIR – Small Business Innovation Research

STTR – Small Business Technology Transfer

RCV-1 – Reuters Corpus Volume 1

GUI – Graphical User Interface

MALLET – MAchine Learning for LanguagE Toolkit

TF – Term Frequency

IDF – Inverse Document Frequency

DR – Dimensionality Reduction

SVM – Support Vector Machines

POS – Part of speech

NE – Named Entity

ASCII – American Standard Code for Information Interchange

UTF – Unicode Transformation Format

SGML – Standard Generalized Markup Language

XML – Extensible Markup Language

HTML – Hypertext markup language

DTD – Document Type Definition

TEI – Text Encoding Initiative

GATE – General Architecture for text engineering

HUBZone – Historically Underutilized Business Zone

SBD – Small Disadvantaged Business

WOSB – Women Owned Small Businesses

SBA – Small Business Administration

CHAPTER 1

INTRODUCTION

1.1 Overview

The purpose of this thesis is threefold: to deliver the Text Annotation Toolkit (TAT), to develop a new dataset for testing, and to analyze Small Business Innovation Research (SBIR) and Small Business Technology Transfer (STTR) abstracts between the years 1983 and 2009 to determine factors that influence funding outcomes. TAT speeds the training of classifiers implemented using machine learning techniques, the testing of classifiers, manual and or automatic annotation, management of a collection of documents, and management of training data. TAT is a set of tools that bring together three independent activities: classification of documents, construction of a training and test set, and the management of trained data used to produce a classifier. TAT provides researchers and practitioners with the necessary framework and plumbing to foster new research and growth in the field of text classification.

We define document classification as the process of placing a set of documents into a predefined taxonomy of topics [40]. Manning et al. [27] notes that the notion of classification is very general and has a significant number of applications outside of the information retrieval discipline. Computers are not necessary for the classification process, books in a library for example have been classified manually for centuries. There are, however, significant drawbacks to manual classification such as the expense of labor and difficultly in scaling the volumes of documents processed [25, 16].

Document classification consists of several components that all need to work in harmony to achieve the desired results. First, a classification algorithm such as Naive Bayes or K-nearest neighbors must be implemented for use in text classification. Second, a data set of interest must be designed and built in order to train and test the classifier. Finally, if the classifier and data set are to be used by individuals other than computer experts, a clean and friendly front end must be developed to facilitate user interaction and management of the dataset and classifier.

There is need across several different industries to reduce the cost of classifying the ever increasing volume of documents. TAT helps facilitate this goal by providing a unified framework and interface. For example, manual classification of news articles by companies such as Reuters requires a prohibitive amount of human capital to maintain. The Reuters Corpus Volume 1 (RCV-1) corpus consists of over 800,000 manually categorized news wire stories for use in research. RCV-1 was produced in an operational setting between August 20, 1996 and August 19, 1997. Annotating the collection was a substantial undertaking. Lewis et al. cites one estimate at 12 person-years to annotate the RCV-1 collection [25]. Additionally, the amount of text available in on-line sources such as web pages is growing at a rapid pace [18]. Machine learning techniques can help to reduce costs and increase the volume of documents that can be processed. Thus, the reduction in the time required to classify documents is valuable for both production and research applications.

1.2 Background

While there are many toolkits and applications available, TAT occupies a unique niche that we believe has not been filled by any existing open source implementations. Other open source applications that provide a general solution to automatic classification tend to focus heavily on the construction of classifiers and classification pipelines, or solely on adding annotations to documents without giving the user access to an automatic classifier. The primary difference between TAT and other classification toolkits is simplicity, end user focus, and a standardized plug-in framework. TAT was developed to address the problem of quickly adding labels to documents, provide seamless change tracking for corpora, and give researchers an easy method to replicate results. TAT records all the information needed to setup and run a classifier so results are reproducible. This approach overcomes a long standing problem when dealing with copyright protected corpora that cannot be redistributed to end users. A researcher can simply redistribute the created metadata that can then be replayed by anyone who can gain access to the original data.

Abstracting the complexity of building and training document classifiers allows users unfamiliar with document classification techniques to quickly come up to speed and start contributing to a development corpus. For example, a domain expert in the medical field can accurately annotate medical documents using TAT without having knowledge of classification techniques. Thus, we think that TAT is an excellent tool for the construction and maintenance of new corpora by domain experts with limited knowledge of document classification. A non-exhaustive list of other open source applications that provide similar functionality includes General Architecture for text engineering (GATE) and Annotea. Section 1.2.1 and 1.2.2 briefly overview each application and compares them to TAT.

1.2.1 GATE

GATE [6] is a mature language engineering framework developed in Java. GATE has been under continuous development for more that 15 years and thus has support for multiple languages and a deep and complex feature set. The framework is designed to separate the various tasks such as data storage, data visualization, classification, and automatic measurement of classifier performance into a modular plug-in structure.

Currently, GATE has support for 28 different languages with more planned for future releases.

Due to the maturity of the platform, GATE provides all of the features of TAT across its plethora of plug-ins. Both manual and automatic annotation [22] modes are available and can operate against custom corpora. Additionally, GATE provides collaborative annotation and change tracking similar to TAT in the form of a web application.

The primary difference between TAT and GATE is in the plug-in structure, integrated annotation and classification environment, and transparent change tracking on working corpora. The change tracking feature allows researchers working on copyright protected corpora the ability to distribute the change sets so that results can easily be replicated by replaying the changes against a clean copy of the corpus.

1.2.2 Annotea

The Annotea project [47] is a W3C lead project to help the advancement of a semantic web. Amaya is the Annotea project's first implementation and provides a complete web browsing and authoring environment. While it does not provide an interface into any classifier, it does provide an interactive environment for adding annotations to Hypertext markup language (HTML) documents. Additionally, Amaya allows you to hook into an annotation server allowing teams of annotators to simultaneously annotate documents.

Amaya has several differences when compared to TAT. First, it does not provide access to an automatic classifier, which can be helpful when annotating a large set of documents all from a specific domain. Second, Amaya was specifically designed to work with a limited number of document types such as HTML. TAT allows the user to work with any document as long is there is a supporting plug-in.

1.3 Thesis Statement

For this thesis, the following research questions are addressed in Chapter 5 using a dataset consisting of SBIR/STTR phase I and phase II proposals and TAT.

Can we determine which phase I proposals are likely to make it to phase II based on the following criteria:

- The amount of money the award requests
- The properties of the proposal such as woman, minority, or Historically Underutilized Business Zone (HUBZoned)
- The origin of primary investigator's name and title
- The textual properties of the submitted abstract

1.4 Methods

TAT consists of several independent components that must work in harmony in order to allow efficient annotation, classification, and reporting. As shown in Figure 1.1, TAT consists of 4 different components. First, a graphical user interface (GUI) presents the classifier, reporting, and dataset to the end user. Second, a plugin structure allows different machine learning toolkits to be used against the chosen dataset. Third, a direct interface into the dataset allows users to review text, add annotations, and build training and test sets. Fourth, a reporting structure allows users to view the results of training and classification in a simple and straight forward manner. Finally, the management of trained classifiers including the transparent change management of all the relevant data to reproduce a trained classifier.

1.4.1 Classifier

TAT provides an interface into the classification toolkit of the users choice through TAT's pluggable architecture. The interface includes the following features:

- Train a new classifier on a selected dataset
- Test a classifier on a selected dataset
- Report performance of trained classifiers on a selected dataset

Training a classifier is dependent on the classification algorithm selected. Different algorithms have different parameters and weights that the user can set, thus presenting a custom interface. All classifiers are evaluated using the same criteria, thus testing results are displayed the same regardless of the classifier implementation.

Evaluating the performance of classification algorithms uses a standard set of measurements widely known and used in the field of Information Retrieval (IR) [27]. First, we measure the average precision and recall of a classifier. Precision is defined as the number of documents correctly placed in a category by the classifier divided by the total number of documents retrieved. Recall is the number of documents correctly placed in a category by the classifier divided by the total number of documents in the category. Finally, taking the recall and precision scores, a weighted average known as the F-measure is calculated and presented to the user. All formulas are defined in Table 2.4.

1.4.2 Dataset

A new data set has been constructed using SBIR/STTR abstracts between the years 1983 and 2009. Developing new collections is important in the field of document classification and annotation. Machine learning algorithms can over-fit by tuning a classifier's parameters to properties in the training set that are accidental. Additionally, readily available non-proprietary datasets allow researchers to test ideas without having to hire human annotators. According to Lewis et al., existing test collections suffer from one or more of the following weaknesses: few documents, lack of the full document text, inconsistent or incomplete category assignments, peculiar textual properties, and or limited availability [25]. Thus, providing a new dataset for the research community to use is a valuable contribution.

The dataset that we built is derived from SBIR and STTR data sources freely available to the public. Unfortunately, the data is spread across 11 different government agencies all with different formats and reporting methods. Thus, we defined a common extensible markup language (XML) format to use in TAT that we then used to represent the SBIR and STTR abstracts and supporting information. It is important to note that the derived data set has been annotated automatically using characteristics of the dataset instead of using humans to annotate the data.

1.5 Artifacts

During the course of this thesis, the following artifacts have been delivered.

- An annotated dataset in XML format consisting of SBIR/STTR phase I and phase II abstracts between the years 1983 and 2009

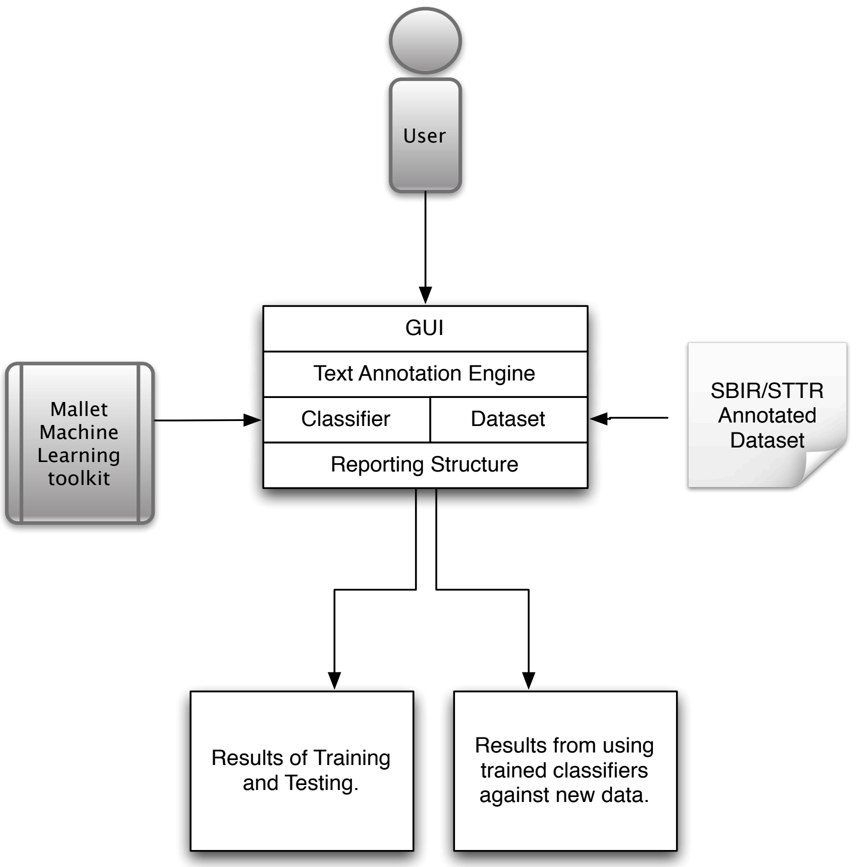

User
GUI
Text Annotation Engine
Classifier
Dataset
Reporting Structure
Mallet Machine Learning toolkit
SBIR/STTR Annotated Dataset
Results of Training and Testing.
Results from using trained classifiers against new data.

Figure 1.1: High level architecture of TAT. TAT consists of several independent components that are designed to work together to give a seamless user experience.

- A Text Annotation Toolkit that allows a user to train, test, and use machine learning algorithms such as Naive Bayes
- A plugin for the mallet machine learning toolkit to be used with TAT

1.6 Summary

Developing TAT provides both the machine learning and information retrieval communities with a valuable tool to facilitate research of text annotation and classification techniques by providing a clean and crisp interface into the classification process. The research questions enumerated in Section 1.3 were answered using TAT to demonstrate the capabilities and usefulness of the toolkit. Additionally, deep insight into the SBIR/STTR award process, which affects small businesses and universities such as Boise State, has been provided to the grant-seeking community.

CHAPTER 2

CLASSIFICATION

2.1 Overview

Finding documents based solely on keywords such as in a traditional book index fail when searching for a document that has the desired content but lacks the required keywords. For example, the Reuters news agency may want to identify documents about corporate acquisitions. However, if the stories do not contain any of the "required" key words, then an interested reader may not be able to find all the relevant documents [16]. Thus, classifying documents by their semantic meaning and providing a method to retrieve a set of documents with high precision and recall provides an invaluable tool in the management of large scale document repositories.

Machine learning based text classification allows a classifier to learn a set of rules, or the decision criterion, from a set of labeled data that has been annotated by an expert. This approach allows for better scaling and a reduced cost in classifying documents when compared to a system that relies on manual input only. Most of the research in the field of machine learning based document classification has been done on binary classifiers [27, 42, 32, 1]. That is, building a classifier from a set of positive and negative examples to then deduce a document's membership in a class. When dealing with a corpus that has documents that belong to multiple classes, the approach thus far has been to build a separate binary classifier for each class in the corpus [25] and then aggregate the results of each binary classifier.

2.2 Classification

Classification can take many forms, from fully automated systems with limited human intervention [16] to semi-automatic systems that employ a hybrid human machine approach [46]. The classification process includes constructing the features that will represent a document, weighting said features, and algorithms to process and rank the documents.

Transforming a document from a string of characters into a representation that is suitable for the selected learning algorithm and classification task is generally the first step in any text categorization process [19]. This process assumes that any unnecessary characters such as formatting tags or other meta-data not related to the text have been removed. Transforming the document entails the selection features that give a good representation of the domain and a weighting scheme that the classification algorithm can use.

The training algorithm attempts to approximate an unknown function f such that $f : d \rightarrow c$ with d representing the document in question and c the class of the document chosen from a available set of classes $\mathbb{C}$. With the function derived from the training phase, we can use f to classify unseen documents into the proper class.

With the classification of documents, there are several difficulties that must be overcome to achieve the greatest level of accuracy. Li and Jain [26] highlight the fact that it is difficult to capture high-level semantics and the abstract concepts of natural languages from a few key words. We can take a look at a selection of words that represent different meanings depending on the context in which they are used. For example, the word "worm" in computer science refers to a self-replicating malicious piece of code. However, the same word used in the context of nature refers to a soft body invertebrate animal. There are also extensive use of synonyms throughout natural language such as the term "student" and "pupil", which mean the same thing. Additionally, large numbers of words and the inter-dependency between these words becomes a problem.

When describing the components of a classification system, the following notation is used:

- $\mathbb{V}$ - The vocabulary of the collection with $|\mathbb{V}|$ equal to the total number of unique features across all the documents
- $\mathbb{C}$ - The set of available class labels
- d - A single document to classify or train on.
- f - A feature of the document.
- $\mathbb{D}$ - A collection of Documents
- A - A sparse matrix consisting of $N \times M$ rows and columns. Each row N is a single document and each column $M \in \mathbb{V}$

2.2.1 Document Representation

Document representation is an important part of the classification process. Each document must be processed into a form that a classifier can operate on and still preserve as much as the original information as possible [8]. The current best known approach is to use a vector space model to represent the document as seen in Equation 2.1 [8]. Each dimension of the document d corresponds to a unique feature f (term) and each feature is determined by the system in use. Typically, a feature is simply a word, however any representation can be used depending on the data being classified.

$$d_i = (f_{1i}, f_{2i}, \cdots, f_{ji}) \tag{2.1}$$

Different methods exist to construct the features of each document. Each method has strengths and weakness and must be evaluated for each given document domain.

Additionally, after the construction of a feature set for $\mathbb{D}$, the number of features may be too large to process or too sparse to give good results.

2.2.2 Bag of Words

One common approach when constructing features is to treat the document as a "bag of words." This approach takes all the words in a document and places them in a vector that represents the feature set of the document. This vector is unordered and does not necessarily represent any structure between the words. An example of the "bag of words" approach can be seen in Table 2.2 with the example sentences listed in Table 2.1. The example in Table 2.2 uses a simple boolean weight approach with 1 representing the feature as present and 0 as not present. More comprehensive weighting approaches that can improve classification accuracy are detailed in Section 2.3.

The biggest advantage to the "bag of words" approach is its simplicity in representation. No special knowledge of the language in question is needed in order to design and implement a classifier [12]. Simply tokenize the given words and construct the document vectors. However, there are some drawbacks with this approach. For example, multi-word semantic phrases in the language are lost during the processing. Additionally, depending on the size of your training corpus you could have a $\mathbb{V}$ that is too large for any classifier to process.

Table 2.1: Example sentences

number	Document
d1	Sally goes to a market
d2	Sally was eaten by a bear
d3	Right to bear arms

Using the example "documents" as listed in table 2.1 $\mathbb{V}$, $\mathbb{D}$ and A are calculated.

$\mathbb{V}$ =<Sally, goes, to, market, was, eaten,by,a,bear,Right, arms>

$\mathbb{D}$ =<d1,d2,d3>

$N \times M = (\mathbb{D}, \mathbb{V})$

Table 2.2: A using simple boolean representation

		Sally	goes	to	market	was	eaten	by	a	bear	Right	arms
$\uparrow$	d1	1	1	1	1	0	0	0	1	0	0	0
N	d2	1	0	0	0	1	1	1	1	1	0	0
$\downarrow$	d3	0	0	1	0	0	0	0	0	1	1	1

$\longleftarrow M \longrightarrow$

2.2.3 N-grams

Another method that can be used to construct the features of a document is a N-gram approach. A N-gram is a N-character slice from a longer string [3]. When constructing features using N-grams, we must note that uni-grams (N=1) would be a simple reflection of the distribution of the letters that make up the documents language alphabet.

N-grams have several attractive features that make them a compelling choice to use in feature construction [3, 12].

- construction is very easy and independent of language
- N-gram features automatically perform certain kinds of stemmings and are robust against misspellings
- multi-word phrases are automatically captured when N != 1

When using N-grams for feature construction several decisions need to be made. For example, would raw text be preprocessed to remove special characters such as

"!%&$" or are these characters kept. There is also the choice of distinguishing between upper and lower case characters.

N-grams sound like an exciting and natural way to represent documents. However, they are not without drawbacks. For example, given that it is possible for a gram to bridge words when N is greater than 1, we can have a vocabulary $\mathbb{V}$ that is too sparse for a classifier to use. Thus, the use of N-grams must be evaluated within the domain to determine an optimal value of N and the size of $\mathbb{V}$.

2.3 Feature Weight

After the features of a document have been constructed, it is often necessary to weight the constructed features in order to improve classifier accuracy. Methods to weight the features of a document vector are an exciting and deep area of research. The simplest and easiest method is to assign the features a simple boolean value as shown in Table 2.2. This, however, can result in sub-par performance as noted in a study by Goller et al. [12]. The following sections detail current weighting methods that have been developed. We review weighting by term frequency, document frequency, and finally a combination of both term and document frequency into a single score.

2.3.1 Term Frequency (TF)

Simple boolean representation can be improved by counting the number of times a specific feature occurs in a document. Equation 2.2 details calculating the weight of a feature from a document, tf_k is the weight for a specific feature f in document k. It is logical to assume that the more times a feature is present, the more important that feature is in determining the semantic meaning of the document [27]. Thus, weighting features with TF singleton features will not skew the training of classifiers.

$$tf_k = freq_{fk} \tag{2.2}$$

TF suffers from poor performance when we have a domain with extremely high feature frequencies that do not provide any useful information. This can happen when classifying documents that all originate from a narrowly focused industry or subject matter. For example, if your document corpus consists only of documents from the auto industry, terms such as “auto” or “car” may have high weights but provide little semantic meaning of the document content [27].

2.3.2 Inverse Document Frequency (IDF)

Inverse Document Frequency (IDF) is a weighting formula listed in Equation 2.4 that attempts to scale down the effect of features that occur too frequently in a collection and thus are not as valuable in determining the semantic meaning of a document.

df_f is the document frequency of a feature and is defined as the number of documents in $\mathbb{D}$ that contain feature f. Taking the number of documents in the collection $|\mathbb{D}|$ divided by df_f, we can determine the IDF of a feature for a particular document. By using the IDF of a document, terms that are rare provide more value than frequent terms in a collection.

$$idf_f = \log \frac{|\mathbb{D}|}{df_f} \tag{2.3}$$

One clear drawback to using IDF is that singleton terms become highly influential when ranking documents. Thus, TF and IDF both have weaknesses when dealing with outlier features.

2.3.3 TF-IDF

Both TF and IDF have qualities that allow features to negatively influence the ranking of documents within a collection. We now look at a method to combine TF and IDF in order to get the benefits while minimizing the drawbacks [1].

$$idf\text{-}idf_{fd} = tf_{fd} \times idf_f \tag{2.4}$$

With TF-IDF, we assign a weight to a feature that has the following characteristics [27, 1].

- Highest for a feature that occurs many times in a small number of documents
- Medium for a feature that occurs fewer times in a document, or occurs in numerous documents in low frequency
- Lowest when we have a feature that occurs in virtually all documents

2.4 Curse of Dimensionality

It is important to note that when classifying textual documents a unique problem surfaces. Several studies [1, 8, 42] state that one important step in text classification is the reduction of the number of features in a training set. High dimensionality of the feature space is not conducive for standard classification techniques because calculating the score of a document becomes computationally prohibitive. Thus, we review several techniques in the following sections such as feature selection, word stemming, and stop word removal.

2.4.1 Feature Selection

Feature selection is also referred to in other texts as dimensionality reduction (DR) [1, 42]. When dealing with natural language, the size of the vocabulary $|\mathbb{V}|$ could prove costly for a learning algorithm to consider. Consequently, reducing $|\mathbb{V}|$ with respect to $\mathbb{D}$ is a very active research topic.

With the ever increasing number of text documents available for training, high-dimensional feature sets are not uncommon. In the context of document classification, a high-dimensional feature set (a large value for $|\mathbb{V}|$) presents computational challenges at the implementation level. At some point, it becomes impractical to represent every word or N-gram in a given language as a feature. High-dimensional feature sets also increase the model complexity for many classifiers, which could lead to over-fitting [12].

When choosing the features to represent a document, there have been two approaches developed. Sabastiani and Fuka both define processes for choosing features [42, 8]. The first method is to simply choose the features of a document such that the chosen features are a subset of the original set. This can be done on either the global level (across all documents) or local level (per document). The second method is to produce a whole new set of features from the existing feature set. That is, the features are not a subset of the original features but rather they are generated by combinations or transformations on the feature set.

2.4.2 Word Stemming and Stop Word Removal

When text is taken as raw input, there is a lot of noise that is not critical for the classification process. For example, words such as “and,” “the,” or “a” do not provide much value when one is attempting to assign a document to one class or the other. Thus, one very simple way of reducing the feature set is to remove these common words, typically referred to as “stop-words” [19]. We can generate a stop word list from a document collection (typically referred to as a corpus) by collecting the most frequent terms and hand-filtering them for semantic content relative to the domain of the documents being classified [27]. These words are then removed from the feature set. Looking at $\mathbb{V}$ from Table 2.2, we can see that the stop words we would filter would be “to” and “a.”

A second very effective way in reducing the number of words that a classifier must deal with during training or classification is word stemming. There has been some research [19] in the area of information retrieval that suggests that representing the features of a document as word stems works well. More precisely each distinct word stem as extracted from the document by a stemmer such as the Porter [37] Stemming Algorithm can be included in the feature set. For example, the words walking, walked, walker, and walk all could be represented by the single word walk [1]. Using our running example from Table 2.2, we can see that “goes” could be reduced to “go,” “eaten,” to “eat,” and “arms” to “arm.”

2.5 Classification Algorithms

There have been many classification algorithms developed over the years. There are several implementations of these algorithms such as those in Machine Learning for Language Toolkit MALLET [29]. In the following sections we provide a high level-review of two very popular algorithms used in the domain of document classification.

2.5.1 Naive Bayes

Naive Bayes is a classifier that is based on Bayes' theorem. As stated in the Stanford Encyclopedia of Philosophy, “Bayes' theorem is a simple mathematical formula used for calculating conditional probabilities” [20]. One problem that is encountered with classifiers based on Bayes theorem is they quickly become intractable with even a small feature set. This makes using Bayes theorem without modifications impractical. Thus, Bayes theorem has been modified with the naive assumption of conditional independence to make it usable for document classification.

As stated by McCallumzy and Nigamy [30], the Naive Bayes classifier uses a simple probabilistic model to classify text. The model “assumes that all attributes of the examples are independent of each other given the context of the class.” This assumption is referred to as the “Naive Bayes assumption.” Zhang [48] proposed an explanation of the performance of Naive Bayes by showing that even with strong dependencies between features, Naive Bayes can still give optimal results.

There are many variations and extensions of the basic Naive Bayes algorithm. Two of the most basic variations deal with the underlying data models that are used to represent the document. The first model represents a document with a vector of binary attributes indicating which words occur in said document. This model does not however capture the number of times a word occurs, only that it did occur. The second model represents a document by the set of word occurrences. Both models disregard the order of the words with their representation [30].

2.5.2 Support Vector Machines (SVM)

Support Vector Machines (SVM) were invented by Vladimir Vapnik and others at AT&T Bell Laboratories in 1995. SVM's map the input vectors into some high dimensional feature space through some non-linear mapping [5]. They are based on the structural risk minimization principle from computational learning theory [19]. A side effect of SVM's being based on a theoretical model of learning is that they come with theoretical guarantees about their performance. In their basic form, SVM's learn a linear threshold function. Nevertheless, by a simple "plug-in" of an appropriate kernel function, they can be used to learn polynomial classifiers, radial basic function (RBF) networks, and three-layer sigmoid neural nets. Besides using SVM in non-linear mapping, it is also possible to use simple linear SVMs (LSVM). Hearst et al. state that simpler LSVMs are fast to learn and provide good generalization accuracy [17].

Joachims asserts that SVMs perform exceptionally well because they are well suited to a high dimensional feature space with few irrelevant features and sparse instance vectors [19]. These three properties are fundamental when dealing with assigning a class to a document. SVMs seek to find a decision surface based on support vectors. They then attempt to maximize the margin between said support vectors to find the optimal hyperplane. We know what the support vectors are because removal of a support vector would obviously change the position of the hyperplane.

2.5.3 Evaluation of Performance

When evaluating classification algorithms, there is a standard set of measurements that we can compare against. Table 2.3 details the data necessary to calculate the

	Human: yes	Human: no
Classifier: yes	true positive (TP)	false positive (FP)
Classifier: no	false negative (FN)	true negative (TN)

Table 2.3: Statistics collected on classifiers

precision	$TP/(TP + FP)$
recall	$TP/(TP + FN)$
F-Measure	$2 * precision * recall/(precision + recall)$

Table 2.4: Formulas for precision recall and F-Measure

standard measures with the formulas given in Table 2.4. Measuring the effectiveness of a classifier using well-known methods allows researchers to compare results across implementations. It is important to note that the quality of the corpus that the classifiers are trained against can have an impact on performance metrics. It is possible for overfitting to occur when training your classifier, which can arbitrarily boost results. We can mitigate the effects of overfitting with k-fold cross validation as explained in Section 3.5, however, with a poor quality corpus it may still not result in an accurate performance measure.

2.6 Summary

This chapter has explored the overall classification process. We looked at ways to represent a document so that a classifier can process it during both the training and classification phase. We also reviewed several methods for weighting features within a document and reducing the number of features that a classifier has to deal with. Finally, we looked at methods to evaluate the overall quality of the implemented classifier to ensure results are in line with expectations.

CHAPTER 3

CORPORA

3.1 Overview

Corpora (singular corpus) are an integral part of the automated classification process. A corpus is defined as “all the writings or works of a particular kind or on a particular subject” [31]. While it may be possible, although improbable, to collect all works currently in existence in a particular domain and classify them for semantic meaning, it is not possible to collect future works and classify them, until of course we invent a time machine. Thus, a corpus representing a particular domain allows us to train a classifier and then use said classifier to classify new unseen documents into the appropriate categories.

In this chapter, we explore methods for creating a corpus that we can use to train a classifier. A brief overview of existing data sets commonly used in research for testing and comparing new classifier algorithms will be presented. Methods for annotating a document with the appropriate class, encoding a document for long-term storage and processing, and finally building training and test sets used to confirm the quality of the classifier will be presented.

3.2 Corpora

The primary purpose of a classifier is to assist in the building and maintenance of large collections of documents. Just as you would have to train a human to place new unclassified documents into the correct class you must also train a classifier. When training a classifier, a corpus acts as a set of examples with which we can compare new instances and then pick the class that has the greatest similarity in features.

Several domain specific corpora have been developed and released for public use over the years and have been used as a benchmark to test new ideas and implementations. In the field of document classification, the most well-know corpora are the Reuters-21578 collection and the newer RCV-1. The Reuters-21578 corpus is a collection of documents that appeared on the Reuters newswire in 1987. It has largely been replaced by the newer and larger RCV-1 [24]. Using these collections for the training of new classification algorithms allows a results-only comparison of classification precision and recall. In addition to an overview of RCV-1, we will present a new corpus derived from publicly available sources, which can be used by researchers in document classification research.

3.2.1 Reuters

RCV-1 consists of over 800,000 manually categorized news-wire stories for use in research purposes. RCV-1 was produced in an operational setting between August 20, 1996 and August 19, 1997. Annotating the collection was a substantial undertaking. Lewis et al. cites one estimate at 12 person-years to annotate the RCV-1 collection [25]. A sample document from the RCV-1 data set is shown in Listing 3.1.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<newsitem itemid="2904" id="root"
     date="1996-08-20" xml:lang="en">
<title>USA: MRS Technology says easing off merger hopes.</title>
<headline>MRS Technology says easing off merger hopes</headline>
<dateline>CHELMSFORD, Mass 1996-08-20</dateline>
<text>
<p>MRS Technology Inc, encouraged by momentum in orders for a line of
    its printers, said Tuesday it plans to &quot;soft pedal&quot; its
    recent effort to court a merger partner.</p>
<p>The company cited two orders from U.S. semiconductor manufacturers
    for its recently introduced Model 5200GHR PanelPrinter in this year'
    s fourth quarter.</p>
<p>Although it plans to ease away from courting a merger partner, the
    company said it will continue to consider forging relationships with
     two or more beta-site users for the company's new Model 70000
    PanelPrinter.</p>
</text>
<copyright>(c) Reuters Limited 1996</copyright>
<metadata>
<codes class=”bip:countries:1.0”>
  <code code=”USA”>
    <editdetail attribution=”Reuters BIP Coding Group”
     action=”confirmed”
     date=”1996-08-20”/>
  </code>
</codes>
<codes class=”bip:industries:1.0”>
  <code code=”I34531”>
    <editdetail attribution=”Reuters BIP Coding Group”
     action=”confirmed”
     date=”1996-08-20”/>
  </code>
</codes>
<codes class=”bip:topics:1.0”>
  <code code=”C11”>
    <editdetail attribution=”Reuters BIP Coding Group”
     action=”confirmed”
     date=”1996-08-20”/>
  </code>
  <code code=”C18”>
    <editdetail attribution=”Reuters BIP Coding Group”
```

```
      action="confirmed"
      date="1996-08-20"/>
  </code>
  <code code="C181">
     <editdetail attribution="Reuters BIP Coding Group"
      action="confirmed"
      date="1996-08-20"/>
  </code>
  <code code="C31">
     <editdetail attribution="Reuters BIP Coding Group"
      action="confirmed"
      date="1996-08-20"/>
  </code>
  <code code="CCAT">
     <editdetail attribution="Reuters BIP Coding Group"
      action="confirmed"
      date="1996-08-20"/>
  </code>
</codes>
<dc element="dc.date.created" value="1996-08-20"/>
<dc element="dc.publisher" value="Reuters Holdings Plc"/>
<dc element="dc.date.published" value="1996-08-20"/>
<dc element="dc.source" value="Reuters"/>
<dc element="dc.creator.location" value="CHELMSFORD, Mass"/>
<dc element="dc.creator.location.country.name" value="USA"/>
<dc element="dc.source" value="Reuters"/>
</metadata>
</newsitem>
```

Listing 3.1: Sample Document from RCV-1

There are several inconsistencies, anomalies with the coding scheme, and outright assumptions within the Reuters RCV-1 collection. Problems include duplicate documents or near duplicate documents, foreign language documents, miscoded documents, and an unknown coding policy. For example, a coding problem that is pervasive within the corpus is shown in Listing 3.2. We see that "AGRICULTURE, FORESTRY AND FISHING" has multiple codes assigned to it with different levels of padding. Lewis et al. state that the coding anomalies were likely caused by constraints imposed by legacy auto-coding software. However, eliminating the anomaly by compressing the codes into a single unique code assumes that the original system did intend for them to be the same. This assumption is at best an educated guess, as Lewis et al. state that reproducing the hierarchical structure in which the codes were originally embedded is exceedingly difficult [25].

```
I0      AGRICULTURE, FORESTRY AND FISHING
I00     AGRICULTURE, FORESTRY AND FISHING
I000    AGRICULTURE, FORESTRY AND FISHING
I0000   AGRICULTURE, FORESTRY AND FISHING
I00000  AGRICULTURE, FORESTRY AND FISHING
```

Listing 3.2: Sample industry codes

Despite the problems within RCV-1, it still provides value to the research community. It represents data collected and annotated in a constrained real-world environment and is a fair approximation of what researchers working on new classification algorithms can expect to see. Keeping the dataset as close to the original as possible allows researchers to evaluate real-world data instead of a clean room implementation that will artificially boost classification results. Lewis et al. outline steps to produce a test collection for research to this end [25].

3.2.2 SBIR-STTR

The SBIR and STTR programs were created by the 1982 Small Business Innovation Development Act and was reauthorized in 2011. The purpose of the SBIR/STTR program is to stimulate high-tech innovation in the United States by engaging in Federal Research and Development that has the potential for commercialization [13]. Current laws require any federal agency with a Research and Development budget in excess of 100 million dollars to allocate 2.5% of their budget to the SBIR and STTR programs.

Each agency with an SBIR/STTR program is responsible for reporting their grants to the Small Business Administration (SBA). The SBA provides a search interface for the grants that have been awarded across all agencies. This search gives the basic abstract information from the submitted proposal. Some agencies such as the USDA provide additional information from their local reporting page such as program progress updates. However, agencies such as the DOD only report abstracts. Thus, the only data that is guaranteed across all granting agencies are the abstracts from awarded proposals.

The current list (November, 2012) of SBIR/STTR granting agencies are as follows:

- United States Department of Agriculture (USDA)
- National Institute of Standards and Technology (NIST)
- National Oceanic and Atmospheric Administration (NOAA)
- Department of Defense (DOD)
- Department of Education (ED)

- Department of Energy (DOE)
- Department of Health and Human Services (HHS)
- Department of Homeland Security (DHS)
- Department of Transportation (DOT)
- Environmental Protection Agency (EPA)
- National Aeronautics and Space Administration (NASA)
- National Science Foundation (NSF)

The SBIR-STTR corpus is constructed from past proposal abstracts from the listed agencies between the years 1983 and 2009. The corpus has been automatically annotated to give classifiers a set of positive and negative examples to train from. Each proposal that was selected for both phase I and phase II is labeled as a positive example. Proposals that were only selected for a phase I award are labeled as negative examples. A sample document shown in Listing 3.3 would be a negative example.

Obtaining the SBIR-STTR data is straight forward, as the program provides an application programming interface (API) to download and or query award information [13]. First, the awards are downloaded in bulk by year. Once we have the awards downloaded, we can query the API to obtain the agency tracking number to correlate phase I and phase II abstracts. Chapter 5 gives an in-depth analysis of the number of positive and negative examples that exist in the corpus and how they were derived. Due to the time gap between a phase 1 and phase 2 award, it may be necessary to omit awards in order to prevent counting a phase I that was awarded in 2009 with a subsequent phase II awarded in 2010 as a negative example.

```
<?xml version="1.0" encoding="UTF-8"?>

<Award>
    <id>67854</id>
    <title>Improved Ultrafilters for Bioprocessing</title>
    <link>http://www.sbir.gov/sbirsearch/detail/67854</link>
    <abstract>Efficient separation and purification of bioproducts on a
        commercial scale are controlling steps in the bioprocessing of
        high value-added products such as therapeutics in human health
        care. This project will examine the feasibility of achieving
        significant improvement in downstream processing through-put by
        the use of flow-induced vortices to reduce or eliminate the
        effect of concentration polarization and fouling of the membrane
         surface in ultrafiltration applications.</abstract>
    <agency>HHS</agency>
    <program>SBIR</program>
    <phase>1</phase>
    <year>1993</year>
    <company>Abel Company</company>
    <ri></ri>
    <agency-tracking>22280</agency-tracking>
    <topic-code>N/A</topic-code>
    <woman-owned>No</woman-owned>
    <minority-owned>No</minority-owned>
    <hubzoned-owned>No</hubzoned-owned>
    <award-id>22280</award-id>
    <pi-name>Kenneth Abel</pi-name>
    <pi-ethnicity></pi-ethnicity>
    <p1top2></p1top2>
</Award>
```

Listing 3.3: Sample Document from SBIR-STTR

Problems and Errors

The SBIR-STTR data set, like most data sets, is not without problems. The dataset is plagued with duplicate awards, missing abstracts, and inconsistent reporting across agencies. When using this dataset for training classifiers, duplicates and missing abstracts need to be filtered out in order to maintain a high quality data set. In Chapter 5, we give a detailed overview of procedures for scrubbing out the bad data from the corpus in order to improve training accuracy.

3.2.3 Other Corpora

There is a plethora of semantically annotated corpora of varying quality and cost available on the web. A brief listing of just a few of the publicly available corpora is shown in Table 3.1. Each of these corpora can be use to train and test new document classification techniques and reinforce current implementations across disparate domains of interest.

In addition to publicly available corpora on the web, another approach is to artificially construct documents from a predefined set of features. This method will produce "documents" that can be used in training classifiers but will not necessarily represent natural language. Generating documents is purely mechanical and is only constrained by the available computing power and time constraints. This eliminates the need for domain experts, which can be costly. This method is valuable to researchers who are developing new classification algorithms and need a controlled sandbox to test and tune behaviors.

Name	Description	Source
New York Times	Annotated Corpus from 1987 to 2007	[41]
GENIA	A semantically annotated corpus for bio-textmining	[23]
Sentiment140	Twitter entries by product or brand	[11]
CLEF	Semantic annotation of clinical text	[38]
Common Crawl	Common Crawl web archive	[7]

Table 3.1: Public corpora for use in text classification

3.3 Annotation

An annotation, as defined by the Merriam-Webster dictionary, is a note added by way of comment or explanation [31]. Annotations can be viewed as metadata to a document that can be used to properly classify a document into its user-defined category. Annotations take a different meaning depending on the domain in question. For example, in the Linguistic domain, an annotation may designate a part of speech (POS) or named entity (NE). For the task of document classification, we designate an annotation simply as a tag attached to a chunk of text whose meaning is determined by the domain and classifier in use. For example, the Reuters RCV-1 corpus consists of XML encoded documents with annotations (tags) representing the category of the document. The SBIR-STTR corpus has annotations designating successful or unsuccessful grant abstracts.

Annotating a corpus is a substantial undertaking as it is typically created manually by domain experts. In order to get the maximum benefit from a machine learning algorithm, a clean, well-annotated corpus of documents is an absolute necessity. Creating a semantically annotated corpus is solely reliant on human labor and thus susceptible to whims and errors in judgment. Mitigating said errors with cogent procedures backed with solid software is the key to creating a gold standard corpus with which to train a classifier.

The cost to produce an annotated gold standard corpus is an important aspect that cannot be overlooked. Corpora such as the Reuters RCV-1 were exceptionally expensive to create [25] proving that accurately estimating the cost of collecting and labeling a data set is important. The cost of a labeled data set is derived from multiple sources. First, we have to look at the actual labor for domain experts and the time it takes them to apply a gold standard label to a document [43]. The cost per label may not be constant given the complexity of the domain in question or the experience of the annotators. Second, we have to look at the cost of improperly labeled documents and how they impact a system [28]. If we have enough documents that are improperly labeled, it is possible to degrade the precision and recall of a classifier to levels that are unusable.

Due to the expensive and time-consuming nature of creating corpora, there has been active research to help automate [21, 33] or supplement [44, 32] the process. While results from studies in automating or supplementing the annotation process have been promising, there will always been a need for a domain expert when dealing with natural language classification. At the very least, we must verify the generated or supplemented annotations.

3.3.1 Annotation Methods

Once a domain expert is ready to apply an annotation to a document that has been analyzed, a method for attaching the annotation to the source text must be devised. Unfortunately, there is no universally agreed upon method to correlate annotations with the source text and is generally domain dependent. For example, Listing 3.4 shows the POS tagged corpus from the GENIA project with in-line annotations placed in accordance with the Penn Treebank POS tagging scheme [45], while Listing 3.1 shows stand-off annotations developed by Reuters when constructing the RCV-1 dataset. Finally, Listing 3.5 shows a sample listing from the Sentiment140 corpus with annotations as numbers that correspond to the polarity of the text as shown in Table 3.2.

3.4 Document Storage and Encoding

Encoding of documents consists of two high-level concepts. First, we have to define how the characters of the document are encoded. Two common examples are the

Field	Meaning
0	the polarity of the tweet (1 = negative, 2 = neutral, 4 = positive)
1	the id of the tweet
2	the date of the tweet
3	the query. If there is no query, then this value is NO_QUERY.
4	the user that tweeted
5	the text of the tweet

Table 3.2: Sentiment140 corpus field key

American Standard Code for Information Interchange (ASCII) or Unicode Transformation Format (UTF). The encoding of the characters is important to note so that archived documents can be retrieved at a later date and their contents read. Second, we have to determine how the structure of the document will be represented, examples include Standard Generalized Markup Language (SGML) or XML.

While there have been attempts at standardizing the representation of text in digital form such as the Text Encoding Initiative [4], there is no universal encoding that is currently applied across all corpora. Most modern data sets follow an XML style formatting with older data sets using SGML, HTML, or a proprietary markup language. For example, Reuters uses XML in their RCV-1 and RCV-2 corpus with a custom Document Type Definition (DTD). The New York Times corpus uses National Imagery Transmission Formant (NITF) to encode their documents, which is XML based, and the Sentiment140 corpus uses a comma separated values format.

3.5 Training and Testing

Once the corpus has been annotated, the performance of a classifier trained with the applied annotations and text must be evaluated. The primary technique in the field of document classification is k-fold cross validation with a common value for $k = 10$. In

k-fold cross validation, a corpus is divided into 2 disjoint sets: a training set TR and a test set TS [42]. The corpus split is typically not even. For example, one may use a 90/10 or 70/30 split depending on the size and domain of the corpus in question. The classifier in question is then trained on the training set and evaluated on the test set. This process is repeated k times with the precision, recall, and F-Measure aggregated across all runs.

3.6 Summary

In this chapter, we reviewed the definition of what a corpora is in the domain of document classification. We introduced several standard corpora that are commonly used in the field to train and test new classifier implementations. Additionally, methods for correlating annotations with raw text in the form of in-line or stand-off annotations were reviewed with examples from existing corpora given. Most importantly, we introduced a new dataset, SBIR-STTR [1], that has been derived from publicly available data sources and automatically annotated for use by the research community.

[1]The SBIR-STTR corpus is freely available for download from trac.boisestate.edu

```
<article>
<articleinfo>
<bibliomisc>MEDLINE:95369245</bibliomisc>
</articleinfo>
<title>
<sentence><w c=”NN”>IL−2</w> <w c=”NN”>gene</w> <w c=”NN”>expression</w>
     <w c=”CC”>and</w> <w c=”NN”>NF−kappa</w> <w c=”NN”>B</w> <w c=”NN”>
    activation</w> <w c=”IN”>through</w> <w c=”NN”>CD28</w> <w c=”VBZ”>
    requires</w> <w c=”JJ”>reactive</w> <w c=”NN”>oxygen</w> <w c=”NN”>
    production</w> <w c=”IN”>by</w> <w c=”NN”>5−lipoxygenase</w><w c=”.”
    >.</w></sentence>
</title>
<abstract>
<sentence><w c=”NN”>Activation</w> <w c=”IN”>of</w> <w c=”DT”>the</w> <w
     c=”NN”>CD28</w> <w c=”NN”>surface</w> <w c=”NN”>receptor</w> <w c=”
    VBZ”>provides</w> <w c=”DT”>a</w> <w c=”JJ”>major</w> <w c=”JJ”>
    costimulatory</w> <w c=”NN”>signal</w> <w c=”IN”>for</w> <w c=”NN”>T
    </w> <w c=”NN”>cell</w> <w c=”NN”>activation</w> <w c=”VBG”>
    resulting</w> <w c=”IN”>in</w> <w c=”VBN”>enhanced</w> <w c=”NN”>
    production</w> <w c=”IN”>of</w> <w c=”NN”>interleukin−2</w> <w c=”(”
    >(</w><w c=”NN”>IL−2</w><w c=”)”>)</w> <w c=”CC”>and</w> <w c=”NN”>
    cell</w> <w c=”NN”>proliferation</w><w c=”.”>.</w></sentence>
<sentence><w c=”IN”>In</w> <w c=”JJ”>primary</w> <w c=”NN”>T</w> <w c=”
    NNS”>lymphocytes</w> <w c=”PRP”>we</w> <w c=”VBP”>show</w> <w c=”IN”
    >that</w> <w c=”NN”>CD28</w> <w c=”NN”>ligation</w> <w c=”VBZ”>leads
    </w> <w c=”TO”>to</w> <w c=”DT”>the</w> <w c=”JJ”>rapid</w> <w c=”JJ
    ”>intracellular</w> <w c=”NN”>formation</w> <w c=”IN”>of</w> <w c=”
    JJ”>reactive</w> <w c=”NN”>oxygen</w> <w c=”NNS”>intermediates</w> <
    w c=”(”>(</w><w c=”NNS”>ROIs</w><w c=”)”>)</w> <w c=”WDT”>which</w>
    <w c=”VBP”>are</w> <w c=”VBN”>required</w> <w c=”IN”>for</w> <w c=”∗
    ”>CD28</w><w c=”JJ”>−mediated</w> <w c=”NN”>activation</w> <w c=”IN”
    >of</w> <w c=”DT”>the</w> <w c=”NN”>NF−kappa</w> <w c=”∗”>B</w><w c=
    ”∗”>/</w><w c=”∗”>CD28</w><w c=”JJ”>−responsive</w> <w c=”NN”>
    complex</w> <w c=”CC”>and</w> <w c=”NN”>IL−2</w> <w c=”NN”>
    expression</w><w c=”.”>.</w></sentence>
<sentence><w c=”NN”>Delineation</w> <w c=”IN”>of</w> <w c=”DT”>the</w> <
    w c=”NN”>CD28</w> <w c=”NN”>signaling</w> <w c=”NN”>cascade</w> <w c
    =”VBD”>was</w> <w c=”VBN”>found</w> <w c=”TO”>to</w> <w c=”VB”>
    involve</w> <w c=”NN”>protein</w> <w c=”NN”>tyrosine</w> <w c=”NN”>
    kinase</w> <w c=”NN”>activity</w><w c=”,”>,</w> <w c=”VBN”>followed<
    /w> <w c=”IN”>by</w> <w c=”DT”>the</w> <w c=”NN”>activation</w> <w c
    =”IN”>of</w> <w c=”NN”>phospholipase</w> <w c=”NN”>A2</w> <w c=”CC”>
    and</w> <w c=”NN”>5−lipoxygenase</w><w c=”.”>.</w></sentence>
</abstract>
</article>
```

Listing 3.4: Sample Document from GENIA

```
"0","1467810369","Mon Apr 06 22:19:45 PDT 2009","NO_QUERY","_TheSpecialOne_","@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer. You shoulda got David Carr of Third Day to do it. ;D"
"0","1467810672","Mon Apr 06 22:19:49 PDT 2009","NO_QUERY","scotthamilton","is upset that he can't update his Facebook by texting it... and might cry as a result  School today also. Blah!"
"0","1467810917","Mon Apr 06 22:19:53 PDT 2009","NO_QUERY","mattycus","@Kenichan I dived many times for the ball. Managed to save 50%  The rest go out of bounds"
"4","2193427163","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","CrazySlutty","@markREED3 Awh, cool! "
"4","2193427181","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","yasminexO","@jonasbrothers - mr. president is a geeee . im off to take my history regents. wish me luck and ill love you forever "
"4","2193427199","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","MirandaJ","@Farcical Ah...  Book!  I'll watch Bogey's and let you know how it is!  "
"4","2193427224","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","Art_Advisor","@Donnette thanks! yeah like I mentioned, was fun  How is your day looking?"
"4","2193427276","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","_ThnksFrThMmrs_","Now Piglet's Dead. "
"4","2193427329","Tue Jun 16 08:26:39 PDT 2009","NO_QUERY","whipzilla","- had a great time with some of the best people last night/2day "
"4","2193427403","Tue Jun 16 08:26:40 PDT 2009","NO_QUERY","_MisterG","@dajbelshaw you'll be just round the corner from me!  Ok a Big Corner then "
"4","2193427413","Tue Jun 16 08:26:40 PDT 2009","NO_QUERY","AnneDouglas","@LorelieBrown Congrats on the safe return "
"4","2193427459","Tue Jun 16 08:26:40 PDT 2009","NO_QUERY","ChoongYH","@MeiTingT Haha, its just for fun, started recently only thou "
"4","2193427479","Tue Jun 16 08:26:40 PDT 2009","NO_QUERY","Nainx","I just found 2 cinema tickets from 06 and 07. Don't know why I still have them, but they brought back so many memories "
```

Listing 3.5: Sample Document from the Sentiment140 corpus

CHAPTER 4

TAT

4.1 Overview

TAT is a collection of independent components packaged together into one open source software application. TAT was engineered to support the document classification process and workflow. Tracking of changes in a working corpus, saving data used in the training of classifiers to ensure reproducibility, and providing a mechanism for interacting with copyright protected corpora are all fundamental issues that TAT addresses. TAT is built using the robust Open IDE [35] framework that allows plug-in developers access to standard well-tested libraries saving years of development time. The main goal of TAT is to minimize the labor intensive process of creating labelled data that can be used to train, test, and deploy machine learning models for automated text annotation. The toolkit can facilitate the annotation of text using different machine learning packages as well as corpora with different metadata specifications. This chapter gives a brief overview of the application and how it interacts with a classification engine and a given corpus.

4.2 Application Overview

There are several functions that the application can facilitate during the process of document classification. First, there is document annotation, which aids the user in adding both manual and automatic annotations. Section 4.2.1 gives a detailed overview of this functionality. Second, there is model training and testing, which allows you to select any supported classifier, a set of annotated documents, and produce trained classifiers. Section 4.2.2 drills down into this functionality. Finally, Section 4.2.3 reviews the built-in corpus change tracking functionality, which provides an easy way for a user to replicate past training results.

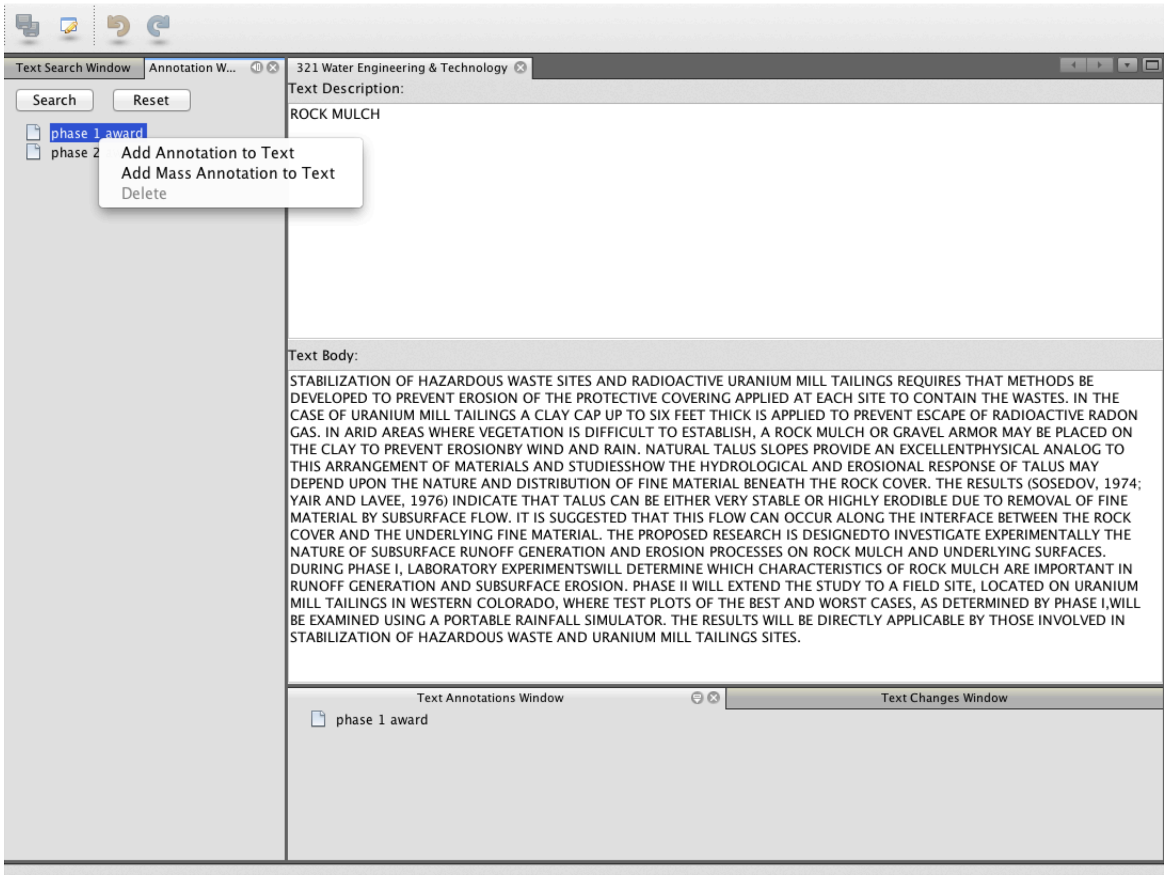
4.2.1 Annotation

When TAT is used for annotation, a user can open a set of records from a document repository specified by the corpus provider plug-in. Each record is subsequently loaded with its contents viewable in the TAT interface. The user can then assign class labels to the record simply by selecting from the annotations tab within the application as seen in Figure 4.1. Upon completion of annotating a record, the user can save the updated document with associated metadata, which can be used in training machine learning models. Annotation mode supports the editing of heterogeneous record sets with automatic annotation recognition. This allows the application to edit raw (unannotated) records in parallel with partial or complete annotated records.

In addition to directly applying a label to a document, TAT supports adding in-line annotations. As shown in Figure 4.2, a user can highlight a portion of text and apply an annotation. This action saves the start and end offset into the document, allowing the labeling of named entities or parts of speech. It is important to note that the original document is unchanged with in-line annotations, the metadata is simply an overlay that can be passed to the relevant classifier.

During the process of adding class labels to documents, the user also has the ability to modify the document text to remove any data that may be unnecessary, incorrect, or restricted by law. For example, if a user is working on creating a corpus of annotated medical records, it may be necessary to remove any data that could possibly be used to identify a patient in order to satisfy privacy laws.

Once a trained classifier has been produced, it is possible to use the classifier to assist the user in labeling new documents. When each document is loaded, the user



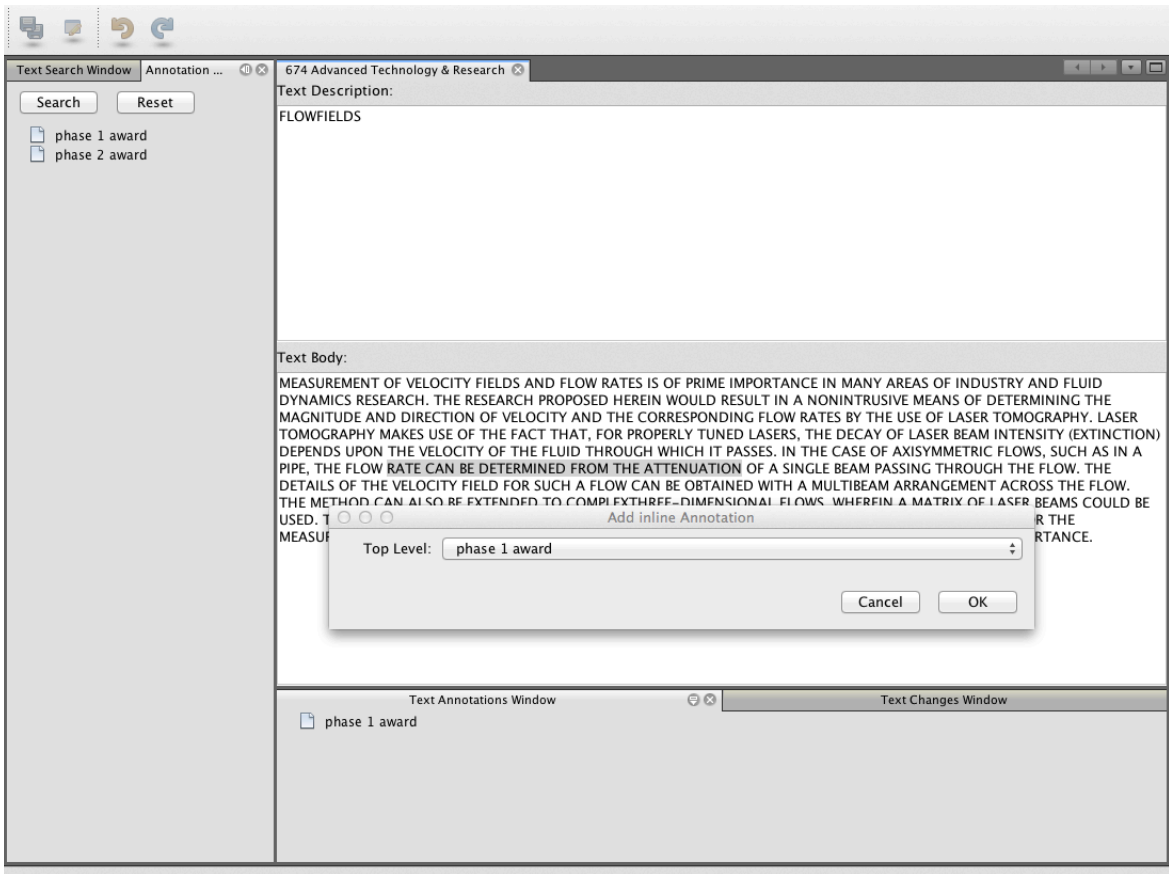
Text Description:

ROCK MULCH

Text Body:

STABILIZATION OF HAZARDOUS WASTE SITES AND RADIOACTIVE URANIUM MILL TAILINGS REQUIRES THAT METHODS BE DEVELOPED TO PREVENT EROSION OF THE PROTECTIVE COVERING APPLIED AT EACH SITE TO CONTAIN THE WASTES. IN THE CASE OF URANIUM MILL TAILINGS A CLAY CAP UP TO SIX FEET THICK IS APPLIED TO PREVENT ESCAPE OF RADIOACTIVE RADON GAS. IN ARID AREAS WHERE VEGETATION IS DIFFICULT TO ESTABLISH, A ROCK MULCH OR GRAVEL ARMOR MAY BE PLACED ON THE CLAY TO PREVENT EROSIONBY WIND AND RAIN. NATURAL TALUS SLOPES PROVIDE AN EXCELLENTPHYSICAL ANALOG TO THIS ARRANGEMENT OF MATERIALS AND STUDIESSHOW THE HYDROLOGICAL AND EROSIONAL RESPONSE OF TALUS MAY DEPEND UPON THE NATURE AND DISTRIBUTION OF FINE MATERIAL BENEATH THE ROCK COVER. THE RESULTS (SOSEDOV, 1974; YAIR AND LAVEE, 1976) INDICATE THAT TALUS CAN BE EITHER VERY STABLE OR HIGHLY ERODIBLE DUE TO REMOVAL OF FINE MATERIAL BY SUBSURFACE FLOW. IT IS SUGGESTED THAT THIS FLOW CAN OCCUR ALONG THE INTERFACE BETWEEN THE ROCK COVER AND THE UNDERLYING FINE MATERIAL. THE PROPOSED RESEARCH IS DESIGNEDTO INVESTIGATE EXPERIMENTALLY THE NATURE OF SUBSURFACE RUNOFF GENERATION AND EROSION PROCESSES ON ROCK MULCH AND UNDERLYING SURFACES. DURING PHASE I, LABORATORY EXPERIMENTSWILL DETERMINE WHICH CHARACTERISTICS OF ROCK MULCH ARE IMPORTANT IN RUNOFF GENERATION AND SUBSURFACE EROSION. PHASE II WILL EXTEND THE STUDY TO A FIELD SITE, LOCATED ON URANIUM MILL TAILINGS IN WESTERN COLORADO, WHERE TEST PLOTS OF THE BEST AND WORST CASES, AS DETERMINED BY PHASE I,WILL BE EXAMINED USING A PORTABLE RAINFALL SIMULATOR. THE RESULTS WILL BE DIRECTLY APPLICABLE BY THOSE INVOLVED IN STABILIZATION OF HAZARDOUS WASTE AND URANIUM MILL TAILINGS SITES.

Figure 4.1: Manual annotation in TAT: After reading a document, shown in the upper-right panel, the user can select an annotation to apply to the text, shown in the left-hand panel. All current annotations that are applied to the document are shown in the bottom panel. This example shows the user applying a label called "phase 1 award" to a document loaded from the SBIR-STTR corpus. This label is being applied as a stand off annotation with no offset information into the document.



Text Description:

FLOWFIELDS

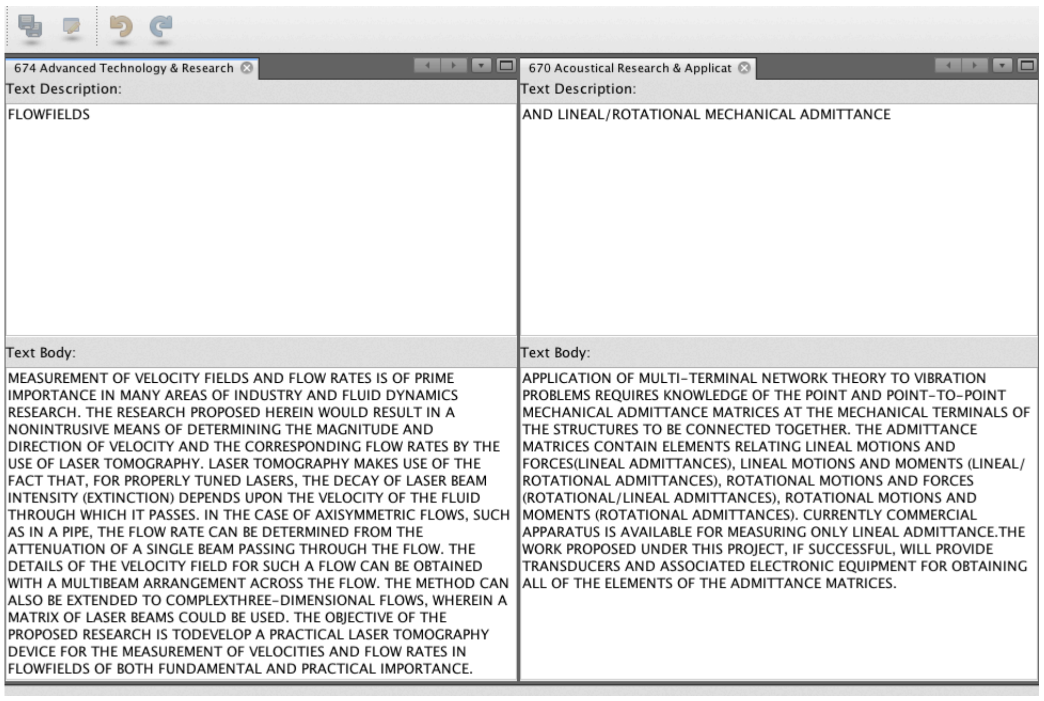
Text Body:

MEASUREMENT OF VELOCITY FIELDS AND FLOW RATES IS OF PRIME IMPORTANCE IN MANY AREAS OF INDUSTRY AND FLUID DYNAMICS RESEARCH. THE RESEARCH PROPOSED HEREIN WOULD RESULT IN A NONINTRUSIVE MEANS OF DETERMINING THE MAGNITUDE AND DIRECTION OF VELOCITY AND THE CORRESPONDING FLOW RATES BY THE USE OF LASER TOMOGRAPHY. LASER TOMOGRAPHY MAKES USE OF THE FACT THAT, FOR PROPERLY TUNED LASERS, THE DECAY OF LASER BEAM INTENSITY (EXTINCTION) DEPENDS UPON THE VELOCITY OF THE FLUID THROUGH WHICH IT PASSES. IN THE CASE OF AXISYMMETRIC FLOWS, SUCH AS IN A PIPE, THE FLOW RATE CAN BE DETERMINED FROM THE ATTENUATION OF A SINGLE BEAM PASSING THROUGH THE FLOW. THE DETAILS OF THE VELOCITY FIELD FOR SUCH A FLOW CAN BE OBTAINED WITH A MULTIBEAM ARRANGEMENT ACROSS THE FLOW. THE METHOD CAN ALSO BE EXTENDED TO COMPLEXTHREE-DIMENSIONAL FLOWS, WHEREIN A MATRIX OF LASER BEAMS COULD BE USED. T R THE MEASU RTANCE.

Add inline Annotation

Top Level: phase 1 award

Figure 4.2: In-line annotations in TAT: In-line annotations allow a user to apply a label to any highlighted portion of text. This example shows the user applying a label directly to the text in the document. This allows users to apply labels to name entities or parts of speech if desired.


674 Advanced Technology & Research
670 Acoustical Research & Applicat

Text Description:

FLOWFIELDS

Text Body:

MEASUREMENT OF VELOCITY FIELDS AND FLOW RATES IS OF PRIME IMPORTANCE IN MANY AREAS OF INDUSTRY AND FLUID DYNAMICS RESEARCH. THE RESEARCH PROPOSED HEREIN WOULD RESULT IN A NONINTRUSIVE MEANS OF DETERMINING THE MAGNITUDE AND DIRECTION OF VELOCITY AND THE CORRESPONDING FLOW RATES BY THE USE OF LASER TOMOGRAPHY. LASER TOMOGRAPHY MAKES USE OF THE FACT THAT, FOR PROPERLY TUNED LASERS, THE DECAY OF LASER BEAM INTENSITY (EXTINCTION) DEPENDS UPON THE VELOCITY OF THE FLUID THROUGH WHICH IT PASSES. IN THE CASE OF AXISYMMETRIC FLOWS, SUCH AS IN A PIPE, THE FLOW RATE CAN BE DETERMINED FROM THE ATTENUATION OF A SINGLE BEAM PASSING THROUGH THE FLOW. THE DETAILS OF THE VELOCITY FIELD FOR SUCH A FLOW CAN BE OBTAINED WITH A MULTIBEAM ARRANGEMENT ACROSS THE FLOW. THE METHOD CAN ALSO BE EXTENDED TO COMPLEXTHREE-DIMENSIONAL FLOWS, WHEREIN A MATRIX OF LASER BEAMS COULD BE USED. THE OBJECTIVE OF THE PROPOSED RESEARCH IS TODEVELOP A PRACTICAL LASER TOMOGRAPHY DEVICE FOR THE MEASUREMENT OF VELOCITIES AND FLOW RATES IN FLOWFIELDS OF BOTH FUNDAMENTAL AND PRACTICAL IMPORTANCE.

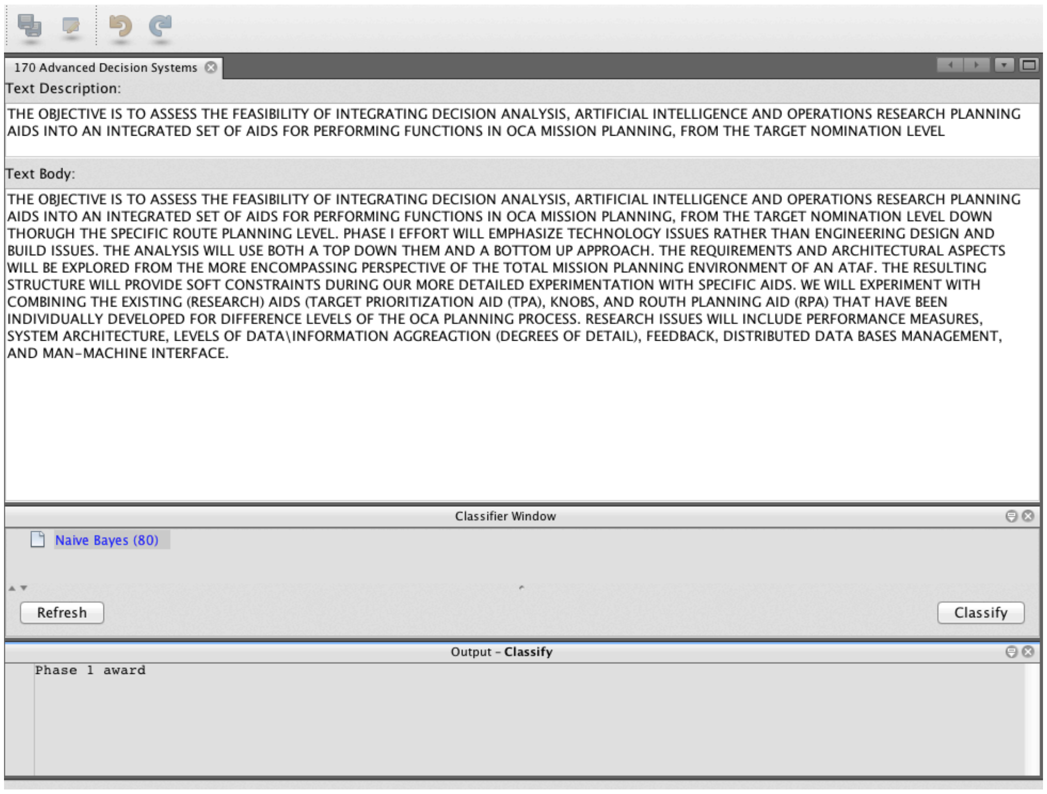
Text Description:

AND LINEAL/ROTATIONAL MECHANICAL ADMITTANCE

Text Body:

APPLICATION OF MULTI-TERMINAL NETWORK THEORY TO VIBRATION PROBLEMS REQUIRES KNOWLEDGE OF THE POINT AND POINT-TO-POINT MECHANICAL ADMITTANCE MATRICES AT THE MECHANICAL TERMINALS OF THE STRUCTURES TO BE CONNECTED TOGETHER. THE ADMITTANCE MATRICES CONTAIN ELEMENTS RELATING LINEAL MOTIONS AND FORCES(LINEAL ADMITTANCES), LINEAL MOTIONS AND MOMENTS (LINEAL/ ROTATIONAL ADMITTANCES), ROTATIONAL MOTIONS AND FORCES (ROTATIONAL/LINEAL ADMITTANCES), ROTATIONAL MOTIONS AND MOMENTS (ROTATIONAL ADMITTANCES). CURRENTLY COMMERCIAL APPARATUS IS AVAILABLE FOR MEASURING ONLY LINEAL ADMITTANCE.THE WORK PROPOSED UNDER THIS PROJECT, IF SUCCESSFUL, WILL PROVIDE TRANSDUCERS AND ASSOCIATED ELECTRONIC EQUIPMENT FOR OBTAINING ALL OF THE ELEMENTS OF THE ADMITTANCE MATRICES.

Figure 4.3: Editing multiple documents simultaneously: This gives the user the ability to do comparison edits. For example, it is possible to load all the documents that were annotated by a specific user and review them side-by-side to ensure that labels have been applied consistently across the corpus.



Text Description:

THE OBJECTIVE IS TO ASSESS THE FEASIBILITY OF INTEGRATING DECISION ANALYSIS, ARTIFICIAL INTELLIGENCE AND OPERATIONS RESEARCH PLANNING AIDS INTO AN INTEGRATED SET OF AIDS FOR PERFORMING FUNCTIONS IN OCA MISSION PLANNING, FROM THE TARGET NOMINATION LEVEL

Text Body:

THE OBJECTIVE IS TO ASSESS THE FEASIBILITY OF INTEGRATING DECISION ANALYSIS, ARTIFICIAL INTELLIGENCE AND OPERATIONS RESEARCH PLANNING AIDS INTO AN INTEGRATED SET OF AIDS FOR PERFORMING FUNCTIONS IN OCA MISSION PLANNING, FROM THE TARGET NOMINATION LEVEL DOWN THORUGH THE SPECIFIC ROUTE PLANNING LEVEL. PHASE I EFFORT WILL EMPHASIZE TECHNOLOGY ISSUES RATHER THAN ENGINEERING DESIGN AND BUILD ISSUES. THE ANALYSIS WILL USE BOTH A TOP DOWN THEM AND A BOTTOM UP APPROACH. THE REQUIREMENTS AND ARCHITECTURAL ASPECTS WILL BE EXPLORED FROM THE MORE ENCOMPASSING PERSPECTIVE OF THE TOTAL MISSION PLANNING ENVIRONMENT OF AN ATAF. THE RESULTING STRUCTURE WILL PROVIDE SOFT CONSTRAINTS DURING OUR MORE DETAILED EXPERIMENTATION WITH SPECIFIC AIDS. WE WILL EXPERIMENT WITH COMBINING THE EXISTING (RESEARCH) AIDS (TARGET PRIORITIZATION AID (TPA), KNOBS, AND ROUTH PLANNING AID (RPA) THAT HAVE BEEN INDIVIDUALLY DEVELOPED FOR DIFFERENCE LEVELS OF THE OCA PLANNING PROCESS. RESEARCH ISSUES WILL INCLUDE PERFORMANCE MEASURES, SYSTEM ARCHITECTURE, LEVELS OF DATA\INFORMATION AGGREAGTION (DEGREES OF DETAIL), FEEDBACK, DISTRIBUTED DATA BASES MANAGEMENT, AND MAN-MACHINE INTERFACE.

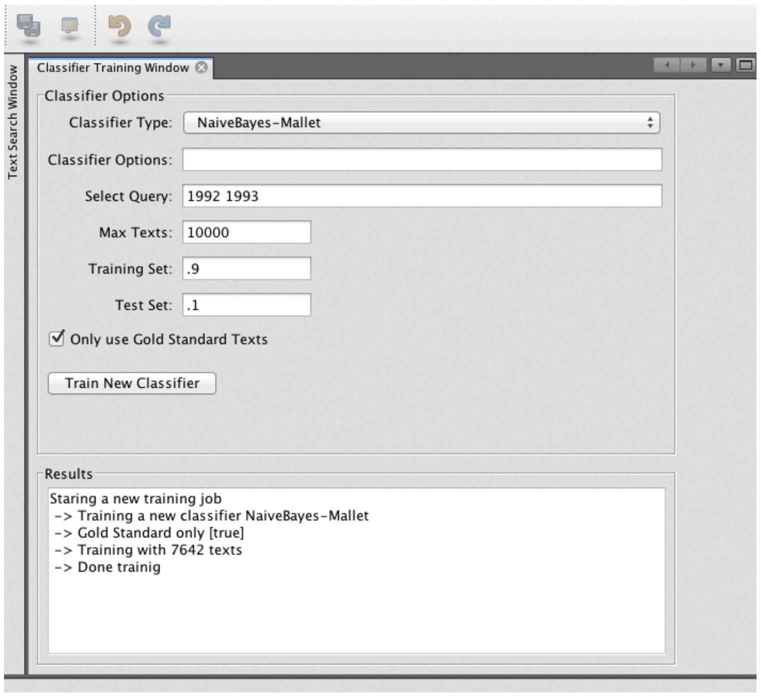
```
Phase 1 award
```

Figure 4.4: Suggested annotations in TAT: When annotating a document, the user can request that a classifier trained on the corpus in question suggest a label to apply. The figure above shows that a suggested label of “phase 1 award” for the currently loaded text.

can view a list of suggested annotations that the selected classifier produces as seen in Figure 4.4. Thus, the user can accept the suggestions and move on to the next document or manually apply a label to improve classifier accuracy.

4.2.2 Model Training and Testing

In model Training and Testing mode, the application allows the user to specify parameters on the loaded classifier and to select a set of annotated files to be used for training, testing, and validation. The user can use TAT to train a machine learning



Classifier Options

Classifier Type: NaiveBayes-Mallet

Classifier Options:

Select Query: 1992 1993

Max Texts: 10000

Training Set: .9

Test Set: .1

☑ Only use Gold Standard Texts

Results

```
Staring a new training job
 -> Training a new classifier NaiveBayes-Mallet
 -> Gold Standard only [true]
 -> Training with 7642 texts
 -> Done trainig
```

Figure 4.5: Training a classifier in TAT: Training a new Naive Bayes classifier from the Mallet plug-in using texts from the years 1992 and 1993 with 90% of the documents going to the training set and 10% of the documents going to the test set.

model on the selected files. This mode also allows the user to load previously trained models for testing (or further training and testing) on selected data. TAT calculates various accuracy measures as described in Section 2.5.3, such as precision, recall, and F-Measure scores, and saves these statistics as metadata. Figure 4.5 shows the training and testing interface as presented to the user.

4.2.3 Freeze Dry and Change Tracking

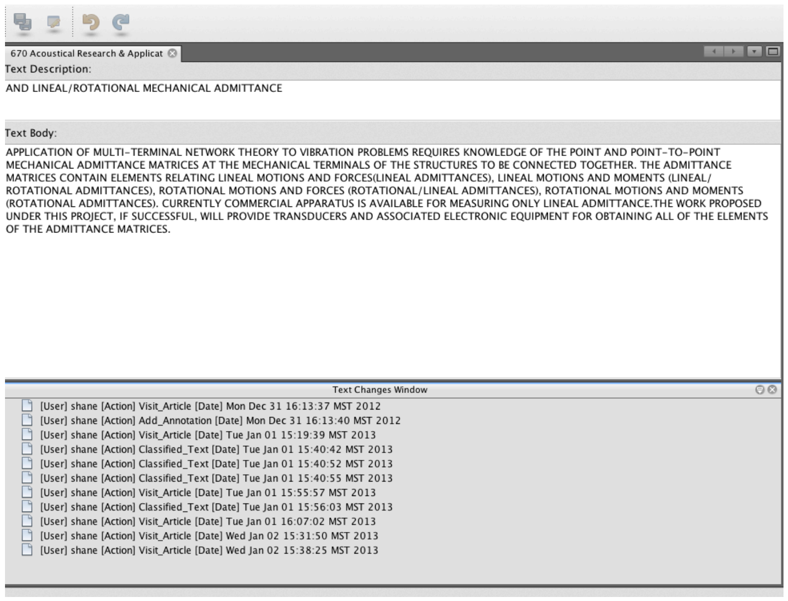
Tracking changes within the corpus is just as critical as tracking changes to source code. As a corpus (or code base) grows, it becomes very difficult to monitor all changes that are committed each day. Thus, it becomes critical to be able to look through the change history to be able to identify events that may have impacted the quality of trained classifiers. Take for example a system that is continuously training and deploying trained classifiers in a production environment. It is safe to assume that such a system would have metrics in place to ensure that the quality of newly trained classifiers does not drop below a certain predefined threshold. If a change to the corpus is made that causes the quality of newly trained classifiers to drop to unacceptable levels, a user can look through the recent history to identify the change that caused the drop.

During the process of annotating a corpus, tracking all changes that are made is important so training results can be replicated. The core framework and classifier plug-in determine the relevant information that needs to be saved in order to reproduce the results. Information can include the particular classification algorithm and version used, such as Naive Bayes or SVM, the set of documents that were used to train, test, and validate the classifier, random seeds used, and any parameter settings needed.

The freeze dry feature of TAT is implemented by leveraging the change tracking feature provided by the core framework. Each change in the form of an annotation, modification, or document visit is logged by the system. Figure 4.6 shows the editing history of a document that is saved. Having a robust history allows the user to find any actions that may have had a impact on the working corpus.

4.3 Application Architecture

TAT is built using a component-based setup and leverages the Open IDE framework [35]. Key functionality is implemented as separate modules so the user will be



Text Description:

AND LINEAL/ROTATIONAL MECHANICAL ADMITTANCE

Text Body:

APPLICATION OF MULTI-TERMINAL NETWORK THEORY TO VIBRATION PROBLEMS REQUIRES KNOWLEDGE OF THE POINT AND POINT-TO-POINT MECHANICAL ADMITTANCE MATRICES AT THE MECHANICAL TERMINALS OF THE STRUCTURES TO BE CONNECTED TOGETHER. THE ADMITTANCE MATRICES CONTAIN ELEMENTS RELATING LINEAL MOTIONS AND FORCES(LINEAL ADMITTANCES), LINEAL MOTIONS AND MOMENTS (LINEAL/ROTATIONAL ADMITTANCES), ROTATIONAL MOTIONS AND FORCES (ROTATIONAL/LINEAL ADMITTANCES), ROTATIONAL MOTIONS AND MOMENTS (ROTATIONAL ADMITTANCES). CURRENTLY COMMERCIAL APPARATUS IS AVAILABLE FOR MEASURING ONLY LINEAL ADMITTANCE.THE WORK PROPOSED UNDER THIS PROJECT, IF SUCCESSFUL, WILL PROVIDE TRANSDUCERS AND ASSOCIATED ELECTRONIC EQUIPMENT FOR OBTAINING ALL OF THE ELEMENTS OF THE ADMITTANCE MATRICES.

```
[User] shane [Action] Visit_Article [Date] Mon Dec 31 16:13:37 MST 2012
[User] shane [Action] Add_Annotation [Date] Mon Dec 31 16:13:40 MST 2012
[User] shane [Action] Visit_Article [Date] Tue Jan 01 15:19:39 MST 2013
[User] shane [Action] Classified_Text [Date] Tue Jan 01 15:40:42 MST 2013
[User] shane [Action] Classified_Text [Date] Tue Jan 01 15:40:52 MST 2013
[User] shane [Action] Classified_Text [Date] Tue Jan 01 15:40:55 MST 2013
[User] shane [Action] Visit_Article [Date] Tue Jan 01 15:55:57 MST 2013
[User] shane [Action] Classified_Text [Date] Tue Jan 01 15:56:03 MST 2013
[User] shane [Action] Visit_Article [Date] Tue Jan 01 16:07:02 MST 2013
[User] shane [Action] Visit_Article [Date] Wed Jan 02 15:31:50 MST 2013
[User] shane [Action] Visit_Article [Date] Wed Jan 02 15:38:25 MST 2013
```

Figure 4.6: The annotation history on a document: The history on a document is shown in the bottom window. The loaded document has been visited, had an annotation added, and had been used in the training of a classifier.

able to support a wide range of scenarios. For instance, if the user needs to access text records located on disk, a plug-in to accomplish this task can be selected. If on the other hand the user needs to load records that reside in a database such as MySQL, a different plug-in can be selected. Out of the box, plug-ins for TAT are provided to interface with the SBIR-STTR corpus, annotations extracted from the SBIR-STTR dataset, and MALLET [29]. There are three extension points that offer a pluggable interface, the classifier, corpus, and change tracking modules. Each extension point is defined as a Java interface that must be implemented and placed into the appropriate service provider. A complete listing of the necessary interfaces are provided in Appendix B. The following sections give a high-level overview of each plug-in module and the default setup.

Using an interface setup within each plug-in allows TAT to maintain maximum flexibility when working with different corpora and classifiers. Each interface provides a buffer between the UI element of TAT and the data elements. For example, existing corpora do not have to be “imported” into TAT as long as the corpus plug-in knows how to read the existing format. TAT can display the information by querying the interface. This allows data to be portable with legacy systems or meet a specific data archive policy. In addition to abstracting the source and format of the data, TAT does not assume to know the best format needed to maximize performance. Leaving the performance requirements in the domain of the plug-in assures that the users can take advantage of any tweaks that TAT cannot predict. For example, companies such as Reuters have a very large system already in place to manage documents. Leveraging these systems allows TAT to eliminate expensive data export and import work.

4.3.1 Classifier Plug-in

Any new classifier can be plugged in by implementing the IClassifier and IClassification interface. Listing B.4 and B.5 in Appendix B shows all the methods needed to add a new classifier into TAT. A general high-level description of all the relevant methods are listed below and visualized in Figure 4.7.

- IClassifier
  - Accuracy - The self-reported accuracy of the classifier
  - Date - The date the classifier was last trained
  - F-Meassure - The self-reported F-measure score
  - Name - The name of the classifier. A good choice for the name includes the algorithm + corpus that was used to train

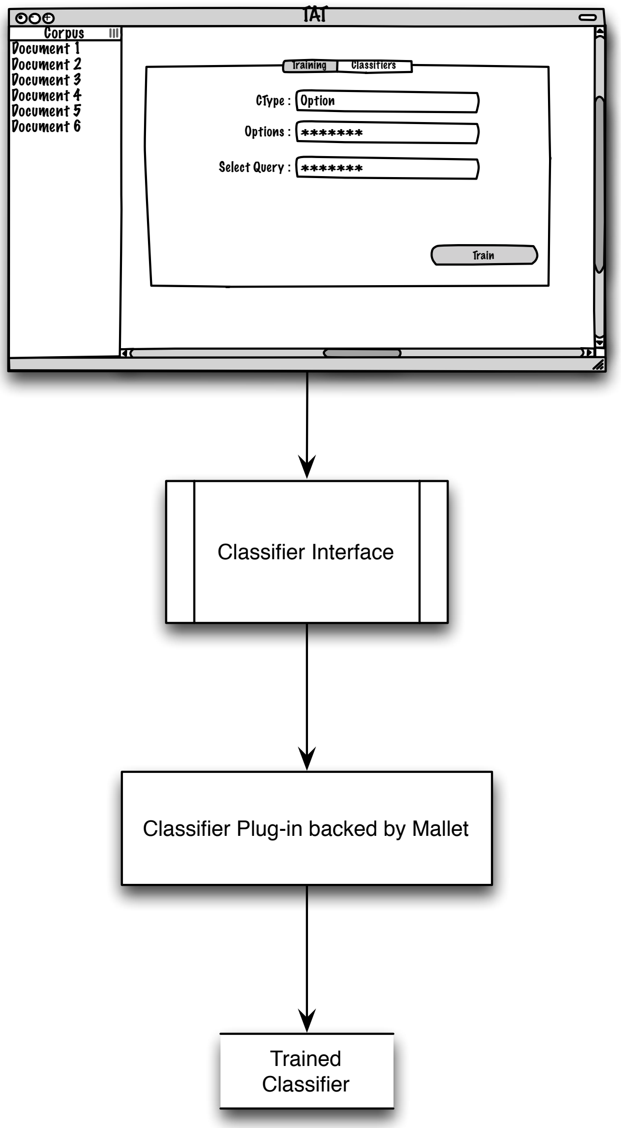


Figure 4.7: This sketch gives a high-level view of the relationship between the Classifier UI, Classifier Plug-in, and backing implementation. The Classifier plug-in serves as a buffer between the UI portion of the application and the actual classifier being used.

  - ID - The ID of the classifier. This field is used for saving the classifier and indexing. This should be unique and is dependent on how the classifier is saved (i.e., saved to a database or disk)
  - Other - Additional information specific to this classifier
  - Precision - The self-report precision of the classifier
  - Recall - The self-reported recall of the classifier

- IClassification
  - ClassifiyTexts - Given a set of texts and a classifier classify each text with the selected classifier and display the most likely tag.
  - Classifiers - Get all current trained classifiers
  - TrainClassifiers - Given a set of texts train a new classifier
  - RemoveClassifier - Delete a classifier out of the database
  - SupportedClassifiers - Returns a list of supported classifiers

Mallet has a very well developed API that makes integration with TAT painless and straight forward. First, the IClassification interface is implemented and configured to return the supported classifiers. The UI queries this interface and presents the list to the user. Once the user has set all the required parameters, the information is sent down to Mallet through the TrainClassifiers API call. Once the TrainClassifiers call has been made, we construct a new instance of the desired algorithm, build a training pipeline, and feed all selected documents to the classifier. Upon completion, we take the resulting classifier and wrap it in an object that implements the IClassifier and pass it back to the UI for the user to interact with.

4.3.2 Corpus Plug-in

When working with existing corpora, it is possible to run into copyright issues that prevent the data from being distributed with any results generated. This is the case with the Reuters RCV-1 corpus. Any researcher that obtains this corpus is, by law, not allowed to redistribute the raw data. Thus, it may be difficult for any other researcher to reproduce any results from a description alone. TAT addresses this problem by allowing researchers to build plug-ins for any copyrighted corpora and distribute the corpora specific plug-ins. By distributing the plug-ins for TAT, any other researcher that can obtain the original dataset can simply load the plug-in for the copyright protected corpus and reproduce the results reported.

The corpus plug-in provides an abstract view of the text used to train new classifiers and text that needs to be classified. A new Corpus can be added by implementing the ICorpus, IText, and ITextFacet interfaces. Listing B.1, B.2, and B.3 in Appendix B shows all the methods needed to add a new corpus into TAT. A general high-level description of all the relevant methods are listed below and visualized in Figure 4.8.

- ICorpus
  - Find - Given a user-defined query find all relevant texts
  - RemoveText - Remove a text from the corpus
  - UpdateText - Update an existing text
  - TextsForUser - Retrieve all texts that were annotated by a specific user
  - TextsForAnnotations - Retrieve all texts with the specified annotation
  - BuildSearchIndex- Build or rebuild the search index (Optional).

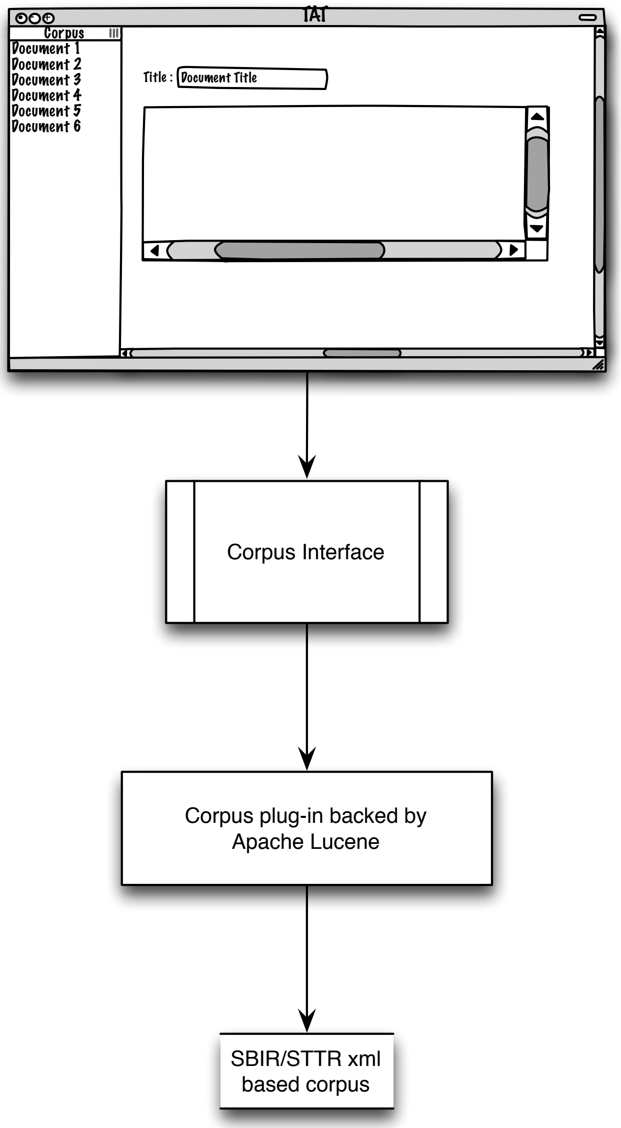


Figure 4.8: This sketch gives a high-level view of the relationship between the Corpus UI, Corpus Plug-in, and backing implementation. The Corpus plug-in serves as a buffer between the UI portion of the application and the backing dataset. This allows users to abstract the actual storage method used for the corpus.

- IText
  - Date - The date of the text
  - Desc - The description of the text
  - GoldStandard - Has this article been marked as a gold standard
  - Id - A unique ID typically used as an index into a database
  - Text - The body of the document
  - Title - The title of the text
- ITextFacet
  - Id - The id of the text facet
  - Property - A property of the facet
  - Object - An extension point for customization

4.3.3 Freeze dry and Change Tracking plug-in

The Freeze dry and Change tracking plug-in provides a method to track changes to the corpus over time. A new change tracking module can be added by implementing IEvent, IUser, and ITappCore interfaces. The storage method of the change is entirely plug-in dependent. The change can be stored as stand off annotations directly to the affected text or in a separate database and linked back to the text in question. Listing B.6, B.7, and B.8 in Appendix B shows all the methods needed to add a new change tracking module into TAT. A general high-level description of all the relevant methods are listed below.

The dataset used to train and test a classifier can have a significant impact on classifier precision and recall. There have been several attempts at establishing a standard dataset for researches to use such as the ModApte split of the Reuters-21578 document collection or the LYRL2004 split from the RCV-1 collection. However, there has not been universal adaptation of these datasets within the community. Researchers have only loosely followed the training and test splits [9, 36], randomly generated new ones [2, 39], only reported on the most frequent classes [1], or generated an entirely new custom dataset [3]. Using TAT, researchers can distribute the data that was generated during the creating of the training and test splits to allow results to be reproduced.

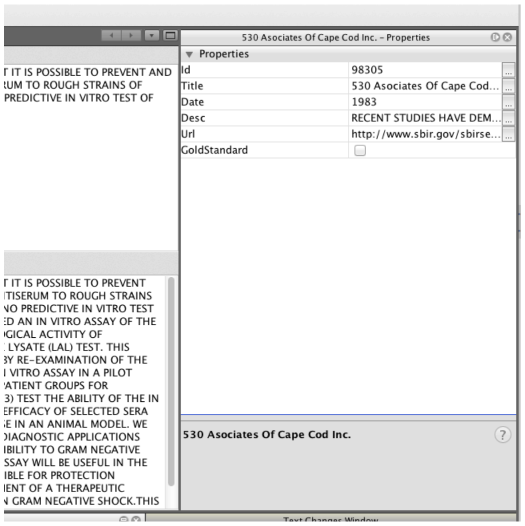
- IEvent
    - Id - The id of this event, typically a unique identifier
    - User - The user associated with this event
    - Date - The date this event happened
    - OldValue - The old value of a change that was made
    - TextId - The unique id of the affected text
    - UserAction - What the user did
    - TextStart - Numerical offset into the document where the change started
    - TextEnd - Numerical offset into the document where the change ended
- IUser
    - Id - The id of this user, typically a unique identifier
    - LoginName - The login name of the user
    - RealName - The real name of the user

  - Password - The users password
  - Date - The date of the text

- ITappCore
  - CreateUser - Adds a new user into the system
  - LookupUser - Find a user
  - DeleteUser - Remove a user from the system
  - EventsForText - Retrieve all the events that exist for the selected Text
  - AddEvent - Add a new event to the system
  - DeleteEvent - Remove an event from the system
  - EventsForUser - Retrieve all relevant events for a specified user

4.3.4 Properties Window

Each plug-in has access to the Properties window of TAT. This allows the user to select a Text, Classifier, or Annotation object within the interface to see any exposed properties of the object. The properties window leverages the Java reflection framework and a list of properties that the plug-in author wants to expose to extract the values. This feature is especially important to the corpus plug-in as it allows a unified approach to viewing attributes for a text object that would otherwise be inaccessible through the GUI. Figure 4.9 shows custom properties as extracted from a Text object.



530 Asociates Of Cape Cod Inc. – Properties

Properties

Id	98305
Title	530 Asociates Of Cape Cod...
Date	1983
Desc	RECENT STUDIES HAVE DEM...
Url	http://www.sbir.gov/sbirse...
GoldStandard	☐

530 Asociates Of Cape Cod Inc.

```
T IT IS POSSIBLE TO PREVENT AND
UM TO ROUGH STRAINS OF
PREDICTIVE IN VITRO TEST OF
```

```
T IT IS POSSIBLE TO PREVENT
TISERUM TO ROUGH STRAINS
NO PREDICTIVE IN VITRO TEST
D AN IN VITRO ASSAY OF THE
GICAL ACTIVITY OF
LYSATE (LAL) TEST. THIS
Y RE-EXAMINATION OF THE
VITRO ASSAY IN A PILOT
ATIENT GROUPS FOR
3) TEST THE ABILITY OF THE IN
EFFICACY OF SELECTED SERA
E IN AN ANIMAL MODEL. WE
DIAGNOSTIC APPLICATIONS
IBILITY TO GRAM NEGATIVE
SSAY WILL BE USEFUL IN THE
IBLE FOR PROTECTION
IENT OF A THERAPEUTIC
GRAM NEGATIVE SHOCK.THIS
```

Figure 4.9: The properties window: Showing properties extracted from a IText object

4.3.5 Licensing of TAT

There are hundreds of licenses available to the general public, we reviewed a few prominent ones in Chapter 1. If you take into account the fact that an individual or organization can write his or her own custom license, then the set of licenses available is infinite. By selecting a license that is backed by a well-known organization, it is possible to get a robust license that meets our needs and keeps costs low. Generally speaking, if the developer of a piece of software wants to ensure that the work and all derivative works stay out of proprietary software, then the best license to use would be the GNU GPL license. However, in the case of TAT, we do not want to restrict usage, as any commercial adoption can help further the development and plug-in ecosystem. Thus, TAT will be licensed under the LGPL to allow the greatest degree of flexibility among potential users.

4.4 Summary

With TAT's modular design, plug-ins can be provided for any document classification interface. Currently, plug-ins are provided for the SBIR-STTR dataset and a classification engine. Given the generic nature of the plug-in system, extending TAT to include other packages should be relatively straight forward for any experienced developer. Doing so will improve the usefulness and robustness of the application and help to further document classification and machine learning research.

CHAPTER 5

RESULTS

5.1 Overview

We can now address the research questions outlined in Chapter 1. We are interested in determining the likelihood that a phase I proposal will be selected to move to phase II based on features that are publicly available. We look at the amount of money that the award requests, the properties of the proposal such as woman, minority, and HUBZoned owned, the surname origin of the primary investigator, and the text of the proposals abstract. Section 5.2 details the work necessary to tune up the SBIR-STTR corpus downloaded from `sbir.gov`. Section 5.3 outlines the characteristics of the tuned corpus that was built. Section 5.4 details the experiments that were executed against the tuned corpus. Finally, Section 5.5 summarizes the results from the experiments.

5.2 Corpus Preparation

This section details the process of cleaning up the raw SBIR-STTR corpus as downloaded from `sbir.gov` and identify problems that must be corrected for accurate analysis. First, we evaluate the abstract for content and filter out awards below a given threshold. Second, we look for duplicate or near duplicate awards present in the dataset. Third, we evaluate the award metadata to identify and eliminate discrepancies. Finally, we detail the process of labeling the positive and negative examples. Section 5.3 lists the statistics for the final corpus.

5.2.1 An Overview of the Raw Data

The raw SBIR-STTR corpus contains 120,584 awards from the years 1983-2009 with 87,800 phase I awards and 32,784 phase II awards. The corpus is organized by year

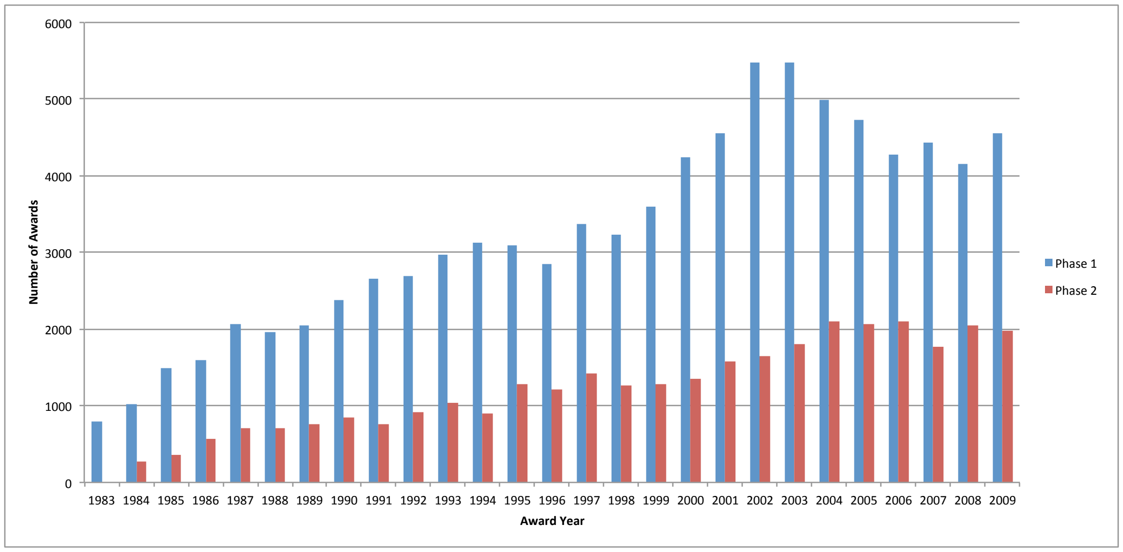


Figure 5.1: SBIR-STTR raw corpus breakdown by year. Each year shows the number of Phase I and Phase II awards that were granted across all eligible agencies. An analysis of the raw data is given in 5.2.1.

and saved in the format detailed in Chapter 3. Figure 5.1 gives a detailed break down by year. Table 5.1 gives a breakdown of the awards in the raw corpus by the category of the award. As we will see in the following sections, not all of the awards are usable in the final dataset.

5.2.2 Award Abstracts

As noted in Chapter 3, the raw SBIR-STTR corpus has a large number of awards with missing abstracts. Additionally, these awards have vague and uninformative titles. Using the search query of “N/A,” we found 23,164 individual awards that need to be filtered out because of missing abstracts. Listing 5.1 shows an example of one of the awards that was filtered. We can see that there is little we can gleam from an award with a missing abstract and a vague title of “A COMPUTER SYSTEM FOR THE MOLECULAR BIOLOGY LABORATORY.”

Category of Award	Number present
Duplicate phase I	780
Additional phase II award	316
Missing award Id	36
Potential usable awards	119452

Table 5.1: SBIR-STTR raw corpus breakdown by category. Duplicate and additional phase II awards are defined in Section 5.2.3. Missing award Id's are explained in Section 5.2.4.

```
<id>148283</id>
<title>A COMPUTER SYSTEM FOR THE MOLECULAR BIOLOGY LABORATORY</title>
<link>http://www.sbir.gov/sbirsearch/detail/148283</link>
<abstract>N/A</abstract>
<agency>HHS</agency>
<program>SBIR</program>
<phase>1</phase>
<year>1983</year>
<company>Dna Star</company>
<ri/>
<agency-tracking>523</agency-tracking>
<topic-code>N/A</topic-code>
<woman-owned>No</woman-owned>
<minority-owned>No</minority-owned>
<hubzoned-owned>No</hubzoned-owned>
<award-id>523</award-id>
```

Listing 5.1: Example of a document with a missing abstract in the SBIR-STTR corpus

5.2.3 Identifying Duplicate Awards

The SBIR-STTR program has been in operation for 29 years. Over time, duplicate awards have found their way into the corpus. A duplicate award is defined as a phase I award with all the same data except the `sbir.gov` ID of the award. Listings 5.2 and 5.3 show an example of a phase I award that has been flagged as a duplicate in the SBIR-STTR corpus. In the raw corpus, there were a total of 780 duplicate awards found and removed.

In addition to duplicate phase I awards, it is possible to have one additional phase II award under the same award-id due to Section 5111 of the National defense authorization act, which governs the SBIR/STTR process. The additional phase II award is only to be used for continued work on the current project not a new project. There were 316 of these awards found and removed.

```
<Award>
    <id>208656</id>
    <title>Simulated Job Performance Assessment</title>
    <link>http://www.sbir.gov/sbirsearch/detail/208656</link>
    <abstract>JPS is designing and developing a standard and affordable
        set of procedures for creating and validating PC-based job
        simulations that can be applied across a broad range of Army MOS
         to assess Soldier job performance.  As part of the Phase I
        effort , JPS is building a simulation for one Army MOS.  In this
        effort JPS is developing a 10 step process for creation of the
        simulations.</abstract>
    <agency>ARMY</agency>
    <program>SBIR</program>
    <phase>1</phase>
    <year>2007</year>
    <company>JOB PERFORMANCE SYSTEMS, INC.</company>
    <ri></ri>
    <agency-tracking>A072-048-0409</agency-tracking>
    <topic-code>N/A</topic-code>
    <woman-owned>No</woman-owned>
    <minority-owned>No</minority-owned>
    <hubzoned-owned>No</hubzoned-owned>
    <award-id>81417</award-id>
</Award>
```

Listing 5.2: Duplicate phase I award with a sbir.gov ID of 208656 and an award-id of 81417.

```
<Award>
    <id>206202</id>
    <title>Simulated Job Performance Assessment</title>
    <link>http://www.sbir.gov/sbirsearch/detail/206202</link>
    <abstract>JPS is designing and developing a standard and affordable
        set of procedures for creating and validating PC-based job
        simulations that can be applied across a broad range of Army MOS
         to assess Soldier job performance. As part of the Phase I
        effort, JPS is building a simulation for one Army MOS. In this
        effort JPS is developing a 10 step process for creation of the
        simulations.</abstract>
    <agency>ARMY</agency>
    <program>SBIR</program>
    <phase>1</phase>
    <year>2007</year>
    <company>JOB PERFORMANCE SYSTEMS, INC.</company>
    <ri></ri>
    <agency-tracking>A072-048-0409</agency-tracking>
    <topic-code>N/A</topic-code>
    <woman-owned>No</woman-owned>
    <minority-owned>No</minority-owned>
    <hubzoned-owned>No</hubzoned-owned>
    <award-id>81417</award-id>
</Award>
```

Listing 5.3: Duplicate phase I award with a sbir.gov ID of 206202 and an award-id of 81417.

5.2.4 Award Meta Data

Individual awards consist of two independent id's. The first ID is an index into the main `sbir.gov` database and is unique across all granting agencies. The second ID (award-id) is an index back into the granting agency's program and is only guaranteed to be unique within the specific agency making the grant. This discrepancy is caused by the decentralized administration across the granting agencies. The agency award id is critical when correlating phase I and phase II awards. Fortunately, the corpus consists of only 36 awards with a blank award-id. Thus, these awards are filtered out due to the fact that they cannot be properly correlated.

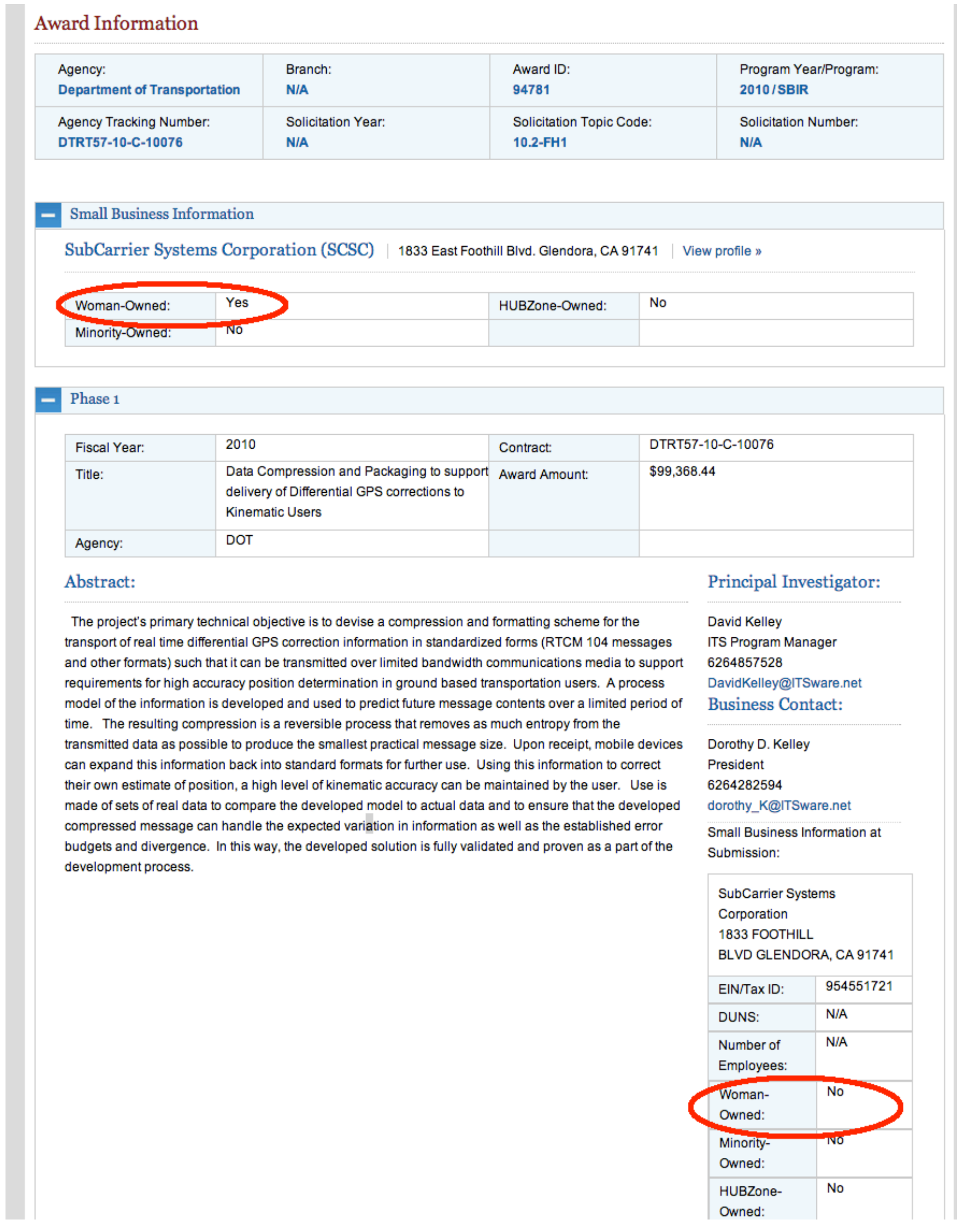
In addition to the awards with blank id's there were 707 phase II awards listed in the database that did not have a corresponding phase I award. According to the laws that govern the SBIR/STTR program, it is not possible to go directly to a phase II award. We attempted to find the missing phase II awards in the corpus by matching other features of the award such as the award title, abstract, agency, program, year, company, research institution, agency tracking number, topic code, woman owned, minority owned, HUBZoned owned, and award id. However, after searching the entire corpus it appears that the phase I awards simply do not exist in the corpus downloaded for `sbir.gov`. We removed the orphaned phase II awards from the tuned corpus.

Finally, the award metadata that indicates if an award is woman, minority, or HUBZoned owned is rife with discrepancies. Figure 5.2 shows an example of a problem award. We can see that for the award shown, the small business is listed as woman owned in one section and not woman owned in another section. Looking across the entire corpus it appears that the second listing is incorrect as all three properties are "No" for every single award. Thus, when extracting the data from `sbir.gov`, we will only look at data from the first section.

5.2.5 Time Gap between Phase I and Phase II Award

A phase I award is generally 6 months or less in time and a phase II is no more than two years. Thus, when looking for awards that have moved from a phase I to a phase II it is logical to assume a 1-year cutoff window to know if a phase I award has in fact been moved to phase II. This is important to note in order to avoid miss-labeling phase I awards as negative examples. Looking at past award data shown in Figure 5.3, we see a maximum time gap between phase I and phase II of 10 years with a minimum time gap of less than a year. The bulk of the awards fall within the predicted 1 year window. Therefore, as the data shows, we set a cutoff window of 3 years to maximize our accuracy when looking for phase I awards that have moved to phase II. Given that our maximum year in the corpus is 2009, the cutoff for all phase I awards that do not have a corresponding phase II award is set to 2007.

As we can see in Figure 5.4, we first find all phase I awards that are within the cutoff window. Once we have the phase I awards, we can look for a corresponding



Award Information

Agency: Department of Transportation	Branch: N/A	Award ID: 94781	Program Year/Program: 2010 / SBIR
Agency Tracking Number: DTRT57-10-C-10076	Solicitation Year: N/A	Solicitation Topic Code: 10.2-FH1	Solicitation Number: N/A

Small Business Information

SubCarrier Systems Corporation (SCSC) | 1833 East Foothill Blvd. Glendora, CA 91741 | View profile »

Woman-Owned:	Yes	HUBZone-Owned:	No
Minority-Owned:	No		

Phase 1

Fiscal Year:	2010	Contract:	DTRT57-10-C-10076
Title:	Data Compression and Packaging to support delivery of Differential GPS corrections to Kinematic Users	Award Amount:	$99,368.44
Agency:	DOT		

Abstract:

The project's primary technical objective is to devise a compression and formatting scheme for the transport of real time differential GPS correction information in standardized forms (RTCM 104 messages and other formats) such that it can be transmitted over limited bandwidth communications media to support requirements for high accuracy position determination in ground based transportation users. A process model of the information is developed and used to predict future message contents over a limited period of time. The resulting compression is a reversible process that removes as much entropy from the transmitted data as possible to produce the smallest practical message size. Upon receipt, mobile devices can expand this information back into standard formats for further use. Using this information to correct their own estimate of position, a high level of kinematic accuracy can be maintained by the user. Use is made of sets of real data to compare the developed model to actual data and to ensure that the developed compressed message can handle the expected variation in information as well as the established error budgets and divergence. In this way, the developed solution is fully validated and proven as a part of the development process.

Principal Investigator:

David Kelley
ITS Program Manager
6264857528
DavidKelley@ITSware.net

Business Contact:

Dorothy D. Kelley
President
6264282594
dorothy_K@ITSware.net

Small Business Information at Submission:

SubCarrier Systems Corporation
1833 FOOTHILL BLVD GLENDORA, CA 91741

EIN/Tax ID:	954551721
DUNS:	N/A
Number of Employees:	N/A
Woman-Owned:	No
Minority-Owned:	No
HUBZone-Owned:	No

Figure 5.2: Award metadata as extracted from the sbir.gov website showing inconsistent data. You can see that the company is reported as woman owned in one section and not woman owned in a separate section.

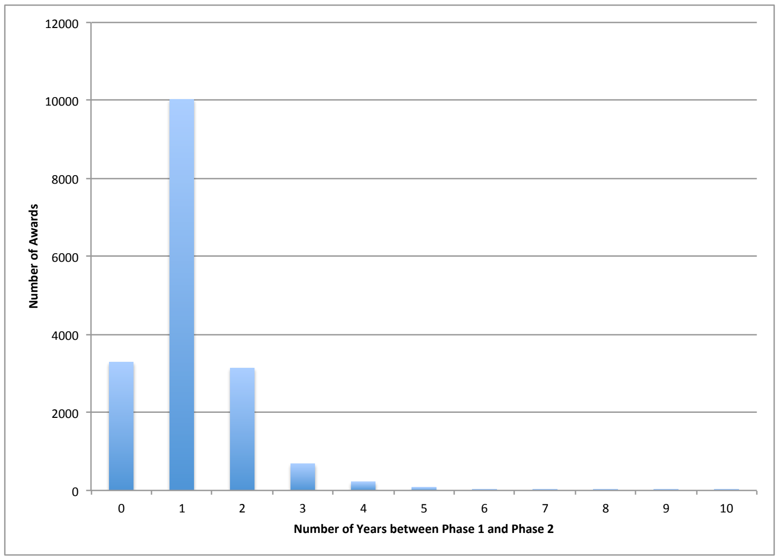


Figure 5.3: Time gap between a phase I and phase II award. As shown above, the bulk of the phase I to phase II awards fall within a 1 year window.

phase II award. We will look for phase II awards outside of the cutoff window so we can accurately label phase I awards that are within the cutoff window. Observe that Figure 5.4 shows two positive phase I awards and one negative phase I award. We can see in order to properly label the award represented by the yellow arrow we went outside the cutoff window to find the corresponding phase II award. When labeling the award represented by the blue arrow, we searched the entire corpus and found no corresponding phase II award, thus we labeled the award as a negative example.

5.3 The Tuned and Labeled Corpus

We construct a tuned and labeled corpus that is suitable for training classifiers as follows. First, we took the raw corpus and constructed a table of phase I and phase II award pairs. Then, we filter out the award pairs for missing abstracts, duplicate awards, missing metadata, and that are outside the cutoff window as described in the

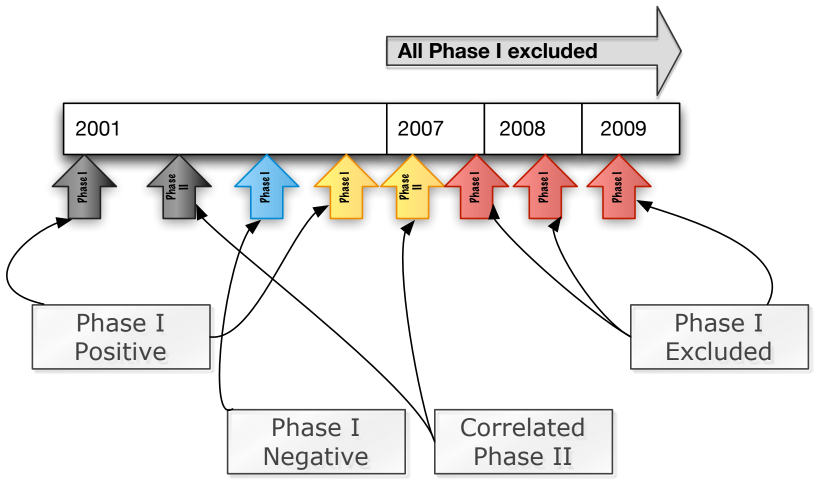


Figure 5.4: Labeling phase I awards as positive or negative. Each arrow represents an award. The black and yellow arrows represent a phase I that was labeled positive. The blue arrow represents a phase I award that was labeled negative. The red arrows represent phase I awards that are excluded from the corpus because they are beyond the cutoff window.

sections above. It is important to note that award pairs are removed from the corpus based on the phase I information. We are interested in finding properties of phase I awards that may indicate success. Thus, if we have a phase I and phase II award pair and the phase I award has a missing abstract, the award pair is removed regardless of what information is in the phase II.

Figure 5.5 shows that for the years 2000 and earlier, there are large numbers of awards with missing abstracts that we cannot use. When we filter out phase I awards due to missing abstracts, we are left with a very large number of negative examples from the years 1983-2000 giving a very unbalanced dataset. Thus, a cutoff is set at 2001, which is after the peak of missing abstracts. This gives us a final year range of 2001-2007, a 7 year range.

The tuned corpus as shown in Figure 5.6 will be the corpus that all experiments

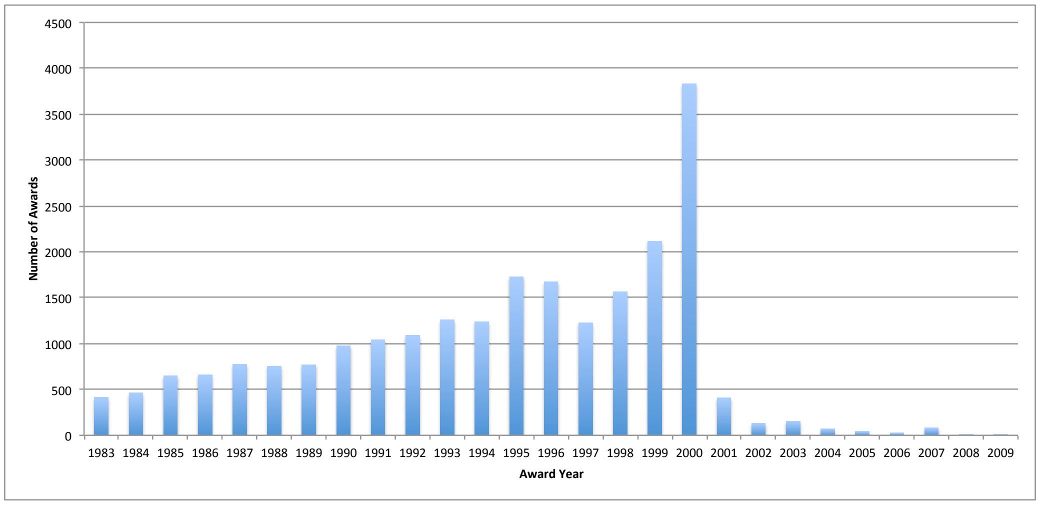


Figure 5.5: Phase I awards that are missing abstracts in the SBIR-STTR corpus

are executed against. The tuned corpus consists of 32,885 awards with 14,293 awards that were moved from a phase I to a phase II (positive examples) and 18,592 awards with only a phase I (negative examples). With any set of labeled data, it is possible to calculate the default accuracy of each class. The default accuracy is found by taking the number of documents in the class of interest divided by the total number of documents. Thus, our default accuracy rate for awards that were only granted a phase I is 57% $(18,592/32,885)$ and 43% $(14,293/32,885)$ for awards that were granted a phase II.

5.4 Tuned Corpus Results

Using the tuned corpus, we look at the possible features that may indicate if a phase I award will move to a phase II. Four different aspects of an award were examined: the amount of money the award requests, the properties of the proposal as defined in Section 5.4.2, the name of the primary investigator, and the text of the submitted

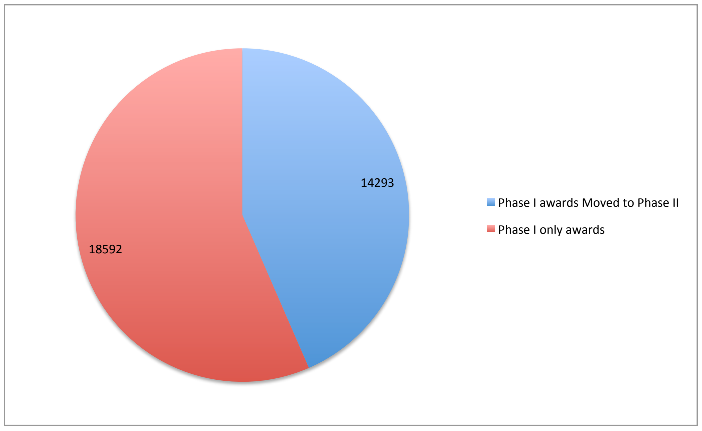


Figure 5.6: SBIR-STTR tuned and labeled corpus showing the total number of positive and negative awards.

abstract. Each section below details the setup and the results. Section 5.5 gives a summary of all the results.

5.4.1 Amount of Money Requested

When submitting an SBIR-STTR proposal, the company submitting has to compile a budget worksheet with estimated costs of the project. Phase I awards normally do not exceed 150,000 dollars in total costs for a period of 6 months. While the budget worksheet is not publicly available, the amount of money that the company requested is public record. Using the tuned corpus described in Section 5.3, we examined all phase I awards that were successfully moved to phase II and take note of the award amount requested. We then compared this to the phase I awards that were not moved to phase II to see if there is a difference in the amount requested.

Results

Figure 5.7 shows the amounts awarded, grouped in 20,000 dollar increments. The majority of phase I and phase II awards fall in the range of 60,000 - 80,000 dollars. The tuned dataset contains 2,649 awards that reported an award amount of zero. This is clearly an error in the dataset and is excluded from the results. It is interesting to note that there were 434 awards from the phase I to phase II collection and 1,258 awards from the phase I only collection that reported an amount in excess of 160,000 dollars. However, this discrepancy is due to how certain agencies, such as the NIH, report the funding amount for each organization [34]. Table 5.2 shows the top three amounts as percentages. We see that 32% of awards that are within the 60,000 - 80,000 dollar range and 24% within the 40,000 - 60,000 dollar range and 17% are in the 80,000 - 100,000 range.

Without access to the full budget of the award, it is not known what percentage of the award is for labor, materials, and overhead. Thus, it is feasible to assume that awards that are asking for more money may have higher material and overhead costs than the less expensive awards. For example, in the field of computer science, a proposal that needs to lease access to a supercomputer to complete the feasibility study would be more expensive than a proposal that can be conducted on a standard desktop machine. Given that the positive and negative awards are clustered around the same amounts, we can conclude that the award amount doesn't appear to have any impact when moving from a phase I to a phase II.

Award Amount	Percent awards moved to a phase II
80,000 - 100,000	17%
40,000 - 60,000	24%
60,000 - 80,000	32%

Table 5.2: Top three amounts for phase I awards moved to phase II.

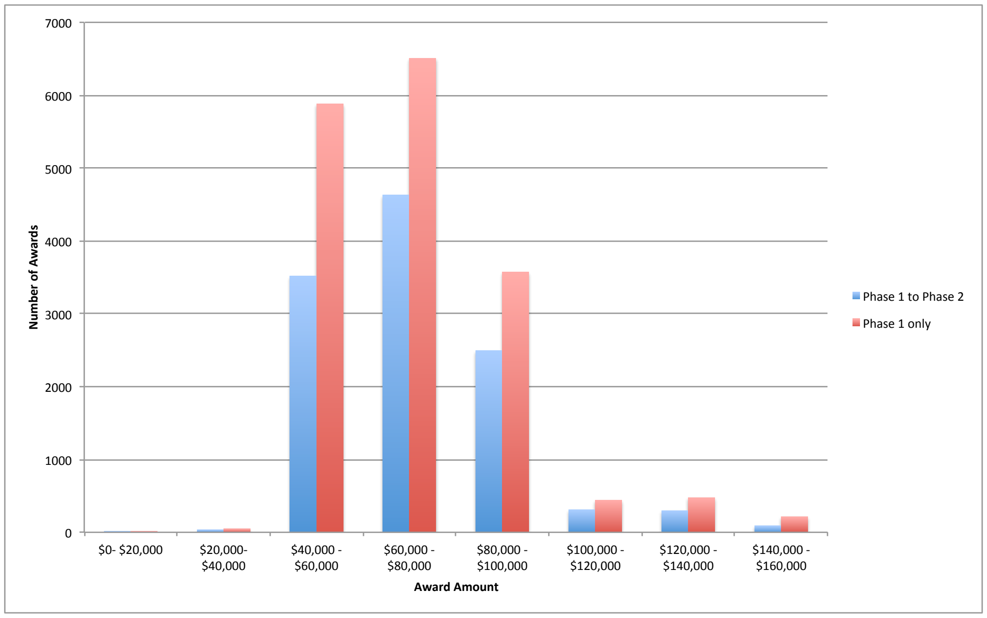


Figure 5.7: Award amounts for both phase I and phase II.

5.4.2 Properties of the Proposal

Looking at the tuned dataset, we can now look at the properties of the proposal as a factor in determining the likelihood of moving from a phase I to a phase II. Individuals that compete for SBIR/STTR awards are required to detail traits of the submitting company's ownership and location. Traits include whether the company is woman or minority owned and if it is located, or has branches, in a HUBZone. While statistics are available on `sbir.gov` regarding the total number of such awards, there has not been, to our knowledge, an analysis of these properties done on awards that have moved from phase I to phase II.

Results

Figures 5.8, 5.9, 5.10, and 5.11 each show the number of phase I awards that were moved to a phase II and awards that received only a phase I. For awards that are not listed as woman, minority, or HUBZoned owned there are, on average, 43% of the submitted phase I awards moved to a phase II, which is the same as the default accuracy of the class. Looking at minority and woman owned businesses the percentage of awards moved to a phase II drops to 37% and 42%, respectively, which is slightly below the default accuracy. HUBZoned owned business are the only group of awards that has a higher percentage of phase I awards moved to a phase II at 48%. Higher conversion rates should be expected for HUBZone companies because they get a 10% price evaluation preference and the federal government has a goal of awarding 3% of all dollars to HUBZoned certified businesses.

Based on the data from `sbir.gov`, it appears that locating a company in a HUBZone can significantly increase the possibility that you are moved to a phase II after receiving a phase I award. Table 5.3 summarizes the data and gives the base rate of a phase I award moving to a phase II. It is important to note that these properties are attached to the submitting company, not the primary investigator. For example, there may be a minority owned company that is submitting a proposal with a non-minority primary investigator.

The SBA is currently, as of October 1, 2012, aware of the lower number of awards granted to women and minority owned business. According to the SBIR Policy Directive [14], each SBIR Federal agency must use 3% of its SBIR budget to attempt to increase participation by Small Disadvantaged Business (SDB)s and

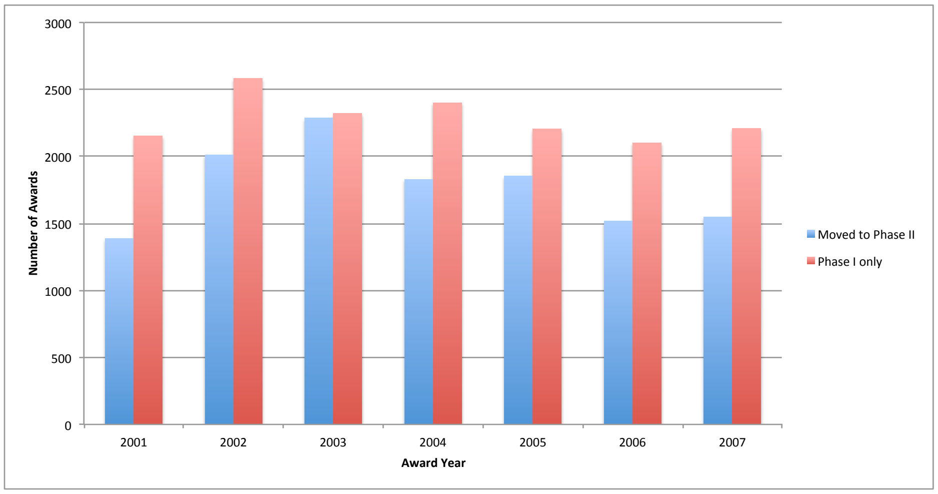


Figure 5.8: Awards moved to phase II that are not woman, minority, or HUBZoned owned

Women Owned Small Businesses (WOSB) in the SBIR Program. Thus, our findings are correlated with what the SBA is already aware of and trying to rectify.

Award Property	Percent moved to phase II	Confidence Interval
Minority owned	37%	2.25
Woman owned	42%	1.85
Default Rate	43%	n/a
NONE	43%	.58
HUBZoned owned	48%	4.15

Table 5.3: Percentage of awards that were move to a phase II based on the awards properties. Confidence Intervals are at a 95% desired confidence level.

5.4.3 Name and Title of the Primary Investigator

We now look at the surname origin and title of the primary investigator listed on the award to see if there are any features that we can extract that indicate if an award will move from a phase I to a phase II. We derive the name origin using the primary investigators surname and information listed on `ancestry.com`. While

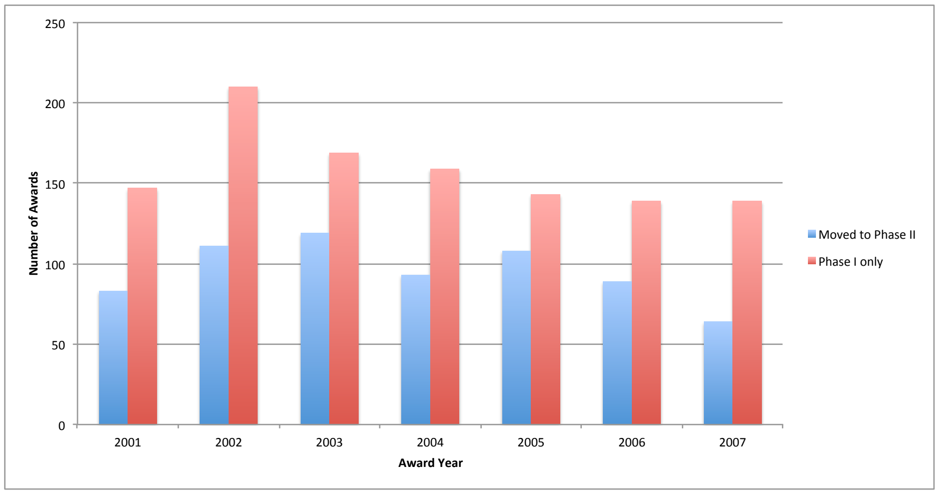


Figure 5.9: Awards moved to phase II with a minority owned company

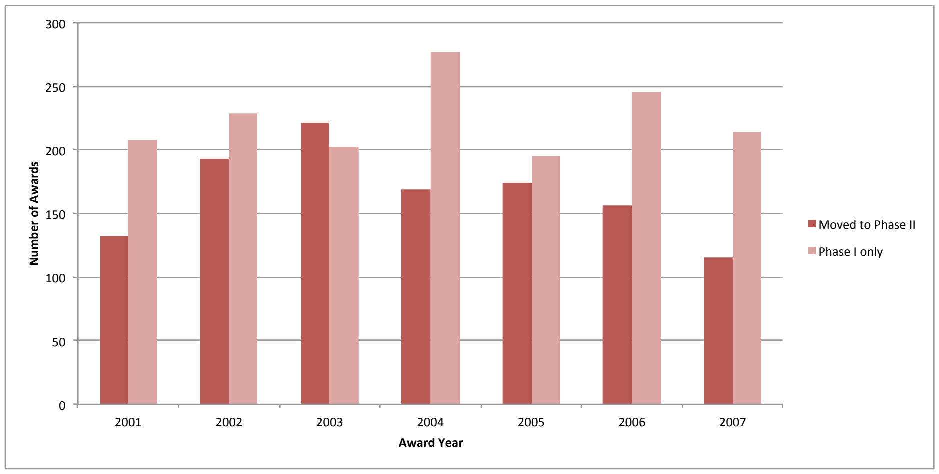


Figure 5.10: Awards moved to phase II with a woman owned company

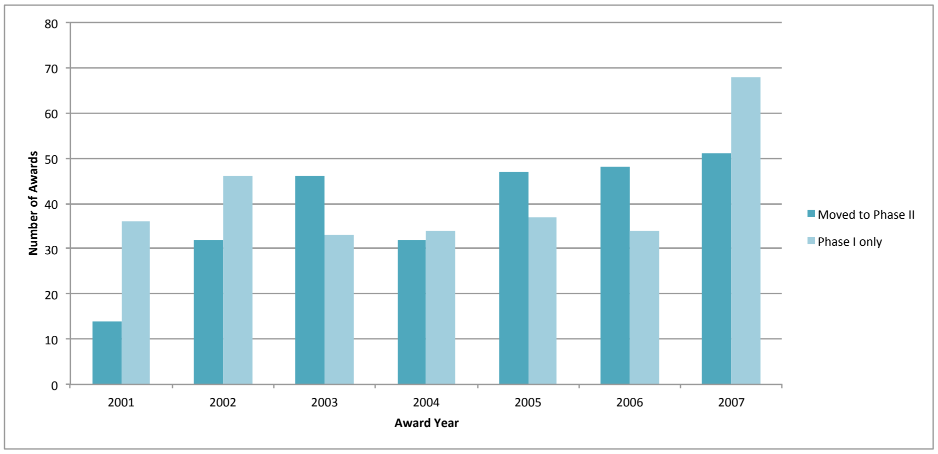


Figure 5.11: Awards moved to phase II with a HUBZone owned company

`ancestry.com` has a very large database of name origins, there are an staggering number of possible name spellings, names that have been hyphenated, and names with multiple origins. Any name that cannot be accurately resolved is labeled as UNKNOWN and is excluded from the analysis. As well as the surname origin, we group awards by the title of the primary investigator. The title is any variation (lower case, upper case, and punctuation) of PHD, DR, MD, and PE when choosing the awards to include.

Results

The database at `ancestry.com` contains very fine-grained name origins. For example, there are names listed as "South German" and "Northern German." While this level of detail is often needed when tracing back relatives (which is what `ancestry.com` is tuned for), it is far too detailed for our needs. Thus, we will take the name origins by country and place them in five broad regions: Asia , Americas, Caribbean, Europe, and the Middle East. Unfortunately, there are names within the database that are listed as belonging to multiple regions. The name "Ramila Philip" is listed as having an origin of "Scottish, Dutch, English, South Indian, etc." which would place the name in both the European (Scottish, Dutch, English) and Asia (South Indian) categories. To solve the problem of multiple name origins, we place the name in the category with the most records listed on `ancestry.com`. For the name "Philip," we see that as of January 2013, Scotland is listed as the top origin for the name with 64 entries and only 16 entries from the South Indian origin. Thus, "Philip" is placed in the European category for the SBIR-STTR corpus.

Figure 5.12 and 5.13 show the results of awards that were moved from a phase I to phase II and awards that only received a phase I. We broke out the results from Europe into a separate chart because Europe was the dominate category, thus making direct visual comparisons difficult. Table 5.4 shows the success rate of awards moving from a phase I to phase II for each region. When compared with the base rate of the corpus, names with origins from the Americas and Europe region have a better conversion rate than names originating from the Caribbean and Middle Eastern region.

Name origin	Percent moved to phase II	Confidence Interval
Middle East	38%	9.81
Caribbean	39%	9.07
Asia	43%	3.97
Default Rate	43%	n/a
Europe	44%	.63
Americas	45%	10.7

Table 5.4: Percentage of awards that were move to a phase II based on the primary investigators name origin.Confidence Intervals are at a 95% desired confidence level.

In the tuned corpus, we found 268 names with a title such as DR, PHD, MD, and PE. The award database at `sbir.gov` does not have a specific field for the primary investigators title, thus it is not known if the 268 names extracted include all possible

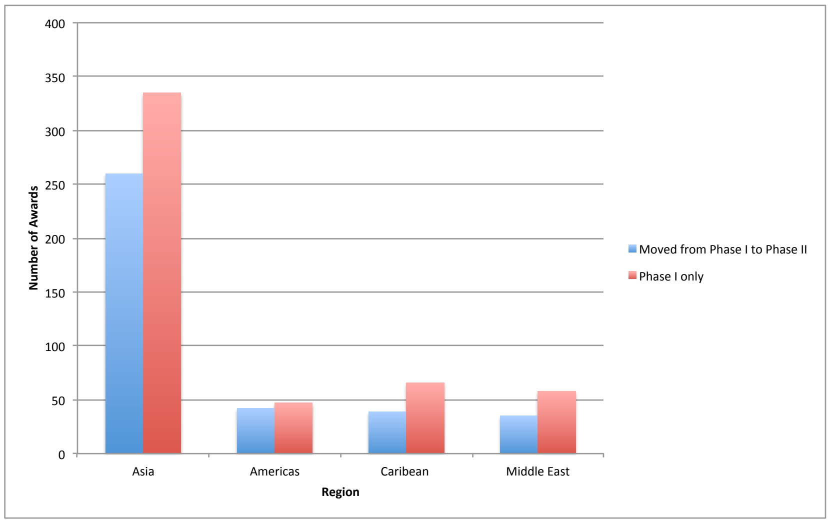


Figure 5.12: Awards moved to Phase II grouped by region

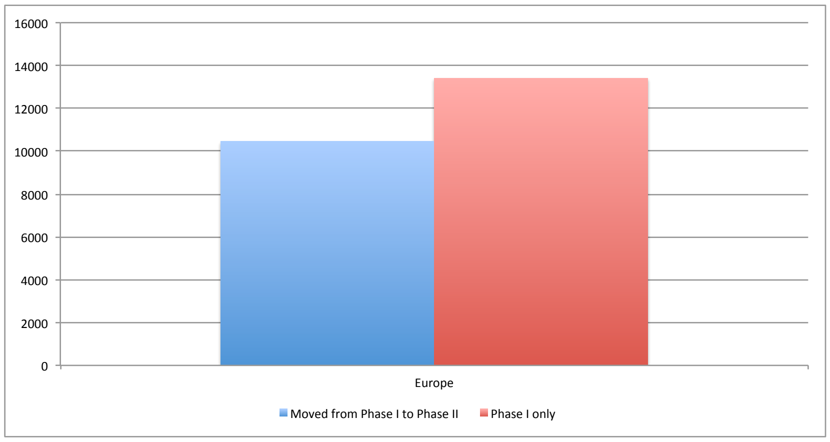


Figure 5.13: Awards moved to Phase II grouped from Europe

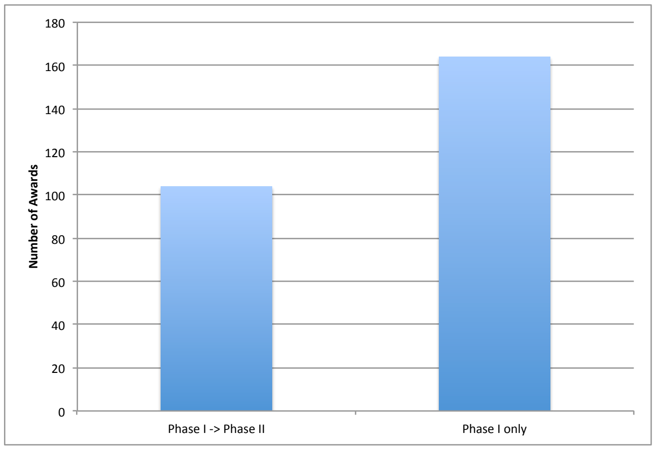


Figure 5.14: Awards moved to Phase II grouped by the title of the primary investigator

results. If the full award were available to the public, it would be possible to extract the title of the primary investigator from the biography section. The information that we do have allows us to at least make an educated guess on the impact a title may have on the success of moving from a phase I to a phase II. Figure 5.14 shows that awards that have a title listed with the primary investigators name have a conversion rate lower than the default rate. The default rate for the class is 43% and when looking at awards with titles listed, we find that only 38% of awards are moved from a phase I to a phase II.

5.4.4 Predicting a Phase II Award with Document Classification

Finally, we look at trying to predict a phase II award using document classification techniques. First, we load the tuned corpus into TAT and train a Naive Bayes, MaxEnt, and Decision Tree classifier from MALLET. Once we have trained classifiers, we evaluate their effectiveness as outlined in Chapter 2. We then determine if there are any textual properties that the classifiers can derive from the abstracts that can give an indication of success.

Results

Even with all the work in removing duplicate awards, awards with blank abstracts, and awards with missing or incorrect metadata, we still have problems that can have an impact when training classifiers. One problem that stands out is the slightly unbalanced nature of the training examples. In an ideal corpus, you want to have a balanced number of positive and negative examples from which to train. In our corpus, we have 14,293 positive examples and 18,592 negative examples. There are approximately 4,299 more negative examples than positive examples in the tuned data set. Thus, it is possible to cheat by biasing your guess toward the largest category. While this is not uncommon in real-world datasets, it presents a problem when trying to train a classifier.

Classifier	Precision	Recall	F-Measure
Naive Bayes	47%	68%	56%
MaxEnt	46%	46%	46%
Decision Tree	35%	27%	35%
Default Rate of 43%			

Table 5.5: Classifier Precision, Recall, and F-Measure for phase I to phase II awards using the SBIR-STTR tuned corpus.

Table 5.5 shows the results of running three different classifiers against the tuned SBIR-STTR corpus. We can immediately see that the Naive Bayes Classifier performs the best with a F-Measure of 56%. The worst performer for this corpus is the Decision Tree classifier with an F-Measure of 35%. Thus, with a Naive Bayes classifier, we can see that we have a slightly better chance than the default rate at predicting if an award will move to phase II. It is important to note that the complete proposal could include proprietary methods, trade secrets, or intellectual property that a company does not want released to the public. When submitting a proposal, the submitting company has the option of removing this critical information from the abstract. The removed information in some cases could be the deciding factor when selecting winners [10].

5.5 Summary

Over the course of this chapter, we have evaluated the SBIR-STTR corpus and evaluated features of awards that have been moved from a phase I to a phase II. We found that companies that are located in a HUBZone or have primary investigators with surnames with an origin from Europe or the Americas have slightly higher conversion rates from a successful phase I. We also found that a Naive Bayes classifier trained against the corpus can give a better indication of success than the base rate.

CHAPTER 6

CONCLUSIONS

6.1 Document Classification

Over the course of this thesis, we have reviewed the overall process of document classification. We first looked at the classification process in Chapter 2, which consists of how to represent the document, weigh the documents features, and algorithms used for training and classifying. We discussed problems that could impact the precision, recall, and overall accuracy such as feature selection, word stemming, and stop word removal.

An important topic in document classification is the selection and building of a corpus, which was reviewed in Chapter 3. Without a collection of documents, researchers would be forced to generate artificial corpora that may not accurately represent real-world data. We reviewed several popular datasets that are commonly used in research as well as introducing a new dataset for the document classification community to use.

The core topic of this thesis was an overview of TAT given in Chapter 4. The application modes and characteristics were enumerated along with the three plug-in points (Classifier, Corpus, and Change Tracking). Releasing TAT for use by researchers working on corpora construction and document classification saves years of development time and allows researchers to be productive quickly.

We saw in Chapter 5 that real-world data is fraught with problems and inconsistencies. It is important to take these problems into account when working on developing new classification algorithms, methods, and corpora. Ensuring that the working corpus that is used to train classifiers is of the highest quality possible is critical for success. If you are training on poor quality data, you will also classify poorly. Finally, we gave the analysis of the SBIR-STTR dataset and released both the raw data and tuned data that can be used for further analysis.

6.2 Future Directions and Work

As with all projects, there is never enough time to do everything that is needed. We have compiled a small list of enhancements to TAT and further research that can use the software and data generated. This includes, but is not limited to, extensions to TAT, further analysis of the SBIR-STTR dataset, and use of TAT to annotate or maintain existing datasets.

6.2.1 Extend TAT with Additional Classification Engines

TAT provides an excellent platform for language analysis and corpora construction and maintenance. To further improve the quality, bindings for other machine learning engines can be developed. There are a number of packages that provide a command line only interface that can benefit from a user friendly front end. Additionally, there are toolkits such as WEKA [15] that provide a powerful GUI interface into a range of classifiers that can be plugged into the TAT classifier module. Embedding the existing WEKA interface into TAT would further enhance both platforms.

6.2.2 Further SBIR-STTR Analysis

As seen in Section 5, a more detailed analysis of the SBIR/STTR reporting methods can provide value to the grant-seeking community. We found a large number of missing abstracts in the central `sbir.gov` repository for awards prior to 2002. As with most government run projects, there are numerous other sources that we can leverage to improve the corpus. Each granting agency typically has their own propriety interface into the awards granted as well as those maintained by agencies such as the Small Business Administration. Thus, it would be interesting to download all the awards data directly from each agency to see if the quality of the corpus can be improved. This is an inherently hard and long task due to the disparate methods of reporting and changes to the SBIR/STTR legislation over the years.

6.2.3 Reuters Corpora Plug-in

Due to the popularity of the corpora released by the Reuters corporation, it would be valuable to provided a default plug-in for TAT to allow researchers to use the RCV-1, RCV-2, and older Reuters-21578 corpora. The plug-ins could be configured to produce some of the more common training and test data splits used throughout the literature, eliminating the need to repeat this tedious work before results can be generated. TAT is ideally suited for corpora such as the RCV-1 and RCV-2 due to the restrictions put in place by the Reuters corporation.

6.3 Final Wrap up

We hope that the artifacts delivered over the course of this thesis provide value to the research community and help further the field of Document Classification. TAT and the new SBIR-STTR dataset are intended to to meet the needs of the research community and facilitate the progression of the science.

BIBLIOGRAPHY

[1] Kjersti Aas and Line Eikvil. Text categorisation: A survey. *Technical report, Norwegian Computing Center*, 941:1–34, 1999.

[2] R.A. Calvo, J.M. Lee, and X. Li. Managing content with automatic document classification. *Journal of Digital Information*, 5(2):1–15, 2004.

[3] W.B. Cavnar and J.M. Trenkle. N-gram-based text categorization. *Ann Arbor MI*, 48113:4001, 1994.

[4] TEI Consortium. Text encoding initiative. http://www.tei-c.org/index.xml, October 2012.

[5] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[6] D.H. Cunningham, D.D. Maynard, D.K. Bontcheva, and M.V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *In Recent Advanced in Language Processing*, pages 168–175, 2002.

[7] Common Crawl Foundation. A corpus of web crawl data composed of 5 billion web pages. electronic, November 2012.

[8] K. Fuka and R. Hanka. Feature set reduction for document classification problems. In *Proceedings of IJCAI-01 Workshop: Text Learning: Beyond Supervision*, 2001.

[9] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In *International Joint Conference on Artificial Intelligence*, volume 19, page 1048, 2005.

[10] Gail and Inc. Jim Greenwood, Greenwood Consulting Group. Sbir proposal writing basics: Protecting intellectual property. http://g-jgreenwood.com/sbir_proposal_writing_basics29.htm, January 2002.

[11] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

[12] C. Goller, J. Loning, T. Will, and W. Wolff. Automatic document classification. *Informationskompetenz-Basiskompetenz in der Informationsgesellschaft*, page 145, 2000.

[13] United States Government. Sbir-sttr program. http://www.sbir.gov/, 10 2012.

[14] United States Government. Sbir-sttr program directive. http://www.sbir.gov/about/sbir-policy-directive, 1 2013.

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[16] P.J. Hayes and S.P. Weinstein. Construe/tis: a system for content-based indexing of a database of news stories. In *Second Annual Conference on Innovative Applications of Artificial Intelligence*, volume 97, 1990.

[17] M.A. Hearst, ST Dumais, E. Osman, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and Their Applications*, 13(4):18–28, 1998.

[18] Alpert Jesse and Hajaj Nissan. We knew the web was big. *http://googleblog.blogspot.com*, 2008. Retrieved September 2009.

[19] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98*, pages 137–142, 1998.

[20] James Joyce. Bayes' theorem. *The Stanford Encyclopedia of Philosophy*, September 2008.

[21] N. Kang, E.M. van Mulligen, and J.A. Kors. Training text chunkers on a silver standard corpus: can silver replace gold? *BMC bioinformatics*, 13(1):17, 2012.

[22] T. Kenter and D. Maynard. Using gate as an annotation tool. *University of Sheffield, Natural language processing group*, 2005.

[23] J.D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. Genia corpusa semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182, 2003.

[24] David D. Lewis. Reuters-21578 text categorization test collection. http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt.

[25] D.D. Lewis, Y. Yang, T.G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.

[26] YH Li and AK Jain. Classification of text documents. *The Computer Journal*, 41(8):537, 1998.

[27] C.D. Manning, P. Raghavan, and H. Schutze. *An introduction to information retrieval*. Cambridge University Press, 2009.

[28] D.D. Margineantu. Active cost-sensitive learning. In *International Joint Conference on Artificial Intelligence*, volume 19, page 1622. LAWRENCE ERLBAUM ASSOCIATES LTD, 2005.

[29] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[30] A. McCallumzy and K. Nigamy. A comparison of event models for naive bayes text classification, 1998.

[31] Merriam-Webster. *Merriam-Webster Online Dictionary*, September 2010.

[32] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2):103–134, 2000.

[33] J. Nothman, T. Murphy, and J.R. Curran. Analysing wikipedia and gold-standard corpora for ner training. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 612–620. Association for Computational Linguistics, 2009.

[34] U.S. Department of Health and Human Services. Nih disclaimer regarding reporting amounts. http://report.nih.gov/award/index.cfm, January 2013.

[35] Oracle. Open ide framework. http://netbeans.org/features/platform/, January 2010.

[36] J.Y.H. Pong, R.C.W. Kwok, R.Y.K. Lau, J.X. Hao, and P.C.C. Wong. A comparative study of two automatic document classification methods in a library setting. *Journal of Information Science*, 34(2):213, 2008.

[37] MF Porter. An algorithm for suffix stripping. *Program*, 14(1):130–137, 1980.

[38] A. Roberts, R. Gaizauskas, M. Hepple, N. Davis, G. Demetriou, Y. Guo, J.S. Kola, I. Roberts, A. Setzer, A. Tapuria, et al. The clef corpus: semantic annotation of clinical text. In *AMIA Annual Symposium Proceedings*, volume 2007, page 625. American Medical Informatics Association, 2007.

[39] M. Rogati and Y. Yang. High-performing feature selection for text classification. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 659–661. ACM, 2002.

[40] Stuart Russell and Peter Norvig. *Artificial Intelligence A modern Approach*. Prentice Hall, 2 edition, 2003.

[41] Evan Sandhaus. The new york times annotated corpus. electronic, October 2008.

[42] F. Sebastiani. A tutorial on automated text categorisation. In *Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence*, pages 7–35. Citeseer, 1999.

[43] B. Settles, M. Craven, and L. Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, pages 1069–1078. Citeseer, 2008.

[44] R. Snow, B. O'Connor, D. Jurafsky, and A.Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263. Association for Computational Linguistics, 2008.

[45] A. Taylor, M. Marcus, and B. Santorini. The penn treebank: an overview. *Treebanks*, pages 1–22, 2003.

[46] A. Vehviläinen, E. Hyvönen, and O. Alm. A semi-automatic semantic annotation and authoring tool for a library help desk service. In *Proceedings of the first Semantic Authoring and Annotation Workshop, ISWC-2006, Athens, GA, USA*, 2006.

[47] W3C. Annotea project. http://www.w3.org/2001/Annotea/, December 2012.

[48] H. Zhang. The optimality of naive bayes. *American Association for Artificial Intelligence*, 1(2):3, 2004.

APPENDIX A

QUICKSTART

A.1 Overview

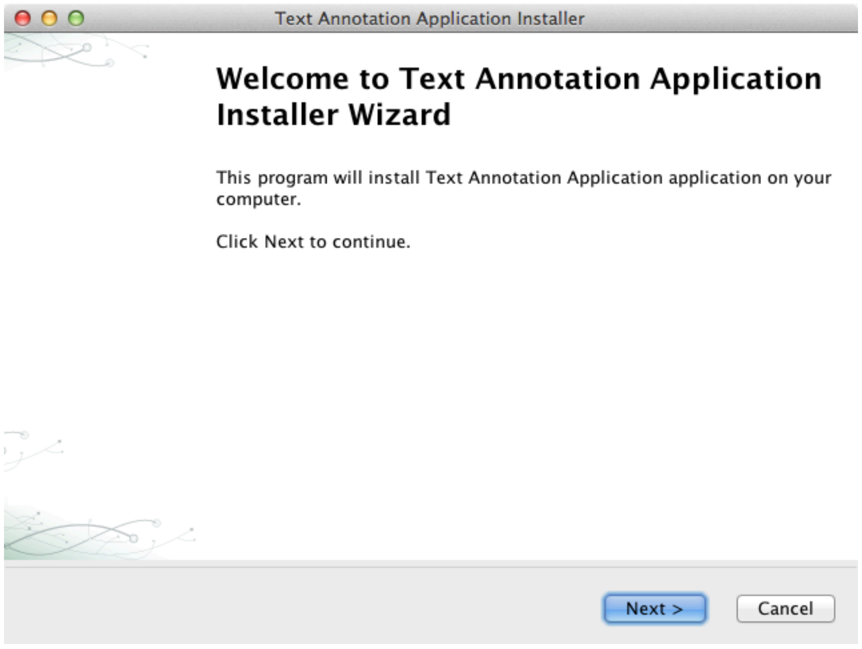
Getting up and running with TAT is a straight forward process. You can choose to download and build the application from source or install from pre-built binary installations. The following sections detail all the dependencies needed to get a complete running system.

A.2 Database Setup

The plug-ins provided out of the box depend on a working MySQL installation. Once you have MySQL installed add a new database named “tapp_core.” The default user name is “bsu” with a password of “bsu.” The user name and password can be updated once everything is installed and running. Give the user “bsu” access to the Schema “tapp_core” with all Object, DDL, and Other Rights. This will allow all necessary tables to be automatically managed by the application.

A.3 TAT

With a working MySQL installation, you can download the installer that is appropriate for your platform from `trac.boisestate.edu`. Figures A.1, A.2, A.3, and A.4 show the installer screens that you should see when running the installer. Figure A.5 shows the default window setup once the installation is complete with Figure A.6 showing the initial indexing of the SBIR-STTR tuned corpus. Finally Figure A.7 shows a document from the corpus being edited.

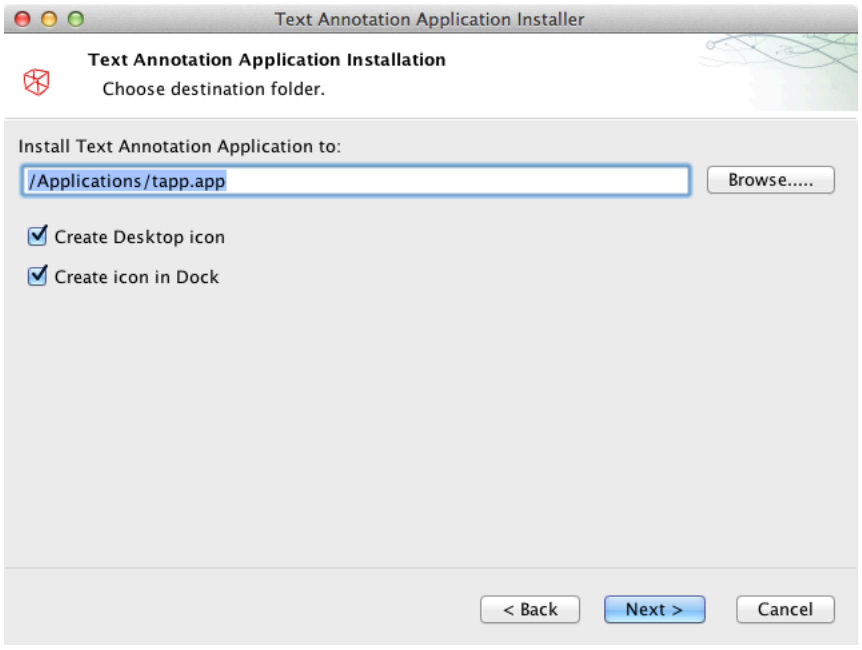


Welcome to Text Annotation Application Installer Wizard

This program will install Text Annotation Application application on your computer.

Click Next to continue.

Figure A.1: Installer first screen



Text Annotation Application Installation

Choose destination folder.

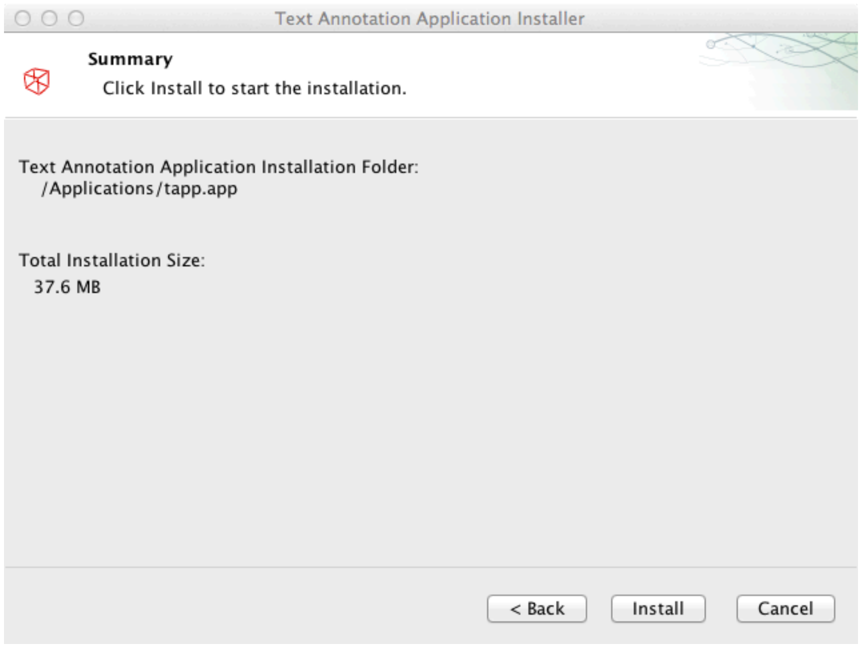
Install Text Annotation Application to:

/Applications/tapp.app

☑ Create Desktop icon

☑ Create icon in Dock

Figure A.2: Installer second screen

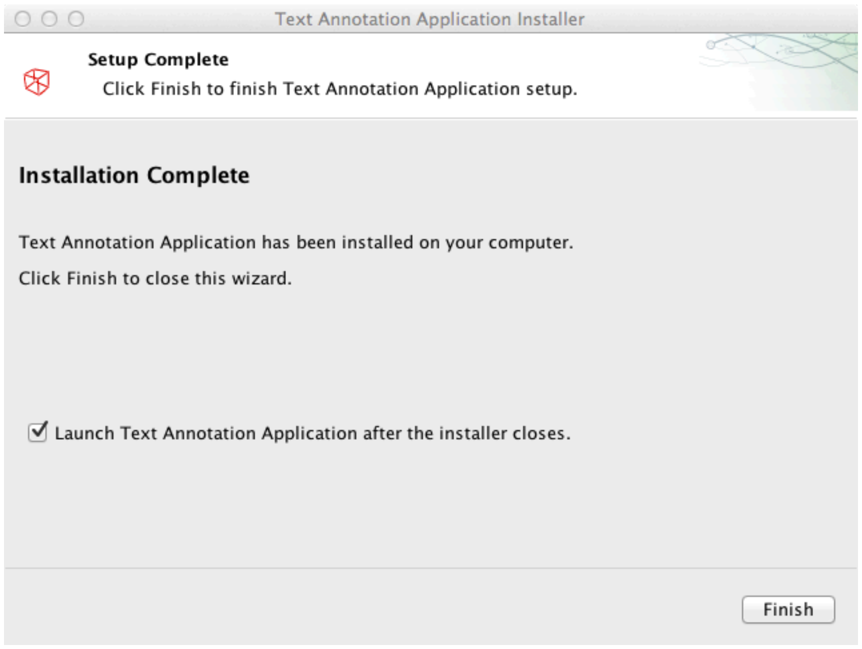


Summary
Click Install to start the installation.

Text Annotation Application Installation Folder:
/Applications/tapp.app

Total Installation Size:
37.6 MB

Figure A.3: Installer third screen



Setup Complete
Click Finish to finish Text Annotation Application setup.

Installation Complete

Text Annotation Application has been installed on your computer.

Click Finish to close this wizard.

☑ Launch Text Annotation Application after the installer closes.

Figure A.4: Installer fourth screen

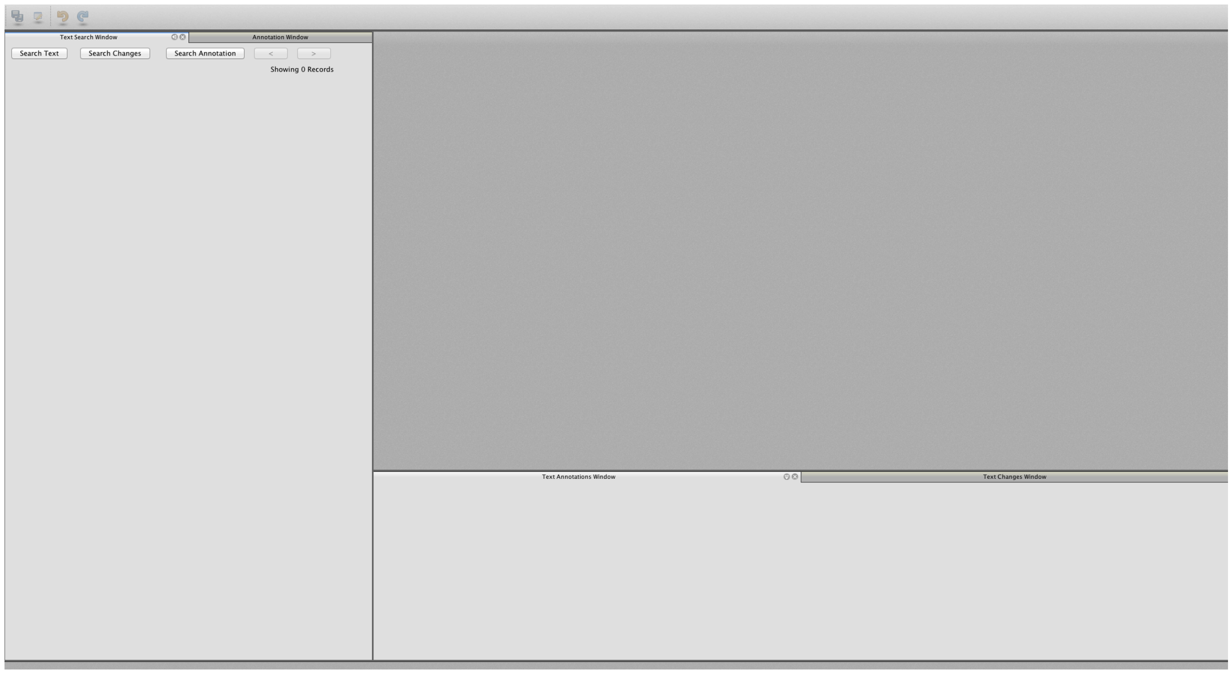
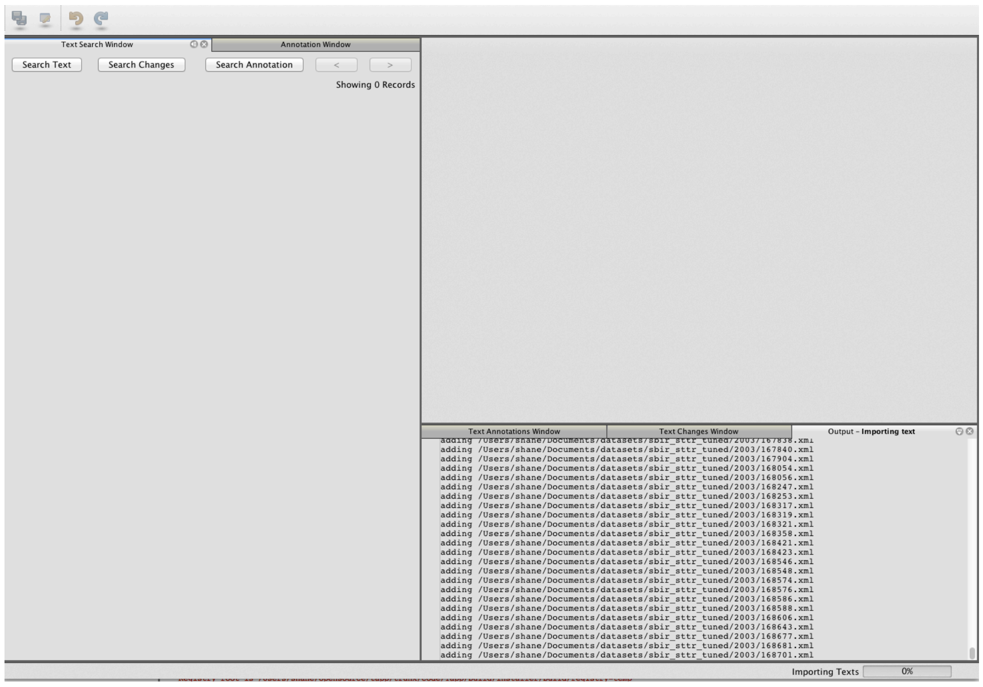
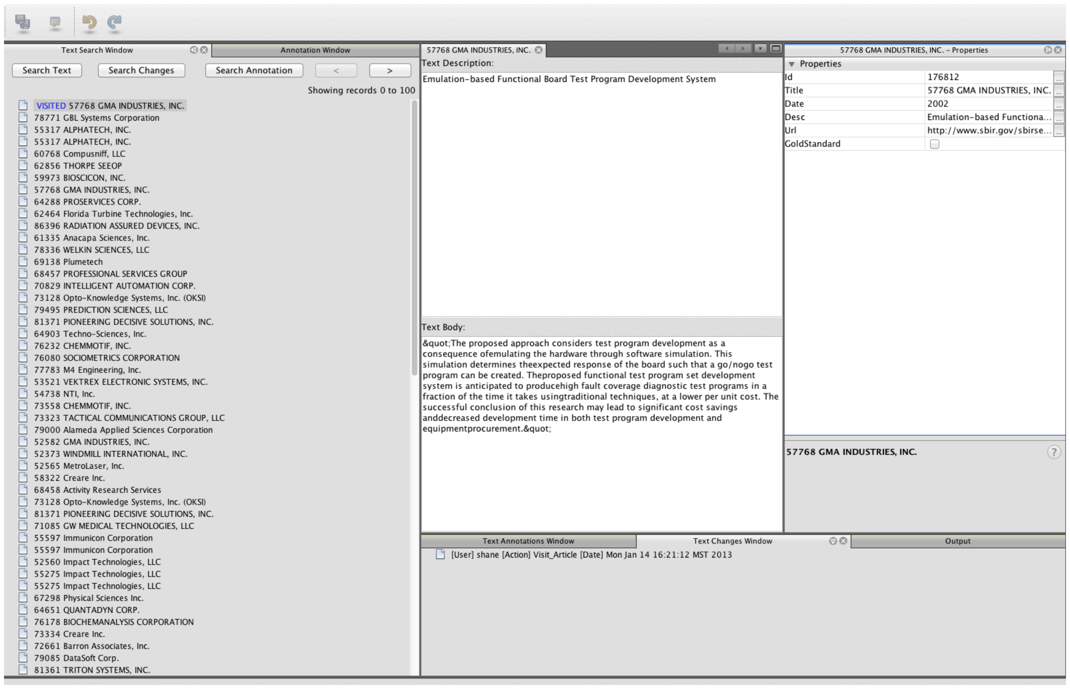


Figure A.5: Default windowing setup after clean install



```
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/167838.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/167840.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/167904.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168054.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168056.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168247.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168253.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168317.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168319.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168321.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168358.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168421.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168423.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168546.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168548.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168574.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168576.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168586.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168588.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168606.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168643.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168677.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168681.xml
adding /Users/shane/Documents/datasets/sbir_sttr_tuned/2003/168701.xml
```

Figure A.6: Indexing the SBIR-STTR corpus after install



Showing records 0 to 100

```
VISITED 57768 GMA INDUSTRIES, INC.
78771 GBL Systems Corporation
55317 ALPHATECH, INC.
55317 ALPHATECH, INC.
60768 Compusniff, LLC
62856 THORPE SEEOP
59973 BIOSCICON, INC.
57768 GMA INDUSTRIES, INC.
64288 PROSERVICES CORP.
62464 Florida Turbine Technologies, Inc.
86396 RADIATION ASSURED DEVICES, INC.
61335 Anacapa Sciences, Inc.
78336 WELKIN SCIENCES, LLC
69138 Plumetech
68457 PROFESSIONAL SERVICES GROUP
70829 INTELLIGENT AUTOMATION CORP.
73128 Opto-Knowledge Systems, Inc. (OKSI)
79495 PREDICTION SCIENCES, LLC
81371 PIONEERING DECISIVE SOLUTIONS, INC.
64903 Techno-Sciences, Inc.
76232 CHEMMOTIF, INC.
76080 SOCIOMETRICS CORPORATION
77783 M4 Engineering, Inc.
53521 VEKTREX ELECTRONIC SYSTEMS, INC.
54738 NTI, Inc.
73558 CHEMMOTIF, INC.
73323 TACTICAL COMMUNICATIONS GROUP, LLC
79000 Alameda Applied Sciences Corporation
52582 GMA INDUSTRIES, INC.
52373 WINDMILL INTERNATIONAL, INC.
52565 MetroLaser, Inc.
58322 Creare Inc.
68458 Activity Research Services
73128 Opto-Knowledge Systems, Inc. (OKSI)
81371 PIONEERING DECISIVE SOLUTIONS, INC.
71085 GW MEDICAL TECHNOLOGIES, LLC
55597 Immunicon Corporation
55597 Immunicon Corporation
52560 Impact Technologies, LLC
55275 Impact Technologies, LLC
55275 Impact Technologies, LLC
67298 Physical Sciences Inc.
64651 QUANTADYN CORP.
76178 BIOCHEMANALYSIS CORPORATION
73334 Creare Inc.
72661 Barron Associates, Inc.
79085 DataSoft Corp.
81361 TRITON SYSTEMS, INC.
```

Text Description:

Emulation-based Functional Board Test Program Development System

Text Body:

&quot;The proposed approach considers test program development as a consequence ofemulating the hardware through software simulation. This simulation determines theexpected response of the board such that a go/nogo test program can be created. Theproposed functional test program set development system is anticipated to producehigh fault coverage diagnostic test programs in a fraction of the time it takes usingtraditional techniques, at a lower per unit cost. The successful conclusion of this research may lead to significant cost savings anddecreased development time in both test program development and equipmentprocurement.&quot;

Id	176812
Title	57768 GMA INDUSTRIES, INC.
Date	2002
Desc	Emulation-based Functiona…
Url	http://www.sbir.gov/sbirse…
GoldStandard	☐

57768 GMA INDUSTRIES, INC.

```
[User] shane [Action] Visit_Article [Date] Mon Jan 14 16:21:12 MST 2013
```

Figure A.7: Editing a document after install

APPENDIX B

PLUG-IN INTERFACES

B.1 Corpus Plug-in

```
package edu.boisestate.services.corpus;

import java.net.URI;
import java.util.Collection;
import java.util.List;
import org.openide.windows.InputOutput;

/**
 * Provides an interface into the desired Corpus.
 * @author Shane Panter
 */
public interface ICorpus {

    /**
     * Given a query return a list of matching IText Objects
     * @param query A plug-in defined query
     * @param field The field to query
     * @param firstResult the first result to return
     * @param maxResult the maxResult
     * @param gold return only gold articles.
     * @return
     */
    public List<IText> find(String query,String field, int firstResult,
        int maxResult, Boolean gold);

    /**
     * Gets all the searchable fields that this corpus supports
     * @return the supported fields
     */
    public List<String> getFields();

    /**
     * Remove the selected text from the corpus
     * @param t the list of texts to remove
     */
    public void removeText(IText t);


    /**
     * Update a give text
     * @param t the text to update.
     */
    public void updateText(IText t);

    /**
     * Get all the texts owned by a specific user
     * @param loginName the name of the user
     * @param first the users first name
     * @param max the maximum number of articles to return
     * @return the texts found
     */
    public Collection<IText> getTextsForUser(String loginName, int first
        , int max);

    /**
     * Get all the texts annotated for a specific user
     * @param annotations the annotations of interest
     * @param start The date to start looking
     * @param finish The data to finish
     * @param firstResult the first result to return
     * @param max the maximum number of articles to return
     * @return the texts found
     */
    public List<IText> getTextsForAnnotations(ITextFacet facet, int
        firstResult, int max);


    /**
     * Build a Search index across the corpus
     * Note: this may not be necessary depending on the
     * internal needs of the plug-in
     */
    public void buildSearchIndex();

    /**
     * Get all facets of a specified type
     * @param type the type of facet to get
     * @return a list of facets
     */
    public List<ITextFacet> getAllFacetsOfType(String type);


    /**
     * All the top level annotations. If you annotations
     * are not hierarchical then this would just return
     * all available annotations.
     * @return
     */
    public List<ITextFacet> getAllTopLevelAnnotations();
}
```

Listing B.1: Corpus plug-in Interface

```
package edu.boisestate.services.corpus;

import java.io.Serializable;
import java.util.Set;

/**
 * Represents a body of text in the system.
 * @author Shane Panter
 */
public interface IText extends Serializable {

    /**
     * The date of the Text
     * @return
     */
    public String getDate();

    /**
     * Returns a description of the text
     * @return A short description
     */
    public String getDesc();

    /**
     * The gold standard status of a text
     * @return true if this text is a gold standard
     */
    public Boolean getGoldStandard();

    /**
     * The Id of a text. Typically a unique identifier
     * @return the id
     */
    public String getId();

    /**
     * The main body of the text
     * @return The body text
     */
    public String getText();

    /**
     * The title of the text. Can be the same as the description
     * @return the Title
     */
    public String getTitle();

    /**
     * The Url of the text if this is a web resource
     * @return the URL
     */
    public String getUrl();

    /**
     * Extension point for the plug-ins
     * @return
     */
    public  Object getWrappedData();

    /**
     * Return all annotations (labels or facets) for this text
     * @return the annotations
     */
    public Set<ITextFacet> getAnnotations();

    /**
     * Sets the data of the text
     * @param date the date to set
     */
    public void setDate(String date);

    /**
     * Sets the Description of the text
     * @param desc
     */
    public void setDesc(String desc);

    /**
     * Sets the text of the text
     * @param text the text to set
     */
    public void setText(String text);

    /**
     * Sets the title of the text
     * @param title the title
     */
    public void setTitle(String title);

    /**
     * Sets the URL of this text
     * @param url the URL
     */
    public void setUrl(String url);

    /**
     * Updates this texts gold standard status
     * @param goldStandard true if this text is a gold standard
     */
    public void setGoldStandard(Boolean goldStandard);

    /**
     * Sets the Id of this text.
     * @param Id the ID
     */
    public void setId(String Id);

    /**
     * Adds a new Annotation to this text
     * @param prop the annotation property
     * @param obj the annotation object
     */
    public void addAnnotation(ITextFacet f);


    /**
```

```
     * Removed an annotation from this text
     * @param f the annotation to remove
     */
    public void removeAnnotation(ITextFacet f);

}
```

Listing B.2: Text or Document plug-in Interface

```
package edu.boisestate.services.corpus;

/**
 * Provides an interface for a text facet(annotation)
 * @author Shane Panter
 */
public interface ITextFacet{

    /**
     * Number representing the id of the facet. This could be a direct
        index into
     * a database or some other identifier.
     * @return The id of the Facet
     */
    public String getId();

    /**
     * Returns the property of this facet
     * @return the property
     */
    public String getProperty();

    /**
     * Returns the Object of this facet
     * @return the facet object
     */
    public String getObject();


    /**
     * Sets the id of the TextFacet
     * @param id The id to set
     */
    public void setId(String id);

    /**
     * Sets the facet property
     * @param property the property to set
     */
    public void setProperty(String property);

    /**
     * Set the Object Property
     * @param object the Object to set
     */
    public void setObject(String object);

    /**
     * Custom extension point for plug-in
     * @return
     */
    public Object getWrappedData();
```

```

}
```

Listing B.3: Annotation plug-in Interface

B.2 Classifier Plug-in

```
package edu.boisestate.services.classification;

import edu.boisestate.services.corpus.IText;

/**
 * This interface provides all the necessary methods needed to allow any
 * classifier to be plugged into the framework
  * @author Shane Panter
 */
public interface IClassifier {


    /**
     * The name of the classifier. A good choice for the name includes
         the
     * algorithm + corpus that it was trained on
     * @return A human readable name
     */
    public String getName();

    /**
     * The ID of the classifier. This field is used for saving the
         classifier
     * and indexing this should be unique and is depending on how the
        classifier
     * is saved (i.e. saved to a database or disk)
     * @return A unique ID
     */
    public String getID();

    /**
     * The self reported accuracy of the classifier
     * @return The Accuracy
     */
    public String getAccuracy();

    /**
     * The self reported recall of the classifier
     * @return The Recall
     */
    public String getRecall();

    /**
     * The self report precision of the classifier
     * @return The Precision
     */
    public String getPrecision();

    /**
     * The self reported F-measure score
     * @return The F-measure
     */
    public String getFMeassure();

    /**
     * The data the classifier was last trained
     * @return A formatted Date
     */
    public String getDate();

    /**
     * Additional information specific to this classifier
     *
     * @return Any other important data
     */
    public String getOther();
}
```

Listing B.4: Classifier plug-in Interface

```
package edu.boisestate.services.classification;

import edu.boisestate.services.corpus.IText;
import java.util.List;

/**
 * Interface into the classification engine
 *
 * @author Shane Panter
 */
public interface IClassification {

    /**
     * Get all trained classifiers that the system knows about
     * @return trained classifiers
     */
    public List<IClassifier> getClassifiers();

    /**
     * Given a lists of texts and a Classifier classify each
     * text in the list with the given classifier
     * @param t the list of texts to classify
     * @param c the classifier to use
     * @return the classified texts
     */
    public List<IText> classifyTexts(List<IText> t, IClassifier c);

  /**
   * Trains a new classifier
   * @param t The list of texts to train on
   * @param type the type of classifier to train
   * @param customOptions options to pass to the classifier
   * @param train the training set split percentage
   * @param test the test set split percentage
   * @return
   */
    public IClassifier trainClassifier(List<IText> t,  String type,
        String customOptions, double train, double test);

    /**
     * Remove a classifier from the engine
     * @param c the classifier to remove.
     */
    public void removeClassifier(IClassifier c);

    /**
     * returns a list of supported classifiers for this plug-in
     * @return A list of supported classifiers
     */
    public List<String> supportedClassifiers();
}
```

Listing B.5: Classification plug-in Interface

B.3 Change Tracking Plug-in

```
package edu.boisestate.services.tappcore;

import java.io.Serializable;
import java.util.Date;

/**
 * Represents an event in the System. This represents the core change
     tracking
 * functionality.
 * @author Shane Panter
 */
public interface IEvent extends Serializable {

    /**
     * The event Id. Typically unique
     * @return the events Id
     */
    public Integer getId();

    /**
     * The user that this event is linked to
     * @return  The user object for this event
     */
    public IUser getUser();

    /**
     * The date of the event
     * @return
     */
    public Date getDate();

    /**
     * The Old Value of the document before this event occured
     * @return the old value
     */
    public String getOldValue();

    /**
     * The Property of this event if this was an annotation event
     * @return the event property
     */
    public String getEventProperty();

    /**
     * The Object of this event if this was an annotation event
     * @return the event object
     */
    public String getEventObject();

    /**
     * The text to which this event applies
     * @return the text id
     */
    public String getTextId();

    /**
     * The action the user took
     * @return the action
     */
    public String getUserAction();

    /**
     * Get the starting offset into the text
     * @return The offset
     */
    public Integer getTextStart();

    /**
     * Get the ending offset into the text
     * @return the ending offset
     */
    public Integer getTextEnd();

    /**
     * Sets the user of this event
     * @param u the user's name
     */
    public void setUser(String u);
    /**
     * Sets the date of this even
     * @param d the date
     */
    public void setDate(Date d);

    /**
     * Set the Old Value of this event
     * @param old the old value
     */
    public void setOldValue(String old);

    /**
     * Set the events property
     * @param prop the property
     */
    public void setEventProperty(String prop);

    /**
     * Sets the event object
     * @param obj the object
     */
    public void setEventObject(String obj);

    /**
     * Set the text Id of the event
     * @param id the id of the event
     */
    public void setTextID(String id);

    /**
     * Set the Action that the user took
     * @param action The users action
     */
    public void setUserAction(String action);

    /**
     * Set the the starting offset into the text
     * @param start the staring offset
     */
    public void setTextStart(Integer start);

    /**
     * Set the ending offset into the text
     * @param end the ending offset
     */
    public void setTextEnd(Integer end);
}
```

Listing B.6: TAT event plug-in Interface

```
package edu.boisestate.services.tappcore;

import java.io.Serializable;
import java.util.List;

/**
 * Represent a user
 * @author Shane Panter
 */
```

```
public interface IUser extends Serializable{

    /**
     * The Id of the user
     * @return The users id
     */
    public Integer getId();

    /**
     * The login name of the user
     * @return The users login name
     */
    public String getLoginName();

    /**
     * The Full name of the user
     * @return the users full name
     */
    public String getRealName();

    /**
     * The password of the user
     * @return The users encrypted password
     */
    public String getPassword();

    /**
     * Get a list of events associated with this user object
     * @return A list of events
     */
    public List<? extends IEvent> getEventList() ;

    /**
     * Sets the id of the user
     * @param id The new user id
     */
    public void setId(Integer id);

    /**
     * Sets the login name of the user
     * @param name The users new login name
     */
    public void setLoginName(String name);

    /**
     * Sets the users Real name
     * @param name The real name of the user
     */
    public void setRealName(String name);

    /**
     * Sets the encrypted password of the user
     * @param pass the users password
     */
    public void setPassword(String pass);

}
```

Listing B.7: TAT User plug-in Interface

```
package edu.boisestate.services.tappcore;
```

```

import java.util.Date;
import java.util.List;
import java.util.Set;

/**
 * Controller for the Change tracking system
 *
 * @author Shane Panter
 */
public interface ITappCore {

    /**
     * Creates a new user
     * @param login The login name of the user
     * @param userName the real name of the user
     * @param pass the users password
     * @return A new User object
     */
    public IUser createUser(String login, String userName, String pass);

    /**
     * Creates a new user using the password of the host system
     * @param login The login name of the user
     * @param userName the real name of the user
     * @return A new User object
     */
    public IUser createUser(String login, String userName);

    /**
```

```
     * Creates a new user using the password of the host system
     * and the real name from the host system
     * @param login the login name of the user
     * @return A new User object
     */
    public IUser createUser(String login);

    /**
     * Looks up a user with the specified login name
     * @param loginName the login name of the user
     * @return A found user or null if no user exists
     */
    public IUser lookupUser(String loginName);

    /**
     * Removes a user from the system
     * @param loginName the login name of the user to delete
     * @return True if the user can be deleted
     */
    public boolean deleteUser(String loginName);

    /**
     * Get all the Events that are logged for the given text id
     * @param textID the id of the text to look for
     * @return A list of found events
     */
    public List<? extends IEvent> getEventsForText(String textID);

    /**
     * Create a new concrete event
     * @param user the user name
     * @param date the date of the event
     * @param oldValue the value to save off
     * @param property the property of the event
     * @param object the object of the event
     * @param textId the text to associate this event to
     * @param UserAction the action the user took
     * @param textStart The start offset into the text
     * @param textEnd The end offset into the text
     * @return
     */
    public IEvent addEvent(String user, Date date, String oldValue,
        String property, String object, String textId, String UserAction
        , Integer textStart, Integer textEnd);

    /**
     * Remove an event from the system
     * @param entry The event to remove
     * @return true if the event can be removed
     */
    public boolean deleteEvent(IEvent entry);

    /**
     * Get all the users currently known in the system
     * @return A list of users
     */
    public List<? extends IUser> getUsers();

    /**
     * Get all the currently known events in the system
     * @return A list of known events
     */
    public List<? extends IEvent> getEvents();

    /**
     * Get all the events for a specific user
     * @param loginName The login name of the user to fetch
     * @param first The first result to return
     * @param max The max number of results to return
     * @return  A list of events found
     */
    public List<? extends IEvent> getEventsForUser(String loginName, int
         first, int max);

    /**
     * Get A list of all the text Ids by user
     * @param loginName The user to look for
     * @param first the first result to return
     * @param max the max number of results to return
     * @return A set of text ids
     */
    public Set<Long> getAllTextIdForUser(String loginName, int first,
        int max);

    /**
     * Get all the events for a specific user and text
     * @param loginName the login name to look for
     * @param textid the id of the text to look for
     * @return A list of events
     */
    public List<? extends IEvent> getEventsForUserAndText(String
        loginName, String textid);
}
```

Listing B.8: Core controller plug-in Interface