Keywords at Work: Investigating Keyword Extraction in Social Media Applications

by

Shibamouli Lahiri

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2018

Doctoral Committee:

Professor Rada F. Mihalcea, Chair
Assistant Professor Danai Koutra
Professor Paul Tarau
Assistant Professor V. G. Vinod Vydiswaran

Shibamouli Lahiri
lahiri@umich.edu
ORCID iD: 0000-0002-7278-2000

I dedicate this work to God Almighty, who is the creator, sustainer, and savior of all.

ACKNOWLEDGMENTS

I would like to thank my advisor Prof. Rada Mihalcea for her unceasing support over the last five years, for being a great listener and problem-solver, and for giving me the opportunity to view life and research in a realistic fashion.

I would like to thank my committee members: Prof. V. G. Vinod Vydiswaran, Prof. Danai Koutra, and Prof. Paul Tarau for their unconditional support and encouragement, and sound advice on research and life.

I would like to thank Dr. Carmen Banea for the sound help and expert advice on two key projects that form a part of this work.

I would like to thank Dr. Po-Hsiang Lai for the help and advice on the keyword extraction project that forms the first part of this work.

I would like to thank my friends and colleagues at the Language and Information Technologies (LIT) group, in the Computer Science and Engineering department, and in Ann Arbor – for being there with me whenever I needed them. I would also like to thank the many friends and colleagues whom I have met since coming to the US in 2008 – at Penn State University, and at University of North Texas. This work is a culmination of all their support and good wishes. Last but not the least, I would like to offer my gratitude to my extended family, esp. my parents – Dr. Saibendu Kumar Lahiri and Dr. Sukriti Lahiri – for being a continuous source of support. This work would not have been possible without them.

Chapter 3 is based in part upon work supported by Samsung Research America under agreement GN0005468 and by the National Science Foundation under IIS award #1018613. Chapter 4 is based in part upon work supported by the Michigan Institute for Data Science. Chapter 5 is based in part upon work supported by the Michigan Institute for Data Science, by the National Science Foundation (grant #1344257), and by the John Templeton Foundation (grant #48503). Any opinions, findings, conclusions or recommendations expressed in

the chapters are those of the author and do not necessarily reflect the views of the funding agencies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This dissertation examines a long-standing problem in Natural Language Processing (NLP) – keyword extraction – from a new angle. We investigate how keyword extraction can be formulated on social media data, such as emails, product reviews, student discussions, and student statements of purpose. We design novel graph-based features for supervised and unsupervised keyword extraction from emails, and use the resulting system with success to uncover patterns in a new dataset – student statements of purpose. Furthermore, the system is used with new features on the problem of *usage expression* extraction from product reviews, where we obtain interesting insights. The system while used on student discussions, uncover new and exciting patterns.

While each of the above problems is conceptually distinct, they share two key common elements – keywords and social data. Social data can be messy, hard-to-interpret, and not easily amenable to existing NLP resources. We show that our system is robust enough in the face of such challenges to discover useful and important patterns. We also show that the problem definition of keyword extraction itself can be expanded to accommodate new and challenging research questions and datasets.

CHAPTER 1

Introduction

Keyword Extraction (KE) is an important and well-studied problem in Natural Language Processing (NLP), with applications ranging from *online advertising* [2] and *clustering websites* [3] to *detecting named entities* [4] and *recommending academic papers* [5]. While Keyword Extraction has mostly been performed in the domain of academic papers (research articles; cf. [6, 7, 8, 9, 10]), it is of interest to observe how this problem can be worked out in other domains, e.g., social media. The reason keyword extraction has mostly been performed in the academic domain is because academic papers usually come with keywords already assigned by authors and/or readers. On the other hand, social media data is often messy and hard to interpret [11]. Their metadata can be complex (cf. Chapter 6), and the text data found on social media are often not amenable to existing NLP tools and resources.

In this dissertation, we circumvent the above problems by designing a keyword extraction system that not only outperforms the state-of-the-art in a challenging and new domain, but also retains robust performance on other social media data very different from the one the system was trained on. We examine the following three research questions:

1. **Can effective supervised and unsupervised methods be designed for extracting keywords from textual data, and can the core keyword extraction method be specialized to enable the extraction of important content related to undergraduate students, graduate applicants, and online consumers of products?** Over time, Keyword Extraction (KE) has spawned into two distinct and effective approaches – unsupervised KE and supervised KE. In the unsupervised KE, a candidate list of keywords is first extracted from documents based on heuristics, the list is then ranked by features such as term frequency and term frequency inverse document frequency (tf.idf), and the ranked list is finally pruned to retain keywords that are valid and that do not contain any obvious syntactic and semantic errors. Among ranking functions, graph-based methods such as PageRank and HITS have been especially popular [7, 12, 13] that work on graphs where nodes are word tokens, and edges are drawn

between consecutive tokens (or tokens within a specified window). In supervised KE, on the other hand, corpora are built that contain text documents, and keywords extracted by human judges on those documents. These corpora with their keywords are then used to train machine learning models, which predict whether a candidate phrase is a keyword on unseen data. In this dissertation, we devise several features, including novel graph-based features (e.g., coreness [14, 15]), for the supervised and unsupervised KE from text, and compare their performance on emails (Chapter 3). To be noted is the fact that graph-based features broadly denote the "centrality" of a word token in a given piece of text, thus quantifying its salience. We further show that Keyword Extraction can be specialized for sentences, and that the method when invested with several relevant features, can be used to identify usage expression sentences in consumer product reviews (Chapter 4) – effectively building a "usage gist" for a given product review. We also show that the method when constrained using an online encyclopedia, can be used to uncover important scientific topics in student statements of purpose (Chapter 5), and be used to predict the performance of students in a large undergraduate course while retaining pertinent concepts taught and discussed in the class (Chapter 6).

2. **Can the KE methods we designed be used to enable different NLP applications?** The ultimate reliability, validity and utility of a KE system depend upon downstream tasks – whether those tasks can leverage the extracted keywords to enable better human-computer interaction scenarios. We showed how the designed KE methods could be effectively used for (1) extraction of salient content from emails (Chapter 3); (2) identification of important specialized content from product reviews (Chapter 4); (3) extraction of important content from student applications and faculty papers for reliable student-faculty matching (Chapter 5); and (4) extraction of relevant content from student forum discussions for the purpose of predicting academic performance (Chapter 6).

3. **Can automatically extracted keywords be used to gain deeper insights into different data collections?** We used the keywords produced to understand the effect of in-domain training (Chapters 3 and 4); to understand keyphrase appropriateness and time taken to manually classify documents (Chapter 3); to look into the temporal flow of scientific topics (Chapter 5); to measure the research diversity, research focus, and content density of faculty members (Chapter 5); and to discern what topics successful students in a large undergraduate class write about (Chapter 6).

This dissertation is organized as follows: Chapter 2 discusses the background and re-

lated work on keyword extraction, followed by related work on each of the subsequent chapters. Chapter 3 goes into the design of our keyword extraction system trained on emails. We provide details on the data collection, inter-annotator agreement, feature engineering, unsupervised and supervised KE systems, feature ranking, effects of in-domain training and intersection gold standard, comparison with state-of-the-art, keyphrase appropriateness, and time taken to manually classify emails. Chapter 4 discusses the problem of identifying *usage expression* sentences in consumer product reviews. This is a completely new and fairly challenging NLP task. We design our own annotated dataset for this problem, which can be viewed as a conceptual extension of keyword extraction. Extensive feature engineering is performed, along with in-domain and cross-domain training in a supervised extraction framework. The final results outperform inter-annotator agreement. We report the learning curve, feature importance ranking, and detailed error analysis for future research.

Chapter 5 goes into the application of keyword extraction, where we match graduate applicants with faculty members to reduce the burden of Admissions Staff at a large midwestern university. We show the application of keyword extraction in the form of exploratory analysis and system design. In terms of exploratory analysis, extracted keywords are qualitatively shown to be very good indicators of faculty research trend over the years, and of the *content disparity* between faculty and applicants. In terms of system design, extracted keywords perform the best in five information retrieval scenarios consisting of graduate applicants and faculty members, and also help us obtain rankings of faculty members in terms of their research diversity, research focus, and content density. This system was used by our Department Admissions Staff for two consecutive years, and anecdotal evidence indicates that the system performed with excellence – often exceeding human intuition and expectations.

Chapter 6 presents our final piece of work on predicting student performance from the text data obtained from an online student question-and-answer forum of a large undergraduate class consisting of 600 students. The dataset was anonymized, and grades binned to obtain alphabetic grade buckets. We designed text-based, non-text-based, and hybrid features to predict the performance of students. Text-based systems (vector space and embedding models) outperformed the majority baseline by close to 9 percentage points, and the hybrid text-non-text system outperformed the majority baseline by 12.50 percentage points. Most of the keyword-based vector space and embedding representations were able to outperform the majority baseline, the best one by 3.13 points. We also present our results on correlation of student performance with text-based features, viz. keywords, stopwords, and LIWC (Linguistic Inquiry and Word Count; cf. [16]) categories, for the purpose of

uncovering important details – words and phrases correlating with high performance, low performance, and not correlating with performance, respectively.

Chapter 7 concludes the dissertation with our contributions, limitations, and future work.

It is of special interest to deliberate upon the value of this work with respect to recent developments in deep learning, esp. as some of our proposed methods use embeddings (cf. Chapters 4 and 6). Note that there are two aspects of the question: (a) are embeddings necessary and sufficient for Keyword Extraction? and (b) are embeddings necessary and sufficient for applications of Keyword Extraction? The answer to both these questions, we opine, is that Keyword Extraction and embeddings are complementary techniques that can re-inforce each other. Embeddings are powerful models that do not require manual engineering and tuning of features. They do, however, suffer from a "black box" approach that raises interpretability concerns. Further, it is much more difficult to examine embeddings and separate the "wheat from the chaff", in the sense that it is difficult to understand and separate the useful dimensions from the strictly non-useful ones. And this is precisely where keywords help us tremendously. As has been shown in all the subsequent core chapters of the dissertation (except Chapter 4), keywords give us a ready visual appreciation of the problem at hand (also cf. [17]), which we will be very hard-pressed to obtain from embeddings. Embeddings – in other words – act as possibilities; whereas keywords act as concepts. They are inter-related and mutually reinforce each other, rather than destroying each other's benefits.

In fine, we would like to give a little thought to the core problem addressed in this dissertation, and individual research directions. Note that the core problem is Keyword Extraction, applied in novel domains. We designed the KE system with email data in mind, and later showed the validity and reliability of the extracted keywords in three different tasks that are unrelated to emails. While the tasks are all reasonably different, there is one common element: social media. We showed that our KE pipeline can successfully delve into, and deal with such messy and hard-to-interpret social media data, and still come up with meaningful and effective keywords that show reasonable performance in downstream prediction tasks. This dissertation, therefore, manifests a co-emergence of three different yet inter-related elements: social media, keyword extraction, and novel downstream applications (prediction tasks).

CHAPTER 2

Related Work

Keyword extraction usually proceeds in three steps: candidate extraction, ranking/classification, and post-processing. In the candidate extraction step, potentially important phrases are identified and extracted from the documents. In the ranking/classification step, these candidate terms are either ranked according to some ranking function derived from the document structure, or they are classified as to whether they represent key terms or not. In the post-processing step, top k terms from the ranked list (or terms that are classified as keywords) are semantically normalized to yield a set of phrases so that each denotes a single concept.

In practice, there are some good heuristics for candidate extraction. Hulth [6] noted that base noun phrases often constitute a predominant form of keyphrase. She further leveraged previous studies in observing that specific patterns of part-of-speech tags are beneficial for keyword extraction. Csomai and Mihalcea [18] experimented with stopword-filtered n-grams and named entities as potential keywords.

The second step – ranking/classification – is trickier, because it is not immediately obvious what ranking function or phrase features to use. In a study by Hasan and Ng [9], *tfidf* was shown to be a surprisingly robust candidate. While conceptually simple, it beats other more complex ranking strategies. Other important features for keyword classification include *tfidf* [8, 19, 20], *first occurrence position* of the phrase [6], *capitalization* [20], *phrase length* (in words) and *is-in-title* [19].

Two salient groups of ranking functions deal with the *phraseness* and *informativeness* of a candidate phrase [21]. Informativeness denotes how much information content a standalone phrase carries with it, and is usually determined by the number of occurrences of that phrase in a *background corpus* [21]. Phraseness, on the other hand, is a measure of how cohesive or *tightly-linked* a phrase is; in other words, whether the constituent words of a phrase come together more often than by chance. Phraseness is estimated based on the co-occurrence frequency of words in a *foreground corpus*. A final ranking of phrases is

produced by some linear combination of phraseness and informativeness scores. Tomokiyo and Hurst [21] proposed the use of *language models* in estimating phraseness and informativeness, whereas Csomai and Mihalcea [18] used chi-squared test. A different formulation is that of *keyphraseness*, proposed by Mihalcea and Csomai [22], where the probability of a word or a phrase to be linked to a Wikipedia article, calculated across the entire Wikipedia, is used as an indication of how likely that word or phrase is to be selected as a keyword.

Note that the above-mentioned phrase ranking strategies are mostly *ad hoc*, and they emerged as a way to heuristically assess the purported importance or relevance of a phrase. There is, however, a completely different class of ranking algorithms that look into this problem from a more cognitively appealing standpoint. These algorithms look into the structure of *word co-occurrence networks*, where nodes are word types and edges are word collocations. Mihalcea and Tarau [7] introduced TextRank and observed that in these networks, important words can be thought of as being *endorsed* by other words, and this leads to an interesting phenomenon. Words that are most important, viz. keywords, emerge as the *most central words* in the resulting network, with high degree and PageRank [23]. A stream of studies ensued after the seminal work of TextRank (see Hasan and Ng [9] for a detailed comparison). While most looked into variants of PageRank, Litvak and Last [13] experimented with the HITS algorithm [24], while Boudin [25] investigated other indices like degree, betweenness and closeness.

The final important step in keyphrase extraction is *post-filtering*. Extracted phrases are disambiguated and normalized for morpho-syntactic variations and lexical synonymy [18]. Adjacent words are also sometimes collapsed into phrases, for a more readable output.

Benchmark datasets for keyword extraction include ICSI – a collection of meeting transcripts divided into 201 segments [26], NUS – a set of 211 academic papers [8], INSPEC – 2,000 titles and abstracts from journal papers [6], and SEMEVAL – 184 academic papers from SEMEVAL 2010 Keyphrase Extraction Task [10]. Given the preponderance of keyword-annotated datasets in the academic domain, most research in keyword extraction has focused on academic papers.

Keyword Extraction continues to be a hot research topic in recent years, with many articles published every year. Here we briefly describe a sample of recent research on Keyword Extraction (KE), organized and synthesized along three principal dimensions: approach, applications, and new tasks. After that, we discuss related research on the four following chapters which form the core of this dissertation.

2.1 Approach

Among new approaches proposed for Keyword Extraction, Wang et al. [27] presented a graph-based keyword ranking strategy using information supplied by word embedding vectors as background knowledge. A weighting scheme was used to compute informativeness and phraseness scores of words using information supplied by word embedding vectors and local statistics. A weighted PageRank algorithm then computed the final scores of words. The work was evaluated on Hulth2003, DUC2001, and SemEval2010 datasets, and evaluation results were comparable to state-of-the-art algorithms.

Bougouin et al. [28] proposed a method for performing keyword extraction and keyword assignment (assigning keywords to a document from a controlled domain-specific vocabulary) in an integrated and mutually reinforcing manner, using graph co-ranking. The method, TopicCoRank, built two graphs: one with document topics and one with controlled keyphrases (training keyphrases). A strategy was designed to unify the two graphs and rank by importance topics and controlled keyphrases using a co-ranking vote. Experiments were performed on three datasets covering different domains of humanities and social sciences, and statistically significant improvements were observed compared to both keyword extraction and keyword assignment state-of-the art methods. Results showed that the approach benefited from both controlled keyphrases and document topics, improving both keyword extraction and keyword assignment baselines. TopicCoRank was able to annotate keywords in scientific domains similar to professional indexers.

Florescu and Caragea [29] proposed a novel position-biased PageRank algorithm for keyphrase extraction. The method – PositionRank – was unsupervised and graph-based, and incorporated information from all positions of a word's occurrences into a biased PageRank to extract keyphrases. The model obtained good improvements in performance over strong baselines. Implicitly, PositionRank incorporated both the relative position and the frequency of a word into a biased PageRank. Experiments conducted on two datasets (KDD and WWW) showed that PositionRank achieved better results than strong baselines, with improvements in performance as high as 26.4%.

Li et al. [30] extracted keywords as bigrams from text documents using integer linear programming (ILP) approach. They used syntactic information to filter and select bigrams, external resources (e.g., word embedding from additional corpus, Wikipedia, Dbpedia, WordNet, and SentiWordNet) to extract features, and a joint learning process for weight training using discriminative learning. Among the internal features based on the test documents were document frequency and bigram positions. The learning model predicted bigram weights and selected summary sentences using the ILP framework at the

same time. The system consistently outperformed prior ILP methods on different TAC datasets, and performed competitively compared to the state-of-the-art.

2.1.1 Neural Network-based Approaches

With the recent surge of interest in deep learning and neural networks [31], it is no surprise that Keyword Extraction has seen neural networks being applied on text documents in novel ways.

Aquino and Lanzarini [32] presented an algorithm for keyword extraction from documents written in Spanish. The algorithm combined autoencoders (good for highly unbalanced classification tasks) with the discriminative power of conventional binary classifiers. In order to improve its performance on larger and more diverse datasets, the algorithm was used to train several models of each kind (autoencoder and conventional binary classifier) using *bagging*. The use of autoencoders captured the properties of important terms, yielding comparable or better results than other well-known keyword extraction algorithms.

Wang and Zhang [33] presented an automatic keyword extraction method based on a bidirectional long short-term memory (LSTM) recurrent neural network (RNN). The results of experiments conducted on product reviews obtained by crawling jd.com (a Chinese website comprising consumer reviews) showed that the proposed approach had a high accuracy for keyword extraction.

2.2 Applications

Keyword Extraction (KE) is an NLP problem that not only engenders new thought waves in terms of methods, but also in terms of downstream applications that change the way we think about text data in particular, and human mind in general.

Horita et al. [34] proposed a system for the extraction of keywords from a document for Wikification [22]. They focused on East Asian language documents and experimented with Japanese text. Their method consisted of two steps. In the first step, they extracted nouns from a document using a morphological analyzer, and extracted candidate keywords by a method called Top Consecutive Nouns Cohesion (TCNC), which connects contiguous nouns and treats them as one compound word. In the second step, they ranked the extracted candidate keywords using two measures for keyword importance, the Dice coefficient and Keyphraseness. They showed that the combination of TCNC and Keyphraseness achieved the best results. Experiments were conducted to verify the effectiveness of the proposed method using 10 articles in Japanese Wikipedia.

Lauscher et al. [35] proposed a combination of two techniques, called Entity Linking and Labeled LDA, in order to address the interpretability issue of topic models. The method identified in an ontology a series of descriptive labels for each document in a corpus, and then generated a specific topic for each label. Having a direct relation between topics and labels made interpretation easier; whereas, using an ontology as background knowledge served to limit label ambiguity. Since the topics were described with a limited number of clear-cut labels, they promoted interpretability and supported quantitative evaluation of the obtained results. The potential of the approach was illustrated by applying it to three datasets, the transcription of speeches from the European Parliament fifth mandate, the Enron Corpus and the Hillary Clinton Email Dataset. In order to estimate the quality of the approach the authors developed an evaluation platform that allowed to have a precise overview of the performance and the drawbacks of each step of the approach: label identification via Entity Linking, label ranking and selection, and the assignment of entity-labels to topics.

Paramonov et al. [36] designed a thesaurus-based method for increasing text-via-keyphrase graph connectivity during keyword extraction for e-Tourism applications. The goal was to solve the task of automatic extraction of keyphrases from a text corpus relating to a specific domain so that the texts linked by common keyphrases would form a well-connected graph. The authors developed a new method that used a combination of well-known keyphrase extraction algorithms (TextRank, Topical PageRank, KEA, Maui) with a thesaurus-based procedure that improved the text-via-keyphrase graph connectivity and simultaneously raised the quality of the extracted keyphrases in terms of precision and recall. The effectiveness of the proposed method was demonstrated on the text corpus of the Open Karelia tourist information system.

Chen et al. [37] developed a supervised natural language processing (NLP) system called Finding impOrtant medical Concepts most Useful to patientS (FOCUS) that automatically identified and ranked medical terms in electronic health record (EHR) notes based on their importance to the patients. An expert-annotated corpus was built, where for each EHR note, two physicians independently identified medical terms important to the patient. Using the physicians' agreement as gold standard, FOCUS was developed and evaluated. FOCUS first identified candidate terms from each EHR note using MetaMap and then ranked the terms using a support vector machine-based learning-to-rank algorithm. Among the features used were: distributed word representation, Unified Medical Language System semantic type, topic features, and features derived from consumer health vocabulary. FOCUS was compared with two baseline NLP systems with statistically significant improvements.

2.3 New Tasks

One of the principal strengths of the idea of Keyword Extraction is that it can be used and re-used in modern NLP tasks that not only stretch the boundary of conventional NLP techniques in terms of the pipelines used, but also offer unique opportunities to peep into human behavior that would have otherwise remained unexposed.

Siddiqui et al. [38] introduced a new task called *Facet Extraction*. Given a collection of technical documents (patent folios, legal cases, real-estate agreements, historical archives, and scientific literature), the goal of Facet Extraction was to automatically label each document with a set of concepts for the key facets (e.g., application, technique, evaluation metrics, and dataset). Major challenges in performing Facet Extraction were: concept extraction, concept to facet matching, and facet disambiguation. The authors developed FacetGist, a framework for facet extraction, that constructed a graph-based heterogeneous network to capture information available across multiple local sentence-level features, as well as global context features. A joint optimization problem was then formulated, and an efficient algorithm was proposed for graph-based label propagation to estimate the facet of each concept mention. Experimental results on technical corpora from two domains demonstrated that Facet Extraction could lead to an improvement of over 25% in both precision and recall over competing schemes. The framework was weakly supervised, and integrated local context signals (e.g., relation phrases, concept suffix) with global structure signals (e.g., paper sections, and topics).

Chen et al. [39] introduced mobile app tagging – assigning a list of keywords indicating the core functionalities, main contents, key features or concepts of a mobile app. Mobile app tags are useful for app ecosystem stakeholders or other parties to improve app search, browsing, categorization, and advertising. The authors proposed a novel auto mobile app tagging framework for annotating a given mobile app automatically, based on a search-based annotation paradigm powered by machine learning techniques. Specifically, given a novel query app without tags, the proposed framework first explored online kernel learning techniques to retrieve a set of top-N similar apps that were semantically most similar to the query app from a large app repository; and then mined the text data of both the query app and the top-N similar apps to discover the most relevant tags for annotating the query app. To evaluate the efficacy of the proposed framework, the authors conducted experiments on a large real-world dataset crawled from Google Play. The results were encouraging, and demonstrated that the technique was effective and promising.

A task on keyword extraction from scientific publications was organized as part of SemEval 2017 (Augenstein et al. [40]) that focused on extracting keyphrases and the relations

between them from scientific documents, with a view to understanding which publications described which processes, tasks and materials. The task was novel, and there were 26 submissions across 3 evaluation scenarios spanning 3 subtasks (mention-level keyphrase identification, mention-level keyphrase identification classification, and mention-level semantic relation extraction). Successful systems varied in their approaches, most using recurrent neural networks (RNNs), often in combination with conditional random fields (CRFs) as well as convolutional neural networks (CNNs). Identifying keyphrases was the most challenging subtask, since the dataset contained many long and infrequent keyphrases.

2.4 Related Work on Keyword Extraction from Emails

There are only three previous studies that we are aware of that considered keyword extraction from emails. Turney [41] reports a study that pioneered email keyword extraction, but his dataset has not been released. Goodman and Carvalho [42] worked with emails, but since their goal was to extract *implicit search queries* from emails, and not keywords, their dataset is not useful to us. Dredze et al. [43] extracted summary keywords from emails using latent concept models, and evaluated the extracted keywords in two novel tasks – *automated foldering* (predicting which folder an email should go to), and *recipient prediction*. While Dredze et al.'s study did look into (unsupervised) email keyword extraction, they performed an extrinsic evaluation of their approach rather than intrinsically evaluating on a gold standard dataset.

Also relevant is the work by Laclavík and Maynard [44], who discuss general strategies for email classification, storage, and integration with other information management systems. They further point out that email communication in a modern organization is mostly *action-oriented*, and that knowledge workers of all kinds interact with their emails on a daily basis. This stands in sharp contrast with keyword extraction in the academic domain,[1] where papers are meant for other researchers who are reasonably familiar with the domain – and therefore use formal and scientific vocabulary. Furthermore, academic communication via papers is *single-way* (author to audience), and *dissemination-oriented*. Also, unlike academic papers, emails mostly discuss topics *at hand*, including urgent ones. Hence, emails stand to benefit from their own keyword extraction system. In fact, Laclavík and Maynard briefly hinted at email keyword extraction as a way to combat the email information overload (cf. Section V).

[1] On which the current state-of-the-art is based [10].

2.5 Related Work on Usage Expressions

Existing research could be organized into six self-consistent psycho-sociological theories, namely psycho-analysis, social theories, stimulus-response theories, trait and factor theories, self-theories, and life style theories. [45] offers a comprehensive review of the literature on consumer behavior and psychological traits. [46] found weak relationships between opinion leadership and innovative buying behavior, but observed that the relationship *strength* varied by product category. [47], and [48] showed that there were correlations between personality traits and the *types* of products used. [49] posited that products as *symbols* were organized into congruent relationships with the consumer's *self-image*. More recently, [50] found that people preferred products with a *product personality* that matched their self-image, and the positive effect of product-personality congruence was independent of user-image congruence.

In natural language processing research, the closest problem to usage expressions is perhaps that of opinion mining from product reviews and product aspects. [51] classified reviews as expressing positive or negative sentiment. They identified four problems with review classification, including rating inconsistency, ambivalence, data sparseness, and skewed distribution. [52] extracted product features from the reviews of a single product, taking user opinion into account. Opinion/product features were mined if a reviewer had commented on them. [53] presented OPINE, an unsupervised information extraction system that mined reviews in order to build a model of important product features, their evaluation by reviewers, and their relative quality across products. OPINE's use of *relaxation labeling* led to strong performance on the tasks of finding opinion phrases and their polarity. [54] presented a "*holistic lexicon-based approach*" for mining *context-dependent* opinion words. The proposed method used an aggregating function for multiple conflicting opinion words in a sentence. The authors further implemented a system called "Opinion Observer" based on their method. Lastly, [55] implemented a special dependency parser for opinion mining that used phrases (rather than words) as the primitive building blocks. Since many product features are in fact phrases, this approach led to good results for extracting relations between product features and opinion expressions.

Yet another related task is that of mining *semantic affordances* [56]. In this task, "usage" of a product can be viewed as an *action* performed on an *object* with the help of the *product*. Relationships between such actions and objects are known as "semantic affordances". As Chao et al. showed, text mining can be very effective at ascertaining affordance relationships between verb and noun classes. Similar verb-noun relationships have also been formulated in the problem of learning *selectional preferences* from text

[57, 58, 59, 60, 61, 62], and more generally, in the problem of *probabilistic frame induction* [63, 64, 65].

Another topic of research related to our work is the problem of *research idea extraction* from academic papers. [66] took the first stab at this problem by implementing a bootstrapping algorithm on dependency tree kernels. Gupta and Manning's method was later refined by [67] who worked with a more crisp set of *idea categories*. We view this problem as conceptually parallel to ours; however, a key difference is that usage expressions are typically more obscure in text as compared to research ideas.

2.6 Related Work on Matching Graduate Applicants with Faculty Members

The problem of matching graduate students with faculty members has three close analogs in natural language processing: authorship attribution, author profiling, and author-topic modeling.

In authorship attribution, the goal is to predict who authored a particular document. The problem is usually cast as a classification task, where we have a large set of training documents with known authors, and a smaller set of test documents with unknown authors. Machine learning models are trained on the training documents, and then deployed on test documents to predict the unknown authors. For details on authorship attribution, please see the surveys by Juola [68], Stamatatos [69], and Koppel et al [70]. In some flavors of authorship attribution, test documents are used as search queries against training documents, and the author of the top-ranked (training) document is considered predicted label [69]. In our study, we consider papers written by faculty members as "training documents", and statements of purpose written by students as "test documents". Performance on the test set is judged based on the ground truth faculty-applicant pairing we have. A potential limitation of this approach comes from the fact that in authorship attribution, we would like to uncover the *writing style* of an author, whereas in this case, we are interested in the *content match* between a paper written by a faculty member, and a statement of purpose authored by an applicant. We resolve this issue by using keywords (cf. Sections 5.3 and 5.4).

Author profiling is very similar to authorship attribution, except that the goal here is to build a "stylistic profile" of an author instead of predicting a class label. The profile is usually a vector of words and/or phrases frequently used by the author, and may also include grammatical constructs and parse tree fragments. An author is represented by several vectors that are built on documents written by him/her. These vectors can be used to identify

the author's unique writing style (*fingerprint*) and to extract other useful properties such as gender, age, education, and personality traits. Author profiling has been discussed in depth in the survey by Stamatatos [69]. In our case, author profiling could serve as a fundamental building block where papers written by faculty members are used to create their authorship profiles, and then a statement of purpose that is most similar to a faculty member's profile, is assigned the corresponding faculty member. This approach, albeit sound in principle, has the same important drawback as authorship attribution; it focuses on *stylistic* rather than content information, and is therefore not very useful.

Content information of authors can be explicitly incorporated in a probabilistic setting, where documents are modeled as a collection of topics, and topics are modeled as a collection of words. Topic generation depends on authors represented as (observed) random variables in the model [71]. An unseen document can be assigned a probability distribution over authors and topics, thereby helping find out which authors are the most likely to have written that document. In our case, we could use the set of papers written by faculty members to train an author-topic model, and then the statements of purpose could be "folded in" the model to extract their most representative author and topic probability distributions.

While all the above ideas are good, we did not find an approach that closely matches our purpose. The only similar study we found comes from IBM India Research Lab [72]. They designed a system called "PROSPECT" to screen candidates for recruitment. Their system combines elements from recommender systems, information retrieval, and author profiling to come up with a software and graphical user interface that improves candidate ranking by 30% and provides faceted search functionality to conduct fine-grained analyses such as highest degree of the candidate, relevant and total work experience, skills, and his/her city of residence. Since companies like IBM receive thousands of job applications for many job postings, it becomes crucial to augment the slow and cumbersome manual candidate-screening process with an automated decision-making tool such as PROSPECT. Our use case is also very similar, in that we want to screen hundreds of graduate applicants and match them with potential advisors. In our case, faculty members serve the same purpose as human resources staff screening job postings, and graduate applicants are similar to job candidates. Inspired by PROSPECT, we pursued five keyword-based approaches to tackle this problem. All approaches use information retrieval techniques, and stand to benefit from *learning to rank*, given enough data [73].

2.7 Related Work on Student Performance Prediction

Performance of students in a course, esp. a large one, depends on many factors. Existing research on student performance prediction can be broadly categorized into two dimensions – one that relates and predicts performance with non-textual data produced by students, and another that does so with the textual data students authored. While our goal in this chapter is to predict student performance from text data, in the following two subsections we describe related studies on student performance prediction from both non-text as well as text data. The first subsection deals with non-text studies, and the second one with text-based prediction of performance.

2.7.1 Student Performance Prediction from Non-textual Data

Numerous research articles have focused on predicting student performance in Learning Management Systems (LMS) and Massive Open Online Courses (MOOC). Some of them have used historical grade information from transcripts [74, 75], student demographics [74, 75, 76, 77], course characteristics [75, 78], instructor information [74, 75], student scores in early performance assessments [74, 76, 79], learning activities such as course video watching [77, 78, 80], metadata from student interaction via LMS and MOOCs [74, 77, 81], and metadata from student forums [76, 77]. We will describe these articles in more detail.

Generally speaking, *student forum analytics* have recently emerged as a flourishing area of research, and a detailed literature review has been presented by Sander [82] on three aspects of student forum analytics: (a) reducing attrition of first year students, (b) inferring the time when learning analytics produce best results, and (c) predicting academic outcomes using learning analytics – based on Learning Management System (LMS) data. Romero and Ventura [83] have given another literature review on the application of Educational Data Science (EDS) in MOOCs, describing and grouping the main studies by the task or issue to be solved, along with the techniques used.

Sweeney et al. [75] developed a system to predict student grades (numeric) in the courses they would enroll in during the *upcoming* semester by learning patterns from historical transcript data, student demographics, course characteristics, and instructor information. Strong connections between instructor characteristics and student performance were found. The data came from five years of courses taught at George Mason University, consisting of more than 30,000 students and 9,000 courses. Matrix factorization algorithms were used to find connections between students and courses, with factorization machines [84], random forests and personalized multi-linear regression [85] models achieving the

lowest prediction error.

Meier et al. [79] proposed two related algorithms (classification and regression) to predict the final grade of students in a course. The features were: student scores in early performance assessments such as homework assignments, quizzes and midterm exams. A confidence estimate for the prediction accuracy was derived and its utility demonstrated on a dataset of about 700 University of California Los Angeles (UCLA) undergraduate students in an introductory Digital Signal Processing course over seven years. For 85% of the students, the system was able to predict with 76% accuracy whether they were going to do well or poorly in the course after the fourth course week. Predictions were robust even when the course was taught by different instructors. In-class exams were found to be better predictors of the overall performance of a student compared to homework assignments.

Elbadrawy et al. [74] used personalized multi-linear regression (PLMR) from their prior work [85], factorization machines [84], course-specific regression, and course-specific matrix factorization (MF) approaches to predict student grades (numeric) in future courses as well as on in-class assessments. Using only historical grade information coupled with available additional information such as transcript data, both PLMR and MF techniques were able to predict next-term grades with lower error rates than other methods such as factorization machines. PLMR was further useful for predicting grades by incorporating features captured through student interaction with LMS and MOOC server logs. Four datasets were used in this study with varying characteristics: George Mason University (GMU) transcript data, University of Minnesota (UMN) transcript data, UMN LMS data, and Stanford University MOOC data. The task was framed as a regression problem, and features were different for the four datasets, which were broadly based on student demographics, instructor qualifications, student performance (GPA and homework grades), and student interaction with the LMS.

Cen et al. [81] addressed three important aspects of collaborative learning [86] on quantitative evaluation and prediction of group performance. (a) They first explored using machine learning to predict group performance based on member interactions data, and sought to identify whether, and to what extent, group performance is driven by specific patterns of learning and interaction. Extreme learning machine [87], and classification and regression trees [88] were used to predict group academic performance from live interaction data. (b) A comparative model was designed to unscramble individual student performances within the group. These performances were then used in a generative mixture model of group grading and compared against the actual group performance to define *collaboration synergy*, the improvements owed to collaborative learning. (c) The impact of *group composition* was evaluated in terms of gender and skills on learning performance

and collaboration synergy. The analysis indicated a high level of predictability of group performance based solely on the style and mechanics of collaboration, and quantitatively supported the claim that heterogeneous groups with a diversity of skills and genders benefit more from collaborative learning than homogeneous groups. The data came from 168 students (72 student groups) in Fall 2013 using a collaborative learning environment tool. The task was formulated as both classification and regression.

Mueen et al. [76] used machine learning to predict and analyze student academic performance based on their academic record and forum participation. Student data were collected from two undergraduate courses from two semesters (Programming Fundamentals and Advanced Operating Systems courses; August 2014 to May 2015). Decision tree (C4.5), multilayer perceptron, and Naïve Bayes (NB) classifiers were used. NB outperformed the other two classifiers by achieving overall prediction accuracy of 86%. The collected data consisted of 60 students, of which 41 passed (68.33%), and 19 failed (31.66%). 38 features were extracted for each student, consisting of student demographics, online student forum metadata, and prior grades received. The task was formulated as pass/fail prediction.

Xu and Yang [78] presented a grade prediction algorithm using student activity features that predicted whether a learner would be able to pass a certification test. The method consisted of a two-step classification process: motivation classification (MC) and grade classification (GC). The MC divided all learners into three groups – certification-earning (i.e., high motivation), video-watching (middling motivation), and course-sampling (low motivation). The GC then predicted whether a certification-earning (i.e., high-motivation) learner would or would not be able to obtain the certification. Prediction accuracy improved due to the fact that the parameters of classification model could be tuned at a finer granularity to fit more learners. The dataset came from 10 courses of the Person-Course Dataset (AY2013; cf. [89]) that included course information (such as course ID, open date, launch date), and learner activities such as video play activity and course forum activity. The prediction algorithm was SVM [90], and six features were used: number of course videos played, number of posts, number of active days, number of days between first event and course opening date, number of days between last event and course closing date, and number of enrolled courses.

Widyahastuti et al. [80] proposed a model to predict student performance from online discussion forums. The data was extracted for 165 students in an *English for Librarians* course from an online discussion forum (E-learning log) at Open University, Indonesia. The authors focused on the features in online discussion forums that were most predictive of student performance. The features explored were: course module instance list viewed,

discussion created, discussion subscription created, discussion subscription deleted, discussion viewed, module course viewed, post created, post deleted, post updated, some content has been posted and user report viewed. The task was formulated as a simple linear regression to predict the numeric grades of individual class assignments. Among all features, discussion created, discussion subscription created, module course viewed, and some content has been posted – had a statistically significant relationship with assignments 1, 2, and 3 in terms of grade prediction. Note that while Widyahastuti et al.'s study is similar to ours, we focus mostly on text features rather than non-text features, and use classification rather than a regression setup.

Qiu et al. [77] used data from *xuetangX*,[2] one of the largest MOOCs from China. In-depth analysis conducted for student demographics and learning activity patterns in course forums, videos and assignments showed significant behavioral heterogeneity in students' course selection as well as learning patterns. For example, students who exerted higher effort and asked more questions were not necessarily more likely to get certificates. Additionally, the probability that a student obtained the course certificate increased three times when (s)he had one or more "certificate friends" (other students who finally got certificates). A unified latent dynamic factor graph (LadFG) model was developed to predict students' learning effectiveness in a classification framework (assignment grade prediction), by incorporating demographics, forum activities, and learning behavior as features. The proposed model significantly outperformed (+2.03-9.03% by F1-score) several alternative methods (SVM [90], Logistic Regression [91], and factorization machines [84]) in predicting performance on assignments and course certificates.

2.7.2 Student Performance Prediction from Textual Data

Note that there are no studies, to the best of our knowledge, that attempt to predict student performance in a course from their writings. The closest study we found comes from Chen et al. [92], who focused on journal writing, an important and common practice in education. Students' reflection journals offer a rich source of data. The authors collected 367 journals from 80 students in different sections of an undergraduate course to educate pre-service teachers.[3] They proposed a method based on topic modeling for the task of *themes exploration* and *reflection grade prediction*. The method was evaluated on the collected sample of journal writings. The topic modeling method was able to discover important emerging themes and patterns in the reflection journals. Weekly topic relevance and word count were identified as two important predictors of journal grades. Prediction models were developed

[2] http://www.xuetangx.com/.

[3] https://en.wikipedia.org/wiki/Student_teacher.

for the grading of reflection journals, where the task was a balanced three-class classification with term frequency and weekly content relevance as features. Naïve Bayes gave the best accuracy (65.10%). Note that while this method was based on text data, it differs from ours because we are dealing with text data from *student forums*, not journals. Journals tend to be longer, more developed, and follow the topics that the author feels interested in; whereas forum posts tend to be shorter, and mostly on topics selected by others. Further, our dataset is much larger (378 students as opposed to 80).

More generally, the field of Automated Essay Scoring [93] deals with automated grading of essays written by students on a given prompt.[4] The essays are typically long and involved, and receive a numeric score. By contrast, in this chapter we focus on student forum data (as opposed to essays), the goal is to grade the students over the course of a full semester (as opposed to a single essay), and the task is 4-class classification rather than the prediction of a numeric grade. This is a new task, and has not been attempted previously.

[4] https://en.wikipedia.org/wiki/Automated_essay_scoring.

CHAPTER 3

Keyword Extraction to facilitate information access: Data-driven keyword extraction method, applied to emails

Emails constitute an important genre of online communication. Many of us are often faced with the daunting task of sifting through increasingly large amounts of emails on a daily basis. Keywords extracted from emails can help us combat such information overload by allowing a systematic exploration of the topics contained in emails. Existing literature on keyword extraction has not covered the email genre, and no human-annotated gold standard datasets are currently available. In this chapter, we introduce a new dataset for keyword extraction from emails, and evaluate supervised and unsupervised methods for keyword extraction from emails. The results obtained with our supervised keyword extraction system (38.99% F-score) improve over the results obtained with the best performing systems participating in the SEMEVAL 2010 keyword extraction task.

3.1 Introduction

With 144.8 billion emails sent every day around the world,[1] emails represent an essential mode of digital communication. The market share of emails is tremendous, and largely untapped. Not only are traditional applications of keyword extraction important for emails, but emails present a unique scenario in their own right. According to Wasserman, up to 28% of workers' time is spent checking emails,[2] and most work emails are not important.[3] It is therefore paramount that we be able to somehow sort this enormous pile of emails into

[1] http://mashable.com/2012/11/27/email-stats-infographic/
[2] http://mashable.com/2012/08/01/email-workers-time/
[3] http://mashable.com/2012/06/07/most-work-emails-not-important-study/

a workable collection so that the more important ones are dealt with immediately, whereas others are relegated to future inspection [44].

While all existing email clients include some form of free text search to help users identify relevant threads of conversation, keyword extraction from emails can help us spot salient phrases from emails, thereby automatically tagging/categorizing them into appropriate folders (or "labels"), and in effect, giving us a *faceted search* functionality complementary to the vanilla text search on emails.

In this chapter, we address the task of automatic keyword[4] extraction from emails, as a way to automatically annotate email content with salient words or phrases that can help us decide on the importance or relevance of an email or thread. The availability of keywords could facilitate the access to email on mobile devices, and they could be used as a preprocessing step for smart email applications that aim to classify or prioritize emails. Keywords could help visualize emails in the form of "keyword clouds" with larger keywords indicating more salience [17], and the clouds can be interactive so that when a user clicks or taps on a certain keyword, all emails pertaining to that keyword get retrieved. Different colors indicate different "keyword topics", so that all keywords under a certain topic get grouped together in one area of the cloud.

Furthermore, keywords form their own social networks [2], and important meta-information about documents can be mined by looking into networks of keywords. Keyword networks also serve as a powerful visualization tool by themselves, so that users who are interested in relationship (or association strength) between two keywords or keyword cliques may benefit from looking at such visualizations.

Keyword extraction is an important problem in natural language processing, where the goal is to identify the most important words and phrases in a document. The keywords can either serve as a short summary of the document, giving users an overview of its contents; or they can indicate the topics that are being discussed.

While it may be argued that keywords are often an impoverished representation of the underlying topic space, and that there are alternative models that capture such spaces [94, 43, 95], it is important to consider that probabilistic topic models need to be trained on large corpora, and they often suffer from scalability issues. Keyword extraction, on the other hand, can be performed in both supervised and unsupervised fashion, and most keyword extraction methods do not need large training corpora.

Keyword extraction has traditionally been the domain of librarians and book indexers, but more recently the problem has seen a number of novel applications. For instance, keywords have been used to thematically group web sites [3], where authors reverse-

[4]We use the term "keyword" to refer to key words or phrases.

engineered a graph of webpages by clustering them, and using keywords to label the clusters. Keywords were used as *anchor phrases* to link to Wikipedia articles in [22], and as summary topics to visualize how topics change over time in online Korean news articles [96]. Keywords have been used to target advertisements on webpages [97], and as indicators of academic paper content and user interest in a content-based paper recommendation system [5].

With such a large number of applications, it is surprising that keyword extraction from emails has not received much attention from the research community. In this chapter, we introduce a new dataset, consisting of single and thread emails manually annotated with keywords, and describe a number of features that can be used for unsupervised or supervised email keyword extraction. Through several evaluations, we show that we can achieve results that significantly improve over several baselines, and also improve over state-of-the-art systems participating in the SEMEVAL 2010 keyphrase extraction task [10].

Note that working with email data raises a valid concern about user privacy, and whether we are in a position to violate such concerns. To this, we respond: since our study is based on the publicly released Enron corpus [98], we do not expressly deal with such concerns. It is, however, of importance to deliberate upon potential privacy breaches that may happen as an application of the technology developed and described in this chapter. We opine that such cases, if any, would need to be dealt with on a case-by-case basis, as no single scheme could benefit all the parties of interest in a potential breach of privacy. This is also a question that ties into the legality of the research we perform; we ensure the reader that our research is always protected by Institutional Review Board (IRB) standards and compliance, which takes privacy into account quite explicitly. Furthermore, we specifically removed all instances of header text and footer text of emails that include personal names – to the extent possible. We also manually anonymized – to the extent possible – all the person names mentioned in emails.

3.2 Keyword Extraction Pipeline

Our keyword extraction systems proceed in five stages:

1. Email processing
2. Candidate extraction
3. Pre-processing
4. Ranking/Classification

5. Post-processing

As a first step, we sentence-segment each email manually, followed by tokenization based on whitespace. We ignore email metadata such as filename, ID, date, from and to, subject, and signature fields. This was done to ensure that our systems are only focusing on the email text. We also remove numbers and words consisting of one or two characters.

In the candidate extraction stage, we generate candidate phrases from a document. We experimented with four types of phrase candidates, and found noun phrases and named entities to be the best (Section 3.3.2).

In the pre-processing stage, we clean up the phrase candidates by removing punctuation, folding to lowercase, and removing numbers and leading and trailing stopwords. We also implemented a pre-processing heuristic, which is a syntactic filter that only considers nouns and adjectives while constructing the word co-occurrence network. This is based on the observation that most keywords consist of nouns and adjectives (along with function words), and therefore a part-of-speech filter at this stage can help eliminate some of the potential noise.

In the fourth and most important stage, we extract keywords from emails using two approaches – unsupervised, and supervised. In the unsupervised (ranking) approach, we (a) rank words using several linguistic and centrality-based features, and then *collapse* the top-ranked adjacent words to form keyphrases; (b) rank *candidate phrases* (noun phrases and named entities) using several phrase features – both linguistic and centrality-based, and then extract the top-ranked phrases as keyphrases. In the supervised (classification) approach, we classify candidate phrases as *keyphrase* vs. *non-keyphrase* using phrase features, and return the ones classified as *keyphrase*. Both approaches are evaluated on our own dataset consisting of 319 keyword-annotated emails.

In the fifth stage, we implement a post-processing heuristic (for word ranking) that constructs longer key phrases starting with the selected keywords, by *collapsing* adjacent words from the top k ranking into phrases.[5] The problem with this collapsing strategy is that the final number of phrases cannot be predicted from the number of input keywords k, and there is no control over the number of collapsed phrases. Other variants of this collapsing strategy that alleviate this problem are possible, but they are found to introduce new complications, e.g., very long keywords or several keywords that are semantically redundant. We therefore use the basic collapsing heuristic described before.

[5]To better understand this heuristic, consider the following example: Assume the words "house" and "white" have been returned as top-ranked for the (tiny) document "POTUS spoke in the White House". In this case, the collapsing heuristic will yield "White House" as a keyphrase.

3.3 Features for Keyword Extraction

We extract two broad classes of features for keyword extraction from emails: word features and phrase features. These features are either used by themselves, in an unsupervised fashion, or together in a supervised setting.

In the following, we describe each feature, along with a short note on its potential utility in keyword extraction. Note that word and centrality features are extracted after removing stopwords.

3.3.1 Word Features

Word features are used to rank *word types* (i.e., unique words) based on their frequency, positional, and surface properties.

- **Tf**: Raw frequency of a word type in a document.[6] It is an important indicator of the word's saliency.
- **Tf.idf**: Raw frequency of a word type multiplied by its *idf* (inverse document frequency) computed on the British National Corpus [99].
- **First position**: Position of the first occurrence of a word in a document. Position is measured by number of word tokens since the beginning of the document. Words appearing towards the beginning of a document often contain introductory information, thereby being important from the perspective of keyword extraction. Position of a word is an important indicator of its "keyword"-ness, and has been exploited in several previous studies [6, 29, 30].
- **Last position**: Position of the last occurrence of a word in a document. Position, as before, is measured by number of word tokens since the beginning of the document. Words appearing near the end of a document may contain summary information, thereby becoming important.
- **Normalized first position**: First position feature, normalized by the number of words in the document. This is similar to the first position feature, except that it explicitly takes into account the length of the document so that longer documents may be penalized more.

[6] By "document", we mean an email or an email thread.

- **Normalized last position**: Last position feature, normalized by the number of words in the document. This is similar to the last position feature, except that it explicitly takes into account the length of the document so that longer documents may be penalized more.

- **Word length**: Number of characters in a word. Longer words sometimes contain richer information owing to word-compounding, morphology, etc.

- **Is capitalized?**: Whether the word is capitalized. This is a binary feature. Word capitalization is often a strong cue for detecting named entities.

- **Is in subject?**: Whether the word appears in the subject line of an email/thread. This is another binary feature. Words appearing in an email's subject line often contain important information, much like the words in the title line of a general document [19].

We also implement word centrality features, which are features defined on word co-occurrence networks [7]. For each email, a word co-occurrence network is constructed by adding all the word types (i.e., unique words) as nodes, and by drawing an edge between the words that occur next to each other.[7] Centrality measures on such co-occurrence networks can yield a powerful set of features for keyword extraction. In this work, we focus on the following centrality features:

- **Degree**: Number of edges incident to a node. Since word types implicitly endorse each other via collocation edges, the more edges that are incident to a word, the more important the word becomes.

- **PageRank**: Stationary probability of a random walk visiting a particular word in the word co-occurrence network. When used with teleportation, such a random walk ends up assigning higher probabilities to more important words in the network [7].

- **Coreness**: Measure of how "deep" a word is in the co-occurrence network. The "deeper" a word, the more its importance. This feature is inspired by the *core-periphery structure* of small-world networks, and computed using the so-called *k-cores decomposition* [14, 15].

[7] It is possible to draw the word co-occurrence network in a way that takes into account a window of words rather than successive words; we, however, have noticed that such networks merely change the neighborhood information of a particular token. In other words, it makes the network denser [100]. Such additional density was not found to be helpful in the canonical graph-based Keyword Extraction study that we followed in this chapter [7].

- **Neighborhood size (order one)**: Number of immediate neighbors to a node. It is a version of node degree that disregards *self-loops*, which can arise in word co-occurrence networks due to constructions such as "again, again and again". The more neighbors a node has, the higher its importance.

3.3.2 Phrase Features

In addition to features reflecting the importance of individual words, we also calculate phrase features, which are used to rank/classify entire *phrases*. More precisely, these features are used to classify *(document, phrase)* pairs, as explained in the next section.

Following [18], we extract four types of candidate keywords: stopword-filtered n-grams (n = 1, 2, 3, 4), stopword-filtered base noun phrases, named entities extracted using the Stanford Named Entity Recognizer (NER) [101], and named entities extracted using an unsupervised heuristic (sequences of capitalized words that never appear without capitalization). We use the CRFTagger [102] for part-of-speech tagging, and Mark Greenwood's NP chunker for base noun phrase identification.[8] The following features are used to rank these candidate keywords:

- **Phrase Tf**: Raw frequency of a phrase in a document.
- **Phrase Idf**: Inverse document frequency of a phrase, computed as the average of *idf*s of its constituent words. The word *idf*s were computed on the British National Corpus.
- **Phrase Tf.idf**: Raw frequency of a phrase multiplied by its *idf*.
- **Within-document frequency**: Number of sentences a phrase appeared in (for a particular document). The more sentences a phrase appears in, the higher its importance.
- **Mean length of containing sentences**: Mean length of the sentences a phrase appeared in (for a particular document) – in word tokens, word types, and (non-space) characters. These three features encode the importance of the containing sentences. Longer sentences should carry more information.
- **Phrase length**: Length of a phrase calculated as the number of constituent word tokens and non-space characters. These two features encode the fact that longer phrases usually carry more information.

[8] Available from http://www.dcs.shef.ac.uk/~mark/nlp/software/gate-plugins/chunkerv11.zip.

- **Length of the containing document**: Length of the document a phrase appears in – in word tokens, word types, and non-space characters. These three features encode the weight of the containing document.
- **Mean length of constituent words**: Average length of constituent words in non-space characters.
- **First and last containing sentences**: Index of the first and the last sentence a phrase appears in (for a particular document). These two features encode positional information of a phrase.
- **Diameter**: Difference between the indexes of the first and last containing sentences.
- **Wikipedia keyphraseness**: Ratio of the number of Wikipedia documents where a phrase appeared as a keyword, and the number of Wikipedia documents where the phrase appeared [22]. This ratio, when computed for phrases with reasonable document counts, provides an estimate of their importance as well as cohesiveness.
- **POS pattern probability**: Probability of a part-of-speech pattern emerging as a candidate keyword from among all base noun phrase patterns. This feature is inspired by a similar feature used in [103]. We included a second probability – probability of obtaining a candidate keyword pattern from among unique base noun phrase patterns. These two probabilities incorporate syntax information in our model.
- **Is in subject?**: Whether the phrase appears in the subject line of an email/thread. This is a binary feature.
- **Overlap with subject**: If the phrase appears in the subject line, then this feature is the length of the phrase in words, divided by the length of the subject line in words; otherwise, zero.
- **Are all words capitalized?**: This binary feature is a strong cue for detecting named entities.
- **Mean degree, PageRank, coreness, and neighborhood size**: Mean degree, PageRank, coreness, and neighborhood size (order one) of the constituent words of a phrase in the word co-occurrence network. Note that stopwords are not included in the word network. These four features indicate the importance of a phrase in terms of centrality. Higher values denote greater importance.

Table 3.1: Keyphrase ranking obtained with three features.

Mean neighborhood size		Mean coreness		tfidf	
afternoon	4	afternoon	6	enron	17.63
pen and pencil	3	phone	6	hector	10.62
tomatoes	2	week	6	chris	4.54
goody package	2	enron	6	tomatoes	4.12
hope	2	desk	6	goody package	3.95
york customers	2	rest	6	pen and pencil	3.48
rest	2	hector	4	golf shirt	2.52
week	2	chris	0	afternoon	2.41
golf shirt	2	golf shirt	0	desk	2.00
hector	2	new york	0	phone	1.59
desk	2	goody package	0	york customers	1.57
new york	2	tomatoes	0	new york	0.79
enron	2	pen and pencil	0	care	0.67
care	1	york customers	0	hope	0.55
phone	1	care	0	rest	0.49
chris	1	hope	0	week	0.41

- **Phrase degree, PageRank, coreness, and neighborhood size**: Degree, PageRank, coreness, and neighborhood size (order one) of a phrase in the *phrase co-occurrence network*. Phrase co-occurrence networks are similar to word co-occurrence networks, except that nodes are candidate phrases instead of words, and edges are defined between candidate phrases that appear in the same sentence. Higher values indicate greater importance.

Note that among the above features, coreness and neighborhood size are novel features in our study, and to the best of our knowledge, they have never been used in keyword extraction. Further, their behavior is different from tfidf. We illustrate an example in Table 3.1 (ECS080; corporate single email), which shows that both the ranking as well as the value ranges are different for phrase tfidf, phrase mean coreness, and phrase mean neighborhood size. For example, the word "afternoon" has a mean neighborhood size of 4 (i.e., it is connected to 4 other phrases) and a mean coreness of 6, but a low tfidf of 2.41 ("afternoon" appears in several of the emails). Yet another example is the word "chris." It has a low mean neighborhood size of 1 and low mean coreness of 0, but the tfidf is much higher (4.54), showing once again that these features are not redundant in their behavior.

Table 3.2: Example of a corporate email and a personal email, along with keyword annotations.

Corporate Email	**Personal Email**
I am faxing you both the Master Firm Purchase/Sale Agreement executed between Enron Gas Marketing (merged now into ECT) and Aquila Energy Marketing Corporation (merged now into Utilicorp.) with respect to Utilicorp. "signing" the agreement in lieu of giving a guaranty. Actually what Utilicorp. did in this agreement is sign as a co-obligor under the agreement (see Section 16.12 of the agreement). They signed accepting joint and several liability with respect to the obligations. If we can get them to agree to the same language in your master agreement that would effectively be as good or better than getting a guaranty. Anything less (like just sticking their "name" on the signature line) may not get us much or be worthless. All this, of course, is subject to any differences between US and UK law or issues under UK law which I will leave in Edmund's capable hands. Let me know if I can be of further service...	Hey, Does your email still work. I was wondering if you had a private email that would be appropriate for non business related correspondence?? Just curious. Still would like to find a way to keep in touch better, the messenger thing is ok, but its hard at times with work and all. At least with email we can say alittle more and at least have some uninterrupted time to "talk". Speaking of talking?.is there any time that is good to call? I would like to hear your voice once in a while! J
Keywords assigned by Annotator 1: Master Firm Purchase/Sale Agreement, Utilicorp, guaranty, obligations, agree to the same language, signing, liability, worthless	Keywords assigned by Annotator 1: email, private email, keep in touch better, hear your voice, talk, non business related correspondence, messenger thing, good time to talk
Keywords assigned by Annotator 2: Master Firm Purchase/Sale Agreement, Enron Gas Marketing, Utilicorp., sign as a co-obligor, language, US and UK law	Keywords assigned by Annotator 2: private email, non business related correspondence, uninterrupted time, keep in touch, 'talk'
Keywords assigned by Annotator 3: Master Firm Purchase/Sale Agreement, co-obligor, Utilicorp, Enron Gas Marketing, Aquila Energy Marketing Corporation, guaranty	Keywords assigned by Annotator 3: private email, non business related correspondence, messenger thing, uninterrupted time, voice
Keywords assigned by Annotator 4: agreement, Utilicorp, guaranty, Master Firm Purchase/Sale Agreement, signed	Keywords assigned by Annotator 4: keep in touch better, private email, call, messenger thing, hear your voice

Table 3.3: Keyword annotation statistics.

Email Category	Mean #Keyphrases	Standard Deviation
Corporate Single	6.68	1.88
Corporate Thread	7.72	2.36
Personal Single	6.79	2.11
Personal Thread	7.36	2.51
All Single	6.70	1.97
All Thread	7.53	2.47
All Corporate	7.06	2.12
All Personal	6.96	2.26

Table 3.4: Inter-annotator Agreement.

"Ground Truth" Annotator	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Exact match				
Annotator 1	27.00	40.96	32.55	23.23
Annotator 2	24.06	55.84	33.63	23.03
Annotator 3	21.16	58.27	31.04	20.96
Annotator 4	20.45	57.95	30.23	20.41
BOW match				
Annotator 1	46.37	58.25	51.63	47.17
Annotator 2	34.40	72.97	46.76	40.29
Annotator 3	33.35	74.33	46.04	40.26
Annotator 4	29.66	76.10	42.68	38.78
Relaxed match				
Annotator 1	40.59	61.59	48.93	44.43
Annotator 2	34.62	80.35	48.39	42.69
Annotator 3	30.71	84.56	45.05	39.14
Annotator 4	29.87	84.64	44.15	39.99

Table 3.5: Pairwise Inter-annotator Agreement.

Annotator-pair	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Exact match				
1 – 2	32.19	24.56	27.86	20.04
1 – 3	35.01	23.27	27.96	19.22
1 – 4	34.61	22.42	27.21	19.58
2 – 3	37.42	32.60	34.85	25.42
2 – 4	36.90	31.32	33.88	24.98
3 – 4	30.58	29.80	30.18	21.01
BOW match				
1 – 2	53.50	37.44	44.05	38.86
1 – 3	56.54	38.07	45.50	40.17
1 – 4	56.66	33.81	42.35	39.68
2 – 3	48.28	46.45	47.34	40.53
2 – 4	51.62	44.00	47.50	43.61
3 – 4	46.90	41.56	44.07	39.43
Relaxed match				
1 – 2	49.32	37.63	42.69	39.50
1 – 3	52.58	34.95	41.99	38.15
1 – 4	51.00	33.03	40.09	37.38
2 – 3	51.35	44.74	47.82	45.94
2 – 4	51.34	43.58	47.14	45.15
3 – 4	45.11	43.96	44.53	40.83

3.4 Evaluation

3.4.1 Dataset

To our knowledge, there is no dataset available for keyword extraction from emails. To evaluate our methods, we compiled our own dataset consisting of 212 single emails and 107 email threads (of 4-8 emails each) drawn from the Enron collection [98].[9]

First, each email and thread within this dataset was manually classified as either "private" or "corporate". The corporate emails discuss issues related to work and office, whereas the personal emails deal with issues related to home, family, and friends. Examples of corporate and personal emails are shown in Table 3.2.

Next, all the emails and threads in the dataset are annotated for keywords by four independent human judges. The annotators were asked to assign 5-20 keywords to each email/thread, ranked in their order of importance. We requested annotators to select keywords that are up to five words in length. While the definition of a "keyword" can vary depending on the annotator, we provided some guidelines and recommendations for increased consistency, e.g., we recommended the selection of noun phrases, named entities, or any other phrases that best capture the essence of a given email/thread.

Example keywords assigned by the annotators are shown in Table 3.2. Table 3.3 shows the keyword statistics of different categories of emails. Overall, threads have more keywords than single emails, and corporate emails have more keywords than personal emails.

We compute inter-annotator agreement by considering one annotator as the ground truth, and the (set union of) remaining annotators as the "system". Further, we consider three forms of agreement:

- **Exact match**: when two phrases match exactly (up to lowercasing and spaces).
- **BOW match**: when two phrases' bags of words (BOW) match exactly after lowercasing (except stopwords).
- **Relaxed match**: when two phrases either match exactly up to lowercasing, or can be made identical by adding a single word to the beginning or end of the shorter phrase [17].

Micro-averaged precision, recall, F-score, and Jaccard similarity under these three settings are shown in Table 3.4. Note that the best agreement under exact match is only

[9] An earlier version of this dataset has been described in detail in [104].

33.63% F-score, which is not very high, thus indicating the difficulty of the keyword extraction task (cf. [105]).[10] However, if we consider the BOW match, the best agreement is much higher (51.63% F-score). The same holds true for the relaxed match. This shows that annotators – although clearly divergent in their opinion, do in fact tend to select very similar *words* to construct their keyphrases. Results on *pairwise agreement* (Table 3.5) present the same evidence.

3.4.2 Evaluation Settings and Metrics

Our experimental results are primarily based on a *combined gold standard*, obtained from the set union of the keywords assigned by the four annotators, with an average of 19.35 keywords per email. Note that we also considered the alternative of creating a gold standard by using the set intersection of the four annotations. This results in zero keywords per email, reflecting the diversity of opinions on the annotations for this task. Instead, we adopt as our intersection gold standard the *union of pairwise intersections* between the annotations, yielding 5.70 keywords per email (on average). This gold standard results in an artificially small dataset that does not accurately reflect the performance of a keyword extraction system. Nonetheless, for the sake of completeness, we also report the results obtained on this intersection set (Section 3.5.3).

All the keyword extraction experiments are evaluated using micro-averaged precision, recall, F-score, and Jaccard similarity. We used F-score as our primary yardstick for comparing different systems. The evaluations are performed at *phrase-level*, where we count a candidate phrase appearing in the gold standard as an exact match (up to lowercasing and spaces).

3.5 Results and Discussion

We perform two sets of experiments, one consisting of unsupervised methods that rely on the individual use of the features described in Section 3.3, and a second set consisting of a supervised framework that combines all the features using machine learning.

[10] Having said that, 33.63% F-score is close to what one can reasonably expect as an upper bound on the inter-annotator agreement. For example, in the SemEval 2010 Keyphrase Extraction Task, the F-score achieved by readers on author-assigned keyphrases was 33.6% (cf. [10], Section 4).

3.5.1 Unsupervised Methods

For unsupervised keyword extraction, we first apply the pre-processing heuristic to select only nouns and adjectives, then rank candidate words according to different features (one feature at a time), followed by a selection of the *top 50%* of the words from the resulting ranked list, and finally use the post-processing heuristic to collapse adjacent words into key phrases. For the binary features (such as *Is in subject?* and *Is the word / Are the words capitalized?*), we take all words/phrases with value 1 instead of *top 50%*.

Table 3.6 shows the performance values obtained for the word and phrase features, and Table 3.7 shows the values for binary features. Note that *mean neighborhood size* yields the best F-score, followed by *phrase tfidf* and *phase tf*. Among word features, *word tfidf* performs the best, followed by *word PageRank*. The superiority of *tfidf* in both cases is in line with the findings by Hasan and Ng [9]. For binary features (Table 3.6), we see that one of them gives the highest precision among all systems (47.11%). However, their recall is very low (2-16%), thereby yielding a relatively low F-score (esp. for *Is in subject?*).

3.5.2 Supervised Methods

For supervised keyword extraction, we apply the same steps as in the unsupervised methods, but perform the ranking of the candidates using a machine learning algorithm applied in leave-one-out cross-validation fashion using all the phrase features. The supervised system includes a few features that cannot be used for keyphrase ranking, but could be potentially useful for the selection of keywords (e.g., document-specific features such as *document length*).

The supervised framework is formulated as a binary classification task, where each *(document, candidate keyword)* pair is classified as relevant or not. Using a small development dataset of 30 emails, we tried nine different classification algorithms, including KNN, Naive Bayes, SVM SMO, J48 decision tree, PART rule learner, OneR, Logistic Regression, AdaBoost and LogicBoost. We found that Naive Bayes and KNN performed best, and therefore used these classifiers in our experiments on the entire evaluation dataset. For all the classification experiments, we use Weka [106]. The performance values are micro-averaged.

The results of the leave-one-out cross-validation on the entire dataset of 319 emails are shown in Table 3.8. Interestingly, the results are comparable to the inter-annotator agreement rates reported in Tables 3.4 and 3.5, which is an indication of how accurate our best systems are as compared to human performance.

For an additional analysis, we also determine and report the most discriminative fea-

Table 3.6: Performance of unsupervised keyword extraction. Best values in different columns are boldfaced.

Feature	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Word features				
Tf	20.33	28.96	23.89	13.57
Tf.idf	**23.44**	**31.22**	**26.77**	**15.45**
First position	14.47	16.99	15.63	8.48
Last position	19.39	24.27	21.56	12.08
Word length	17.97	23.64	20.42	11.37
Degree	20.25	28.88	23.81	13.51
PageRank	21.41	28.37	24.41	13.90
Coreness	15.13	18.15	16.50	8.99
Neighborhood size	20.79	29.60	24.42	13.91
Phrase features				
Tf	25.65	33.16	28.92	16.91
Idf	25.63	33.14	28.91	16.90
Tf.idf	26.91	34.79	30.35	17.89
Wikipedia keyphraseness	22.68	29.32	25.58	14.66
Phrase length (words)	22.67	29.30	25.56	14.65
Phrase length (non-space chars)	25.02	32.34	28.21	16.42
Overlap with subject	21.91	28.32	24.71	14.09
Mean length of constituent words	25.27	32.67	28.50	16.62
Mean length of containing sentences in words	19.76	25.55	22.28	12.54
Mean length of containing sentences (unique words)	19.91	25.74	22.45	12.65
Mean length of containing sentences (non-space chars)	20.01	25.87	22.57	12.72
First containing sentence	17.35	22.43	19.56	10.84
Last containing sentence	20.56	26.58	23.18	13.11
Diameter	24.09	31.15	27.17	15.72
Within-document frequency	24.62	31.84	27.77	16.12
Mean degree	22.05	28.50	24.86	14.20
Mean PageRank	21.73	28.10	24.51	13.96
Mean coreness	20.58	26.61	23.21	13.13
Mean neighborhood size	**27.33**	**35.33**	**30.82**	**18.22**
Phrase degree	22.94	29.66	25.87	14.86
Phrase PageRank	23.71	30.66	26.74	15.44
Phrase coreness	21.09	27.26	23.78	13.49
Phrase neighborhood size	22.72	29.37	25.62	14.69

Table 3.7: Performance of binary features in unsupervised keyword extraction. Best values in different columns are boldfaced.

Feature	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Word features (binary)				
Is capitalized?	25.69	**16.97**	**20.44**	**11.38**
Is in subject?	**47.11**	2.66	5.04	2.59
Phrase features (binary)				
Are all words capitalized?	**33.29**	**11.34**	**16.91**	**9.24**
Is in subject?	30.96	2.70	4.96	2.54

Table 3.8: Performance of supervised keyword extraction. Best values in different columns are boldfaced. Performance values are micro-averaged in leave-one-out cross-validation.

Classifier	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
KNN	31.94	**50.03**	**38.99**	**24.22**
Naive Bayes	**45.40**	28.87	35.30	21.43

tures (by Information Gain), as shown in Table 3.9. Note that Phrase Tf.idf and Mean neighborhood size appear among the most discriminative features, which is not surprising since these two features are also among the best in the unsupervised approach (Table 3.6). Note further that Phrase Tf appears to be more discriminative than Phrase Tf.idf, and that the index of the first containing sentence and length of the containing document are also among most discriminative features.

3.5.3 Additional Evaluations

To further analyze the results of our supervised methods, we perform three additional evaluations.

Table 3.9: Most discriminative keyword extraction features by Information Gain on the email dataset.

Feature	Information Gain
Phrase Tf	0.04279
Phrase Tf.idf	0.03804
First containing sentence	0.03545
Length of containing document in word types	0.03287
Length of containing document in non-space characters	0.03284
Length of containing document in word tokens	0.03166
Mean neighborhood size	0.02775
Within-document frequency	0.02655

Table 3.10: Results of **in-domain training**. Best values in different columns are boldfaced. Performance values are micro-averaged in leave-one-out cross-validation.

Classifier	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
KNN	32.45	**51.64**	**39.85**	**24.89**
Naive Bayes	**45.69**	31.27	37.13	22.80

Table 3.11: Performance of supervised keyword extraction under **intersection gold standard**. Best values in different columns are boldfaced. Performance values are micro-averaged in leave-one-out cross-validation.

Classifier	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
KNN	22.04	**35.87**	27.30	15.81
Naive Bayes	**28.56**	32.52	**30.41**	**17.93**

First, we evaluate the effect of *in-domain training*, where for each of the four categories of emails in our dataset – personal single, personal thread, corporate single, and corporate thread – we restrict the training set to other documents in the same category. Table 3.10 shows the overall results obtained in this evaluation. Although the in-domain constraint results in a net decrease of the training set size, performance values improved because emails are more similar within a category than across categories. The F-score improvement with respect to the open-domain results from Table 3.8 are relatively small: 0.67 percentage point for KNN and 1.37 percentage point for Naive Bayes. Acknowledging that the size of the data used to train these in-domain systems is smaller than that used to train the open-domain data, the lesson learned from this experiment is that if domain-specific data is available, the same performance can be obtained with a fraction of the data.

Second, we evaluate our supervised methods against a gold standard dataset formed by using the intersection of the *pairwise annotations* produced by the human judges. As noted in Section 3.4, taking the intersection results in a very small dataset, which is not ideal for measuring the performance of an automatic system. We nonetheless report these results in Table 3.11, to show the ability of our system to identify these keywords that were agreed upon by both annotators.

Finally, to understand the performance of our keyword extraction methods on different types of emails (e.g., single emails versus threads; personal emails versus corporate emails), we perform separate evaluations of our supervised methods on each of the four different subsets of our dataset. Table 3.12 shows these comparative results. As seen in the table, personal emails are significantly more difficult to process than corporate emails. The highest F-scores are obtained with the KNN classifier on single emails, which may be

Table 3.12: Performance of supervised keyword extraction on subsets of our dataset: single emails; threads; personal emails; corporate emails.

Classifier	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Single emails				
KNN	36.02	53.17	42.94	27.34
Naive Bayes	51.48	24.03	32.76	19.59
Threads				
KNN	26.78	45.46	33.71	20.27
Naive Bayes	40.72	35.92	38.17	23.59
Personal emails				
KNN	30.64	46.48	36.93	22.65
Naive Bayes	48.19	27.05	34.65	20.96
Corporate emails				
KNN	32.89	52.77	40.52	25.41
Naive Bayes	43.66	30.27	35.76	21.77

due to the fact that there is less variance in the topics covered by the emails in this data (as opposed to threads, where there may be topic shifts).

3.5.4 Comparison with Existing Systems

To place our results in perspective, using our email dataset we evaluate five previously introduced systems for keyword extraction. We chose two state-of-the-art unsupervised keyword extraction systems – **SingleRank** and **ExpandRank** [12, 9], two top-performing systems in SEMEVAL 2010 keyphrase extraction task [10] – **KX_FBK** [107] and **SZTERGAK** [108, 109], and **KEA** [110] – a well-known supervised keyword extractor.[11] Table 3.13 shows the results obtained by these five systems, in comparison with our two best unsupervised methods, and our two supervised settings. Our Naive Bayes system gives the best precision, which is very encouraging in a subjective task like keyword extraction. Overall, our systems are found to be better than the state-of-the-art, with our KNN system leading to the best F-score (38.99%), which is 12.76% better than the best state-of-the-art system (SZTERGAK) on this dataset. This improvement is significant ($p < 0.00001$) using a two-sample test for equality of proportions with continuity correction. Also, our unsupervised systems performed better than state-of-the-art systems, with best F-score of 30.82%.

[11] We used Kazi Saidul Hasan's C++ implementation of SingleRank and ExpandRank, the publicly available TextPro implementation of KX_FBK (http://textpro.fbk.eu/), and the gitHub (Java) implementation of SZTERGAK (https://github.com/begab/kpe). KEA source code is available from https://code.google.com/archive/p/kea-algorithm/downloads.

Table 3.13: Comparison with existing systems. Best values in different columns are boldfaced. Performance values are micro-averaged. Systems marked with ^{O} are ours, ^{U} are unsupervised, and ^{S} are supervised. KX_FBK and SZTERGAK are two of the top performers in SEMEVAL 2010 keyphrase extraction task.

System	Precision (%)	Recall (%)	F-score (%)	Jaccard (%)
Phrase Tf.idf^{UO}	26.91	34.79	30.35	17.89
Mean neighborhood size^{UO}	27.33	35.33	30.82	18.22
KNN^{SO}	31.94	**50.03**	**38.99**	**24.22**
Naive Bayes^{SO}	**45.40**	28.87	35.30	21.43
SingleRank^{U}	36.77	19.13	25.16	14.39
ExpandRank^{U}	36.61	19.05	25.06	14.32
KX_FBK^{U}	24.47	25.35	24.90	14.22
SZTERGAK^{S}	41.03	19.27	26.23	15.09
KEA^{S}	26.52	6.91	10.96	5.80

Table 3.14: Keyphrase appropriateness in a post-hoc evaluation.

Email Category	Returned Keyphrases	Appropriate Keyphrases	Percentage
Corporate Single	141	111	78.72
Corporate Thread	73	63	86.30
Personal Single	109	94	86.24
Personal Thread	60	51	85.00

3.5.5 Post-hoc Evaluation of Keyphrases

As a final evaluation, we set up two experiments that allow us to measure the quality of the keywords extracted by our system in an extrinsic way.

First, we perform a post-hoc evaluation, where the keywords produced by our system are manually annotated by a human judge for appropriateness. We set this evaluation as follows: for a given email (single or thread), first the human judge carefully reads the email text to make sure she is familiar with its content; next, the judge is presented with a set of keywords, and her task is to determine which of the keywords reflect important content of the email text.

We take a random sample of 40 single emails (20 personal, 20 corporate), and 20 thread emails (10 personal, 10 corporate), and generate keyphrases using our best system (KNN). The human judge then classifies these keyphrases as appropriate or not, as described above.

Table 3.15: Email classification results. Standard deviations in parentheses.

	Text only	Keyphrase only
Accuracy (%)	90.0 (11.83)	85.0 (10.25)
Time (Seconds)	10.07 (1.10)	6.53 (1.13)

Table 3.14 shows the fraction of keywords found to be correct for each email type. The results suggest that in such a post-hoc evaluation, a significantly larger fraction (78-86%) of the keywords produced by our system are found to be acceptable by a human judge. This is in line with previous work on keyword extraction [103], which showed that there can be large gaps between the ad-hoc and post-hoc evaluations of keywords, as humans often have a difficult time generating a comprehensive list of keywords for a given text, yet they do agree with the appropriateness of a larger set of keywords when presented to them.

The second experiment consists of an application-based evaluation, where we simulate a potential classification task that a user has to accomplish when presented with a set of emails (e.g., the daily incoming email). Specifically, a human judge is given the task to classify each email in a set as being either "personal" or "corporate." We compare the scenario where the classification is performed by only reading the extracted keywords, versus reading the entire text, and measure both the correctness of the classification (against our own existing gold standard annotations) as well as the time it takes to perform the task in each scenario.[12]

We perform 10 rounds of simulation, where in each round we select 10 random emails and their corresponding keyphrases. The presentation order of the email texts or email keyphrases is randomized to remove any sequence effect. The classification accuracy of this simulation process – averaged over the 10 rounds – is shown in Table 3.15 (standard deviations in parentheses). Note that just by reading the keyphrases, the human judge was able to correctly classify the emails 85% of the time, whereas reading the full text leads to 90% – which is only 5% improvement. Note further that the time taken to classify the emails just by reading the keyphrases is 6.53 seconds on average, whereas reading the full text takes at least 10 seconds. This shows that keyword extraction can be very helpful in real life by substantially reducing the time taken to *triage* emails, at comparable accuracy levels.

3.6 Conclusion

Keyword extraction from emails is largely an open problem, with potentially important benefits given the growing number of emails that we have to handle in our daily communication. In this chapter, we described and evaluated methods for unsupervised and

[12]While we acknowledge that in a real-life setting, the name of the sender is often sufficient to classify an email as either personal or corporate, we use this task as an approximation for a generic email classification task. We believe this approximation is reasonable, given the fact that the human judge performing the task is (1) not provided with the sender name; and (2) is agnostic to the personal and corporate relationships of the actual email owner.

supervised keyword extraction from emails. We defined two types of features – word features and phrase features – which we then evaluated on a novel dataset consisting of emails manually annotated with keywords. To the best of our knowledge, our work is the first attempt to extract keywords from emails after the seminal study by Turney [41]. Our unsupervised experiments highlighted the role played by the different features for keyword extraction from emails. We also combined all the features using a supervised framework, and obtained results that improved significantly over the use of individual features. The results obtained with our best system represent a significant improvement over state-of-the-art in general-purpose keyword extraction, which is an encouraging result given the informal nature of emails and their difference from academic abstracts. Moreover, two extrinsic evaluations have further demonstrated the quality of the keywords extracted with our system.

The manually annotated email dataset introduced in this chapter is publicly available from http://lit.eecs.umich.edu.

CHAPTER 4

Specialized Keyword Extraction to identify product usage in consumer reviews: Data-driven usage extraction

In this chapter we introduce the problem of identifying usage expression sentences in a consumer product review. We create a human-annotated gold standard dataset of 565 reviews spanning five distinct product categories. Our dataset consists of more than $3,000$ annotated sentences. We further introduce a classification system to label sentences according to whether or not they describe some "usage." The system combines lexical, syntactic, and semantic features in a product-agnostic fashion to yield good classification performance. We show the effectiveness of our approach using importance ranking of features, error analysis, and cross-product classification experiments.

4.1 Introduction

Identification of *usage expressions* — phrases or sentence snippets describing product use in reviews — is an important problem in mining consumer product reviews. Identifying such usage expressions accurately allows us to view the relationship between consumers and products more clearly (e.g., by indicating how frequently a consumer uses a product). Further, the language and style employed in describing product use bring relevant and unseen aspects of the products to the fore (e.g., describing usage of a product in non-traditional and unique ways).

Usage expressions can take several forms, such as which aspects of the product are used, why the product is used, where it is used, how it is used, when it is used, and so forth (c.f. Section 4.2 for specific examples). The product could be used by a consumer in a number of ways, sometimes in unique ways not intended for originally. Hence enumerating all possible uses of a product is computationally intractable. In this chapter, therefore, we

Table 4.1: Product categories in our dataset.

Product category	Product	# Reviews	# Sentences	Avg # Sentences per Review
Laundry product	Scent booster	125	695	5.56
Cooking agent	Olive oil	110	588	5.35
Cooking agent	Vinegar	110	623	5.66
Medicine	Aspirin	110	463	4.21
Household item	Toothpaste	110	651	5.92
Total	–	565	3020	5.34

focus on four specific cases of product usage: why the product is used, where it is used, how it is used, and if there are any non-standard or non-traditional use (cf. Section 4.2).

While the relationship between product usage and consumer behavior has mostly been discussed by marketing researchers and psychologists, the question of whether the phenomenon of *usage* has any detectable signature in terms of the *language* used by consumers has not been addressed thus far. In this chapter, we introduce the task of identifying usage expressions from consumer product reviews. In particular, we focus on classifying review sentences as to whether they contain a *usage expression* or not. We create our own human-annotated corpus of 565 reviews on five distinct product categories containing more than 3000 sentences. We introduce a system that classifies sentences according to whether they contain a usage expression or not with 87.2% accuracy. We also show that an appropriate combination of lexical, syntactic, and semantic features performs better than individual feature categories.

4.2 Building a Usage Expression Dataset

Product reviews often contain usage information. Specifically, in addition to opinions on product quality, reviewers often share how, where, or why they use the product. We therefore build our dataset of product usage expressions starting with a collection of product reviews.

We collect Amazon product reviews for five different product categories, as shown in Table 4.1. The particular product lines we use are: a *laundry product:* specifically, Downy Unstopables In Wash Fresh Scent Booster 13.2 Oz; two kinds of *cooking agents*, namely, *Olive oil:* Baja Precious Extra Virgin Olive Oil from Baja California (750ml Bottle) and *Vinegar:* Raw Organic Apple Cider Vinegar by Bragg (1 gallon); a *Medicine:* Kirkland Signature Low Dose Aspirin, 2 bottles – 365-Count Enteric Coated Tablets each; and a *household item*, namely *Toothpaste:* Colgate Optic White Toothpaste, 4 Ounce (Pack of 2). The reviews are split into sentences, with the total number of sentences and average

number of sentences per review as shown in Table 4.1. In all, there are 3020 sentences in 565 reviews, with an average of 5.34 sentences per review.

With the help of three linguistics undergraduate students, each sentence in the dataset was annotated as containing a usage expression or not. Initially, as an early trial, we asked the annotators to indicate if a sentence contained a usage expression. This approach led to low inter-annotator agreement, so we refined the annotation process to a two-step process as follows.

In the first step, we instructed the annotators to read each product review carefully, identify all usage expressions in the review (examples below), and write them in a given textbox, one usage expression per line. Annotators were requested to write the usage expressions in their own words. This component was employed to make sure annotators carefully read and understood the review.

The second step involved answering the following four questions on usage types:

(A) Does the sentence describe why the product was being used? (usage reason/purpose) E.g., *"I used unstopables to freshen my room."*

(B) Does the sentence describe where the product was used? E.g., *"I used unstopables with my cat litter."*

(C) Does the sentence describe how the product was used? E.g., *"I use three cups of Downy Unstopables in every wash."*

(D) Does the sentence describe any non-traditional or non-standard usage of the product? E.g., *"I always love to add some hot water to unstopables and make my own DIY air freshener !"*

If a sentence had a positive answer to one or more of these four questions, then it was labeled as containing a usage expression.[1]

Additionally, several specific instructions were added to deal with potentially difficult or complex cases, by asking annotators to (1) consider the context (one sentence before and after the target sentence) before deciding whether to mark a sentence or not. (2) determine if a sentence contains an opinion (*"Love it"*, *"Hate it"*, etc.) or a recommendation (*"I'd recommend this product to all aspiring gardeners"*), and if so, pairing it with an explicit usage expression in some form. (3) determine if a sentence talks about usage of another product that is not the primary focus of the review (i.e., a secondary product), then mark the sentence only if the primary product is being used in addition to the secondary product. (4)

[1]Note that in this chapter, we ignore the different ways of product usage (why, where, how, non-traditional), but we plan to utilize the detailed annotations in future work.

Table 4.2: An example review and its annotations.

Sample Review
I used this recently when I washed my blankets and towels, and I was definitely impressed. Just a small amount (half a capful) was necessary to give my blankets and towels an extra burst of freshness. The scent is a little bit floral and lasts for a few days. I put the Downy booster directly into the washer. (Instructions say NOT to put in your dispenser) And it does work fine with high efficiency washers. I do recommend this for times when you may want extra freshness for your clothes or towels.
Usage annotations (agreed by all)
I used this recently when I washed my blankets and towels, and I was definitely impressed. Just a small amount (half a capful) was necessary to give my blankets and towels an extra burst of freshness.
Non-usage annotations (agreed by all)
The scent is a little bit floral and lasts for a few days.
Mixed usage/non-usage annotations
I put the Downy booster directly into the washer. (Instructions say NOT to put in your dispenser) And it does work fine with high efficiency washers. I do recommend this for times when you may want extra freshness for your clothes or towels.

determine if the secondary product is used instead of the primary product: *"Unstopables were not good, so I used sheets instead."*, or if only the secondary product was used: *"I used sheets, they are better."* then do not label the sentence. (5) focus only on products, and ignore other (named) entities like persons, organizations, locations, and dates.

Table 4.2 shows an example product review, and sentences that were agreed upon by all annotators to contain, or not, a usage expression. We also show sentences on which there was no consensus. Note that such sentences have a fair amount of ambiguity. For example, the sentence *"I do recommend this for times when you may want extra freshness for your clothes or towels."* does not seem to contain an explicit usage expression, but it does indicate that the consumer used the product to obtain extra freshness for clothes or towels. Sentences like this demonstrate the difficulty of identifying usage expressions in product reviews.

Inter-annotator agreement values, shown in Table 4.3, indicate that the task is moderately difficult. We can see that different products have different difficulty levels, with Vinegar being the least difficult (highest A_3 agreement as well as highest κ), while for the other four products, κ was between 0.43 and 0.48. This is presumably owing to the fact that Vinegar is a cooking agent and used in many different ways, thus providing more opportunity to find a usage sentence (by several people) in a product review.

To construct a gold standard, we took the majority of the three votes assigned by the

Table 4.3: Majority label statistics, and three-way inter-annotator agreement. A_3 is the % of sentences where all three annotators agreed. κ is the Fleiss' kappa among three annotators [1].

Product type	Majority Yes	Majority No	Majority Not Sure	All Yes	All No	A_3	κ
Scent booster	201	494	0	80	385	66.91	0.46
Olive oil	91	493	4	40	395	73.98	0.43
Vinegar	190	430	3	139	369	81.54	0.71
Aspirin	94	366	3	47	282	71.06	0.48
Toothpaste	137	514	0	56	411	71.74	0.46
Overall	713	2297	10	362	1842	72.98	0.52

three annotators to each sentence. There were 36 sentences (1.19% of all sentences) that did not have a majority. One of the authors manually arbitrated these sentences into "usage" ($n = 22$) and "not usage" ($n = 14$) classes.

4.3 Finding Usage Expression Sentences

Once the annotated dataset was finalized, our primary goal was to build a classifier to predict if a given sentence contains usage expressions or not. We learn the classifier over five categories of features extracted from the sentence and neighboring context. In this chapter, we show the performance using a logistic regression classifier, chosen based on its performance on a small development dataset of usage-annotated sentences drawn from 20 product reviews. The following features are included:

(A) Lexical features: As n-grams are usually very helpful in document classification, we explore their utility on the task of usage expression sentence classification. We use word unigrams and bigrams, part-of-speech (POS) bigrams, and character trigrams. We use the CRFTagger [102] for POS tagging.

(B) Embeddings: Embeddings encode *latent semantics* and could reflect usage patterns. We train a word embedding using word2vec [111] over a large corpus of 55,463 product reviews. This corpus is constructed from all Amazon reviews associated with any product that has "Unstopables", "Olive oil", "Vinegar", "Aspirin", or "Toothpaste" in its title. Once the word embedding is trained, a sentence is represented by the weighted average of the embeddings of all the unique words in it. It is to be noted that such averaging is fairly common and gives good results [112, 113, 114].

(C) Syntax: We use bags of constituency and dependency production rules, obtained from the output of the Stanford parser [115, 116]. For constituency grammar, we use terminal and non-terminal rules separately as well as together. For the dependency grammar, we use the (collapsed) dependency types (*amod*, *nsubj*, etc.), and the lexicalized dependencies

(e.g., *(nsubj, Kirkland, seems)*) as separate features.

(D) Style: We extract thirteen shallow surface-level and style features to encode the stylistic properties of a sentence, in the hope that they would be predictive of whether the sentence contains a usage expression. These features are: sentence position, average word length (in chars), sentence length (in words and characters), type-token ratio, Flesch Reading Ease [117, 118], Automated Readability Index [119], Flesch-Kincaid Grade Level [120], Coleman-Liau Index [121], Gunning Fog Index [122], SMOG Score [123], Formality [124], and Lexical Density [125].

(E) Semantics: Since *usage* is above all a semantic phenomenon, a *semantic space* should be able to capture the dominant properties of the usage expression. We use the following feature sets to capture a semantic space for a sentence. Each feature set effectively describes a *lexicon*, and we turn "on" the features in the lexicon that are present in the target sentence.

1. **Product categories:** This feature set consists of the list of product categories obtained from the Walmart API.[2] We use both main categories and sub-categories.

2. **Concreteness:** The set of words, along with their concreteness scores, available as part of the Free Association Norms Database [126]. There are more than 3,000 words available as part of the database.

3. **Levin classes:** The set of coarse and fine-grained variations of Levin verb classes and verb alternations, leading to four types of features [127].

4. **LIWC:** Like Levin classes, we included another set of features derived from the LIWC dictionary of psychological word categories [16].

5. **Semantic lexicons:** Like Levin classes, we use the Roget thesaurus and WordNet Affect [128] word categories, with a binary feature representation. If a word falls under any of the Roget word categories, the corresponding feature is set.

6. **Named Entities:** We use the Stanford NER [101] to identify named entities in our corpus, and then use these entities as bag-of-features. We use the terms, the entity types, and the lexicalized entity types (terms + entities) as our bags. Standard tf, tfidf, and binary representations are used. We use the seven-class typology of named entities (Location, Person, Organization, Money, Percent, Date, Time).

7. **Spatial Prepositions:** Recent studies have shown prepositions to be a precious source of semantic information [129, 130, 131]. We use a lexicon of *spatial prepo-*

[2] https://developer.walmartlabs.com/

sitions[3] as a bag-of-words feature. The rationale was to observe if spatial properties of usage of objects ("use olive oil **with** celery", "put detergent **in** washer") can be captured in terms of prepositions such as *on*, *in*, *by*, *with*, etc.

8. **Semantic Distance:** Finally, we added the (weighted) WordNet distance[4] between all words and the verb *use*, where weights are set as binary, tf, and tfidf, as before. The rationale behind this feature is that it captures words similar to the verb *use* in the sentence, and their relative importance.

4.4 Evaluation

We use the dataset introduced in Section 4.2 to evaluate the accuracy of the usage detection classifier. 20% of the data for each product is held out as test data, and the remaining 80% is used for training.

We start by evaluating each individual feature using a ten-fold cross-validation on the training data. We then explore three combination methods, applied on a subset of seven feature sets, selected based on their performance and diversity: word unigrams, POS bigrams, character trigrams, embeddings, constituency rules, product categories, and concreteness. We combine these features through: classifier voting, where we assign the class predicted by the majority of the classifiers; feature fusion, where we join all the individual features into one feature vector used in the classification; and meta-learning, where we use the output of the individual classifiers as input into another classifier (again using logistic regression for the meta-learner). Table 4.4 shows the results of these evaluations. As seen in the table, while simple features, such as word n-grams and character trigrams, lead to the best performance among the individual features, better performance is obtained when they are combined with other features (bottom rows of Table 4.4). Table 4.4 further shows that *keyphrases* extracted using the system we designed in Chapter 3 – when used as features – give the best recall in this task (68.80%).

The meta-learner based combination strategy resulted in the best performing classifier during the cross-validation experiments on training data. We next evaluate this classifier on the test data consisting of 20% reviews of all five products. Table 4.5 shows the results obtained on the test data. For comparison, the table also shows the performance of the word unigram classifier, as well as a majority class baseline that labels every sentence as "non-

[3]Obtained by combining the two lists at https://owl.english.purdue.edu/owl/resource/594/04/ and http://www.firstschoolyears.com/literacy/sentence/grammar/prepositions/resources/Spatial%20Prepositions%20word%20bank.pdf.

[4]We use the Wu-Palmer similarity [132].

Table 4.4: Micro-averaged sentence-level results (%) under 10-fold cross-validation on the training data. Maximum value in each column (within each section) is boldfaced.

Feature Type	Prec.	Rec.	F-score	Accu.
Word unigrams	71.56	54.94	62.16	**83.88**
Word bigrams	**77.06**	30.85	44.06	81.13
Character trigrams	70.06	**57.19**	**62.98**	83.80
POS bigrams	55.72	39.69	46.36	77.87
Embeddings	71.92	47.49	57.20	82.88
Constituency	70.49	52.17	59.96	83.22
Dependency	57.53	33.10	42.02	78.00
Style	54.17	11.27	18.65	76.33
Product categories	67.19	44.37	53.44	81.38
Concreteness	59.61	53.21	56.23	80.04
Levin classes	59.72	37.26	45.89	78.83
LIWC	57.14	38.13	45.74	78.20
Semantic lexicons	56.02	50.78	53.27	78.54
Spatial prepositions	41.67	3.47	6.40	75.57
Semantic distance	66.29	20.45	31.26	78.33
Keyphrases	45.37	**68.80**	54.68	72.53
Classifier voting	66.84	**67.76**	**67.30**	84.13
Feature fusion	63.92	60.49	62.15	82.25
Meta learner	**73.61**	59.45	65.77	**85.09**

Table 4.5: Micro-averaged sentence-level results (%) on the **test set** (20% of all products). Maximum value in each column is boldfaced.

Feature Type	Prec.	Rec.	F-score	Accu.
Majority	0.00	0.00	0.00	76.13
Word unigrams	71.82	58.09	64.23	85.92
Meta learner	**76.92**	**58.82**	**66.67**	**87.20**

usage." As before, the meta-learner significantly improves over the unigram classifier,[5] and also over the majority class baseline.[6]

We also report the performance of the meta-learner classifier on individual products in Table 4.6. Across all the products, vinegar appears to have the highest F-score. This can be partly explained by the high inter-annotator agreement: the same product had the highest three-way agreement in the manual annotations, as shown in Table 4.3, likely an indication of a less difficult dataset.

[5]Paired t-test, p-value=0.07

[6]Paired t-test, p-value < 0.0001

Table 4.6: Micro-averaged sentence-level results (%) *per product* using the meta learner.

Product	Prec.	Rec.	F-score	Accu.
Scent booster	78.57	68.75	73.33	87.69
Olive oil	50.00	25.00	33.33	89.26
Vinegar	81.58	79.49	80.52	88.37
Aspirin	70.00	36.84	48.28	84.54
Toothpaste	80.00	53.33	64.00	85.00

Table 4.7: Feature importance ranking for four feature types. We show ten top-ranked features along with their importance scores. For the meta-learner, we show the ranking over the subset of seven feature sets used in this classifier.

Word unigrams		Category words		Concreteness		Meta learner	
and	0.023	the	0.040	smell	0.025	Character trigrams	0.309
my	0.019	my	0.036	use	0.024	Word2vec	0.236
smell	0.014	smell	0.029	day	0.023	Word unigrams	0.171
day	0.014	a	0.028	for	0.019	Constituency	0.119
use	0.014	use	0.025	clothes	0.017	Concreteness	0.077
it	0.011	day	0.023	i	0.016	Category words	0.053
clothes	0.010	this	0.020	with	0.014	POS bigrams	0.035
a	0.010	clothes	0.018	drink	0.014		
bought	0.009	daily	0.015	water	0.013		
drink	0.009	drink	0.013	daily	0.013		

4.5 Additional Analyses

To gain further insights, we perform several additional analyses, to determine: the role played by different features; the relation between classifier performance and amount of training data; the role of in-domain vs. cross-domain classification; and finally the types of errors produced by the system.

4.5.1 Feature Importance Ranking

Table 4.7 shows the top features (ranked by their Gini importance [88]) for three prominent individual feature-based classifiers — viz. word unigrams, category words, and concreteness — and the meta-learner. Note that top-ranking words include product properties (*smell*), secondary objects on which the product was used (*clothes*), how the product was used (*day*, *daily*, *drink*, *water*), usage verbs (*use*), prepositions and conjunctions (*and*, *for*, *with*), pronouns (*i*, *it*, *this*), and articles (*a*, *the*). For the meta learner, lexical features (character trigrams and word unigrams) and embedding features (Word2vec) are among the top-ranked feature classes.

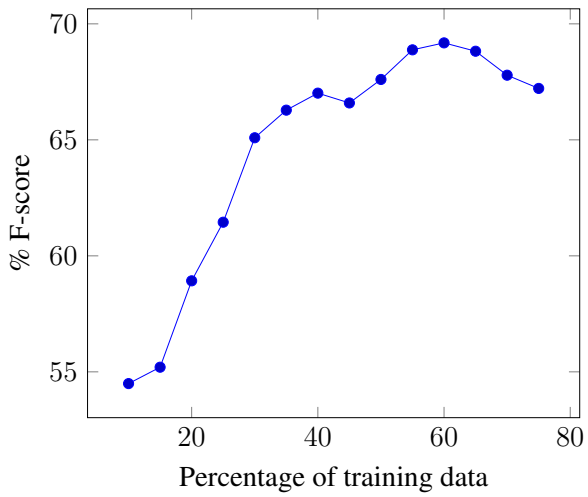

Figure 4.1: Learning curve using micro-averaged sentence-level results for the meta-learner classifier.

4.5.2 Learning Curve

Next, we experiment with varying the size of the training data to understand the learning curve. We gradually increased the amount of training data from 10% to 80%, in steps of 5%; and evaluated on the full test data. Figure 4.1 shows the variation of F-score achieved by the meta-learner as the training data is increased, smoothed over three consecutive data points. The test performance was the highest when trained on 60% of training data and then decreased gradually, which suggests that the system might not benefit from additional training data.

4.5.3 The Role of In-Domain Data

To understand the role played by in-domain data, we further experiment with two different configurations of training and test sets.

In one configuration, we train on four products, and test on the remaining product (*cross-domain training*). As can be seen from Table 4.8, this results in lower F-scores than Table 4.5. This suggests that identifying usage expressions of a product is intimately related to the identity of the product, echoing the findings by [50].

In the second configuration, we train on 80% of a product, and test on 20% of the same product (*in-domain training*). The results, averaged over the five products, are shown in Table 4.9. Note that the F-score values are much improved compared to the previous configuration, and are comparable to the results shown in Table 4.5. This suggests that when storage/memory might be a concern, we could simply use training data from *within the*

Table 4.8: Cross-domain classification: Micro-averaged sentence-level results (%), where test set is an individual product, and training set is four other products. Maximum value in each column is boldfaced.

Feature Type	Prec.	Rec.	F-score	Accu.
Baseline	0.00	0.00	0.00	76.39
Word unigrams	69.15	35.20	46.65	80.99
Meta-learner	70.62	38.43	**49.77**	**81.69**

Table 4.9: In-domain classification: Micro-averaged sentence-level results (%), where test set is 20% of an individual product, and training set is 80% of the same product. Maximum value in each column is boldfaced.

Feature Type	Prec.	Rec.	F-score	Accu.
Baseline	0.00	0.00	0.00	78.24
Word unigrams	74.19	50.74	60.26	85.44
Meta-learner	76.53	55.15	**64.10**	**86.56**

domain to achieve comparable performance. This strategy also results in a faster training time and a smaller model, similar to the findings in [133].

4.5.4 Error Analysis

Finally, we also conducted a manual inspection of two broad categories of errors – **false positives**, i.e. "not usage" sentences marked as "usage" ($n = 25$), and **false negatives**, i.e. "usage" sentences marked as "not usage" ($n = 56$). This analysis revealed the following sub-categories for the false positives:

- **Number expressions:** Seven instances (29.17%) of errors can be attributed to numeric expressions occurring within sentences ("two years", "3am", "third bottle", etc.).
- **Erroneous gold labels:** Six instances (25%) were actually correctly labeled as "usage" by the system, whereas the gold label was wrong (*"I really love the smell of fresh laundry, and the smell of Downy."*).
- **Shortcomings:** Six examples (25%) talk about actual or perceived shortcoming(s) of a product. *"Olive oil used for healthy properties doesn't keep well in plastic.[sic]"*
- **Others:** Five instances (20.83%) were not captured by the above categories: *"I used to drink a small shot each day, but haven't for a while."*

False negatives have the following sub-categories:

- **Positive adjectives and adverbs:** 21 instances (37.5%) can be attributed to positive adjectives ("good", "great", "excellent"), and/or positive adverbs ("really", "impressively", "well"). *"It smells amazing and lasts forever."*

- **Use-related verb in primary clause:** Eleven examples (19.64%) contain a use-related verb ("use", "help", "need") in the primary clause: *"I use this to eat, not to cook with."*

- **Erroneous gold labels:** Nine instances (16.07%) are actually correctly labeled as "not usage" by the system, but the gold label was wrong (*"When I have to hang dry clothes, they get this horrible egg water odor."*).

- **Non-traditional usage:** There are three instances (5.36%) that talk about non-traditional or innovative usage of a product: *"I have since made small sachet bags for my closets, car and as gifts."*

- **Others:** Twelve instances (21.43%) were not captured by the above categories: *"I actually saw results after the first use."*

4.6 Conclusion

In this chapter, we introduced the task of identifying usage expression sentences in consumer product reviews. A dataset comprising more than 3,000 annotated sentences was created from reviews of five products. We also trained a binary classifier to identify sentences that talk about the usage of a product. Extensive feature tuning and fusion experiments resulted in performance values comparable to the inter-annotator agreement. Detailed feature ranking, error analysis, and per-product performance numbers have been reported. Directions for future research include: experiments on a larger dataset of reviews with more diverse product types, expanding to other genres of reviews such as product blogs, and identifying types of usage expressions (how, where, why, and non-traditional uses). The work can also be extended to model the "personality" of a product with the "personality" of users – perhaps measured by the average personality of all people using the target product.

The annotated dataset is publicly available for research use from `http://lit.eecs.umich.edu/downloads.html`.

CHAPTER 5

Matching students to faculty: Keyword Extraction to find student interests and match them to faculty research topics

Every year, millions of students apply to universities for admission to graduate programs (Master's and Ph.D.). The applications are individually evaluated and forwarded to appropriate faculty members. Considering human subjectivity and processing latency, this is a highly tedious and time-consuming job that has to be performed every year. In this chapter, we propose several information retrieval models aimed at partially or fully automating the task. Applicants are represented by their statements of purpose (SOP), and faculty members are represented by the papers they authored. We extract keywords from papers and SOPs using a state-of-the-art keyword extractor. A detailed exploratory analysis of keywords yields several insights into the contents of SOPs and papers. We report results on several information retrieval models employing keywords and bag-of-words content modeling, with the former offering significantly better results. While we are able to correctly retrieve research areas for a given statement of purpose (F-score of 57.7% at rank 2 and 61.8% at rank 3), the task of matching applicants and faculty members is more difficult, and we are able to achieve an F-measure of 21% at rank 2 and 24% at rank 3, when making a selection among 73 faculty members.

5.1 Introduction

Every year, millions of students worldwide apply for graduate education in the United States. In Fall 2012 alone, US universities received 1.98 million graduate applications, and more than 461,000 students enrolled in graduate studies for the first time between Fall 2011 and Fall 2012.[1] With such a high number of students applying to US universities for

[1] https://www.cgsnet.org/us-graduate-schools-report-slight-growth-new-students-fall-2012

graduate studies, and that number increasing over the years,[2] the problem of processing this voluminous amount of applicant data into a more manageable and more automated pipeline assumes paramount importance.

Ph.D. applicants in particular pose a greater challenge because they need to be screened for funding offers and matched with potential advisors. While some applicants do specify the group or the professor with whom they would like to work with, many do not provide a selection. The problem is somewhat alleviated by having a separate survey in online application forms that allows applicants to mention which faculty members they would like to work with, and rank those faculty members in order of preference. Still, it largely remains the university's and ultimately the departments' responsibility to ensure Ph.D. applicants are matched with appropriate faculty members. Departments typically employ a faculty subgroup or separate staff members to read through graduate applications, forward them to appropriate faculty members, and create online "profiles" of applicants so that they could be matched more easily with faculty members. The problem, however, is that not all faculty members toward whom an applicant shows interest can offer financial support or have a matching interest in the applicant.

Our goal in this project is to *automate the process of matching applicants with faculty members*. In particular, we want to leverage the free text available as part of the applications to aid us in the decision process. To showcase our approach, we use the applicant data from the Computer Science and Engineering department at a large Midwestern university that had over 1,100 graduate applications in Fall 2014. Manual matching of Ph.D. applicants with appropriate faculty members was also available. We designed several information retrieval systems that would:

1. Match applicants and research areas:
    (a) given an applicant, retrieve the most likely research areas the applicant would match;
    (b) given a research area, retrieve from the pool of available applicants those most likely to be a good match;
2. Match applicants and faculty:
    (a) given an applicant, retrieve those faculty members with similar research interests;
    (b) given a faculty member, retrieve the most likely applicants to possess similar research interests;

[2] http://www.cgsnet.org/ckfinder/userfiles/files/R_IntlApps12_I.pdf

(c) given an applicant, retrieve the most likely research areas the applicant would match, and then from those, select faculty members with similar research interests.

The rest of the article is organized as follows. We outline related studies in Section 2.6, followed by a description of our dataset in Section 5.2. Section 5.3 presents exploratory analysis of the keywords extracted from faculty published work and applicants' statement of purpose, setting the stage for Section 5.4, where we describe information retrieval systems and the importance of keywords in constructing them. Section 5.5 concludes the chapter, outlining future research directions.

5.2 Data Description

Since our problem formulation involves the ranking of *faculty members* against *applicants* (and vice versa), we need a convenient textual representation for both. We opt to represent applicants by their *statements of purpose* (SOP), and faculty members by the papers they have (co-)authored in the prior 12 years (between 2004 and 2015). Anonymized statements of purpose are available for all applicants in the Fall 2014 cohort at the Computer Science and Engineering department at the university in question. Note that SOPs usually talk about what the applicant has achieved in the *past*, what (s)he is doing at *present*, what (s)he would like to do/be in the *future*, and how all these *connect* with the particular department and its faculty.

Papers were collected for 73 faculty members from their Google Scholar Citations[3] and DBLP[4] profiles. We collected 4,534 papers authored between 2004 and 2015, and converted their PDFs into text using UNIX *pdftotext* utility. Sometimes multiple faculty members collaborate on a single paper; we counted those papers multiple times, once for each participating faculty. Authorship statistics of the 5 most prolific authors are shown in Table 5.1 (faculty names have been anonymized to protect privacy). Note that a few of the most prolific authors wrote over 200 papers between 2004-2015, or almost 17 papers a year. This data follows a power-law distribution with exponent $\alpha = 3.45$ (statistically significant with p-value = 0.999).

We also obtained a pairing of Ph.D. applicants (Fall 2014 cohort) with faculty members, constructed manually by a small group of faculty. Note that each applicant is identified by a numeric ID and may be matched with multiple faculty members. On the other hand, a

[3]http://scholar.google.com/

[4]http://www.informatik.uni-trier.de/ ley/db/

Table 5.1: Number of papers (co-)written by several faculty members (anonymized) between 2004 and 2015.

Faculty Member	Number of Papers
Tommy M. Rosenbalm	331
Thomas M. Burns	300
Ali H. Salgado	212
Richard G. Meza	146
Nicole L. Thompson	140

Table 5.2: Number of applicants assigned to several faculty members (anonymized).

Faculty Member	Number of Applicants
Richard C. Hardy	45
George E. Ford	45
Robert S. Peters	42
Jeff L. Jurgens	41
Dennis R. Salisbury	38

Table 5.3: Research areas at the Computer Science and Engineering department at a large Midwestern university. Highest value in each column is boldfaced. Applicants are from Fall 2014 pool.

Research Area	Number of Faculty	Applicant to Faculty Ratio	% of Applicants in Area
Artificial Intelligence	**29**	7.03	**67.11**
Chip Design, Architecture, and Emerging Devices	22	3.41	24.67
Databases and Data Mining	6	**16.00**	31.58
Embedded and Mobile Systems	12	6.67	26.32
Human-Computer Interaction	8	6.63	17.43
Languages, Compilers, and Runtime Systems	13	3.54	15.13
Networking, Operating Systems, and Distributed Systems	16	4.56	24.01
Robotics in CSE	7	11.71	26.97
Secure, Trustworthy, and Reliable Systems	22	4.32	31.25
Theory of Computation	10	5.4	17.76
Warehouse-Scale and Parallel Systems	19	4.16	25.99

Table 5.4: Keyword statistics.

Keyword Type	SOPs Keyword Count	Papers Keyword Count
All keywords	53,166	123,171
Multi-word keywords	44,473	98,470
All keywords after filtering	13,472	27,563
Multi-word keywords after filtering	6,022	10,170

Table 5.5: Top multi-word keywords from SOPs, ranked by *tf.idf*.

machine learning	1166.78
computer vision	713.66
computer science and engineering	706.06
artificial intelligence	679.31
computer architecture	651.95
data mining	562.43
electrical engineering	516.14
natural language	493.75

faculty member is represented by his/her username, and may be matched with (or express interest in) several different applicants. There were 1107 applicants in total, of which 304 were matched with a faculty member. Different faculty members received a different number of applications. Faculty members receiving the highest number of applications in Fall 2014 cohort are shown in Table 5.2 (faculty names have been anonymized to protect privacy).

The faculty conducts research in 11 different areas, as shown in Table 5.3. The areas vary in terms of number of faculty, percentage of applicants, and applicant-to-faculty ratio. Artificial Intelligence (AI), for example, has the highest number of faculty members and the highest percentage of applicants. Databases and Data Mining, on the other hand, comprises the lowest number of faculty and the second highest percentage of applicants, which leads to the highest applicant-to-faculty ratio across all research areas. These observations could be helpful in identifying areas where additional faculty members need to be recruited.

5.3 Exploratory Analysis of Keywords

To represent the SOPs and papers by their *content* rather than *style*, we use an automatic system to extract keywords. We employ a state-of-the-art system previously used in the email domain [134]. Keyword statistics are provided in Table 5.4; note that we also include the counts for filtered keywords using Wikipedia article titles to obtain a more salient listing of keywords.

We first want to see *what students talk about most* in their SOPs in terms of key-

Table 5.6: Most important keywords from papers published in different years. Importance was measured by *tf.idf*. Top keywords that are unique to each year are shown in boldface.

2004	**2005**	**2006**	**2007**
test set	file system	file system	natural language
ubiquitous computing	sensor network	natural language	file system
power management	error rate	data set	data race
computer science	virtual machine	error rate	ad hoc
lower bound	power management	ad hoc	energy consumption
2008	**2009**	**2010**	**2011**
file system	**network virtualization**	power consumption	data race
control logic	control logic	file system	shared memory
power consumption	power consumption	**reward function**	**episodic memory**
energy consumption	data set	computer science	error rate
computer science	virtual machine	signal processing	**medical device**
2012	**2013**	**2014**	**2015**
energy consumption	electrical engineering	**social media**	**homomorphic encryption**
energy efficiency	**data center**	anomaly detection	data mining
nash equilibrium	computer science	power consumption	anomaly detection
computer science	natural language	data mining	**data science**
electrical engineering	energy efficiency	**computational linguistics**	**reinforcement learning**

words. Table 5.5 shows that the most salient keywords in SOPs are general and trendy terms such as "machine learning," "artificial intelligence," "data mining," and "computer vision." Other terms are even more generic, such as "computer science and engineering," and "electrical engineering.". These keywords indicate that students are indeed familiar with the trendy terms and buzzwords in Computer Science and Engineering, and most students want to go to those areas. In comparison, when we look at *what the faculty talk about most* in their papers (cf. Table 5.6), we observe highly technical terms and domain-specific keywords such as "nash equilibrium" and "episodic memory." This observation leads support to the fact that students are usually not sufficiently aware of the publication records of different faculty members (*information gap*), and students usually apply to "hot" areas rather than established areas (where there are more papers), perhaps because of increased media attention to those areas. This information gap further shows that our problem is complex, as we need to match texts from students and texts from faculty containing disparate sets of keywords.

An intriguing question at this point is to explore *how the keywords change over the years*. We analyzed the publications of all faculty members by year and ranked the keywords used by *tf.idf*. Table 5.6 shows that there is a distinct *trend* in the top-ranked keywords, in the sense that each year seems to focus on some particular problems (perhaps at the expense of others), and each year has some *new problems* that were not salient before. Year 2014, for example, introduces "social media" as a salient keyword, whereas year

Table 5.7: Ranking of faculty members (anonymized). Top faculty members that are unique to a particular ranking are shown in boldface.

Diversity	**Focus**	**Content Density**
Nicole L. Thompson	**Stephen M. Evans**	Jack J. Santoro
Francis G. Okelley	**Richard C. Hardy**	Rodolfo C. Hayes
John L. Wheatley	**Kevin D. Llanes**	Nicole L. Thompson
Tommy M. Rosenbalm	**George E. Ford**	James C. Rhinehart
Ali H. Salgado	Michael M. Lewis	Francis G. Okelley

2015 introduces "data science." It is important to note that graduate applicants are often not aware of such subtle variations and trends going on in the research community and thus cannot prepare accordingly.

We next explore *how the faculty members rank according to their diversity and focus* of research topics, as related to applicants. While diversity is usually defined as the opposite of similarity in Information Retrieval [135], we measured *diversity* in the context of keywords by Jaccard Similarity[5] between all keywords of a faculty and keywords from all applicants, whereas *focus* was measured by Jaccard Similarity between all keywords of a faculty and keywords from applicants assigned to him/her. Table 5.7 (faculty names have been anonymized to protect privacy) shows that these two rankings are substantially different. Furthermore, looking at *content density* (total number of keywords as a fraction of total number of words – averaged over papers), we see that the ranking changes again. It is important to note such subtle differences, because they help applicants make an informed decision.

Intriguingly, we find *focus* to be highly positively correlated with *popularity* (Spearman's $\rho = 0.8$), where the latter is measured by *how many students are assigned to a faculty* (cf. Table 5.2). *Diversity* and *popularity* are only moderately correlated (Spearman's $\rho = 0.28$), whereas the correlation between *diversity* and *focus* is even lower ($\rho = 0.12$). Very low correlation is observed between *content density* and *focus* ($\rho = 0.04$). Similarly low values are obtained for correlations between *content density* and *popularity*.

5.4 Information Retrieval Models

The objective of our study is to help academic departments *match applicants with faculty members*. We cast this problem as an *information retrieval*-like task, where given an applicant as query, our system retrieves research areas and faculty members. The system is also able to retrieve applicants with respect to faculty members as queries. We consider the

[5] https://en.wikipedia.org/wiki/Jaccard_index

following use cases:

1. match applicants and research areas

   (a) consider an applicant's statement of purpose as a query, while all publications in a given research area form a single document, and retrieve the most similar of these documents; retrieval is done among 11 research areas. We will call this variation *SOP as query, research areas as documents*.

   (b) consider all publications in a research area as a query for which we seek to retrieve the strongest matching statement of purpose pertaining to the applicants; retrieval is done among 304 applicants. We will refer to this variation as *research area as query, SOP as documents*.

2. match applicants and faculty

   (a) consider an applicant's statement of purpose as a query, while all publications pertaining to a given faculty as a single document; the retrieval is done for the most similar documents. This variation is represented as *applicant as query, faculty members as documents*; retrieval is done among 73 faculty members.

   (b) consider the cumulative publications of a faculty member as query, while each applicant is represented through his / her statement of purpose. This variation is referred to as *faculty as query, applicants as documents*. Retrieval is performed among 304 applicants.

   (c) consider an applicant's statement of purpose as a query. Retrieval of the most relevant faculty members is performed hierarchically, first with respect to the best matching research groups (represented through the totality of articles published by faculty in that group), and then with respect to the best matching faculty members from within the top groups. We will refer to this variation as *applicant as query, faculty members as documents – hierarchical*; retrieval is first performed against the 11 research areas, and then against the faculty members in the top research areas.

While applicant publications and/or data gathered from application forms could potentially be used to match applicants with faculty, we considered such an approach to be problematic because of the difficulty in gathering data, lack of prior publications (esp. for Master's applicants), and penalizing applicants that mostly have industry experience.

5.4.1 Vector Generation

For each one of the approaches mentioned above, vectors are generated for different feature types, filtering, and weighting options.

Feature types. Two types of vectors are derived to represent a query or a document: using the vocabulary of single words encountered in the text (*unigrams*), or using the keywords encountered in the same text (*mwe*[6]). While the first technique is straightforward, for the second technique we extract keywords from applicant statements of purpose (SOPs) using a state-of-the-art supervised keyword extractor [134] trained on two keyphrase extraction corpora. The first corpus consists of a set of 211 academic papers with keyword annotations [8], while the second corpus was released as part of the SEMEVAL 2010 Keyphrase Extraction Task [10] and also encompasses a set of 184 academic papers annotated for keywords. The extractor uses noun phrases and named entities as candidates, as well as surface, frequency, phraseness, and graph-based features; it performs shallow post-processing after extraction to remove punctuation.

Filtering. The unigrams and the keywords mentioned above are referred to in the ensuing experiments as *all*, since they do not undergo filtering. A second instance of these features is derived, based on whether they are associated with a Wikipedia article[7]; this list is referred to as *filtered*, and retains fewer, higher quality and more salient entries.

We should emphasize that all the vectors are constructed on keywords/unigrams extracted from SOPs rather than those appearing in the published articles. The SOP-derived keyword list / vocabulary tends to be more generic and concise, as applicants do not yet have an in-depth grasp of various research areas and their SOP is shorter than an article, thus allowing the vectorial space to model applicants more closely while also being more efficient.

Weighting options. The above feature types are weighted using three common weighting schemes: *binary*, term frequency (*tf*), and term frequency inverse document frequency (*tf.idf*).

Information retrieval framework. Using a query vector, document vectors are ranked with respect to their cosine similarity computed against the query vector, and the top k are retrieved by the system. The system predictions are evaluated against ground truth faculty-applicant pairings that were manually derived by a small group of faculty members. Performance was measured using standard precision, recall, and F-score at different ranks (k).

[6] "mwe" stands for multi-word expressions.

[7] Listing of article titles retrieved from https://dumps.wikimedia.org/

Overall, we construct 12 vector space models encompassing all the combination of parameters detailed above. The most robust results are obtained using: keywords and unigrams (for vocabulary), tf.idf (for feature weighting), and all and filtered (for filtering). As such, in the subsequent discussions we will focus on these variations. The baseline is represented through the combination *unigram all tf.idf*, namely using all the vocabulary encountered in the SOPs as unigrams with tf.idf weighting.

5.4.2 Matching Applicants and Research Areas

Our first use case scenario matches applicants and research areas. This scenario allows departmental faculty or staff to be provided with the best research areas for a given candidate, and then manually assign candidates to faculty in those areas, thereby simplifying the matching process. We explore two venues:

1. Applicant as query, research areas as documents.

2. Research area as query, applicants as documents.

Figure 5.1 shows the interpolated precision-recall curve for the first approach (SOP query, area documents), while Figure 5.2 shows the same metrics for the second approach (area query, SOP documents) all of these derived for rank $k = 5$. We note that the first approach performs significantly better, achieving an interpolated precision level of over 80%, compared to the best performing variation falling under the second approach, which achieves an interpolated precision level of approximately 60%. Focusing on the first approach, the best performing variation is *mwe all tf.idf*, but is closely followed by *mwe filtered tf.idf*. Given that the former uses approximately 44 thousand dimensions, while the latter uses only 6,022 dimensions, we can conclude that (1) modeling via multi-word keywords is significantly better than accounting for the entire vocabulary, and (2) filtering these keywords for saliency achieves a more compact and efficient model, without a meaningful drop in performance.

Figures 5.3 and 5.4 show the corresponding F-score curve for the two approaches, this time for different ranks. We notice that the best F-score of 61.8% occurs at rank 3 (21.2% higher than the corresponding baseline), while the second best F-score of 59.9% is encountered at rank 4 (where the baseline F-score is the highest, yet the prediction still surpasses it by 17.5%). A higher F-score is to be expected in this scenario compared to results achievable for matching students and faculty, since here we are limiting our match to 11 research areas.[8] We should stress that the optimal usability outcome for this task is repre-

[8]The random baseline in this scenario is 9.1%.

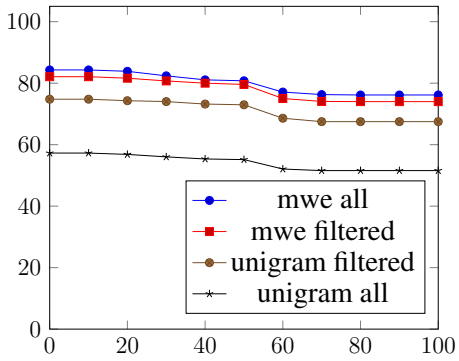

Figure 5.1: Interpolated precision-recall curves (at $k = 5$) showing the "SOP query, area documents" approach for matching applicants and research areas, with four variations. X-axis shows the recall level (%), while Y-axis shows the interpolated precision level (%).

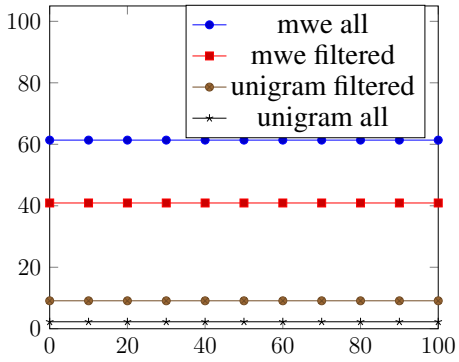

Figure 5.2: Interpolated precision-recall curves (at $k = 5$) showing the "Area query, SOP documents" approach for matching applicants and research areas, with four variations. X-axis shows the recall level (%), while Y-axis shows the interpolated precision level (%).

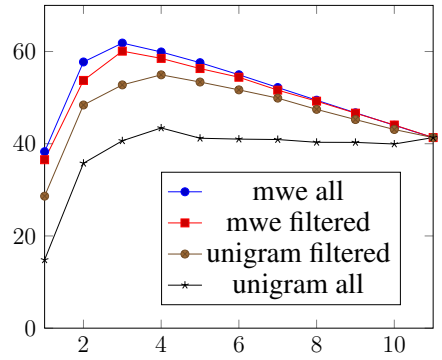

Figure 5.3: F-score curves for the "SOP query, area documents" approach for matching applicants and research areas, with four variations. X-axis shows the Rank, while Y-axis shows the F-score (%).

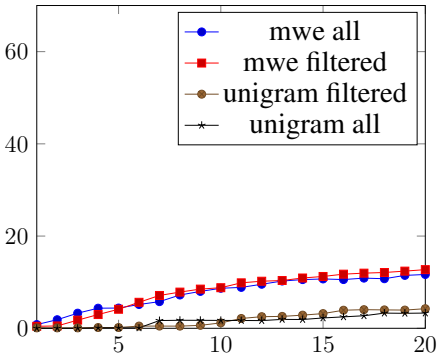

Figure 5.4: F-score curves for the "Area query, SOP documents" approach for matching applicants and research areas, with four variations. X-axis shows the Rank, while Y-axis shows the F-score (%).

sented through high performance at low ranks, i.e. the system should *correctly* retrieve a few matching research groups for a given SOP; as shown, the system achieves very high performance for ranks 2 through 4. This should accurately guide the process of assigning professors from those top retrieved groups and reduce the amount of manual work involved.

5.4.3 Matching Applicants and Faculty

The second and more desirable scenario consists of matching applicants and faculty. This allows the entire task to be automated, and therefore provides most savings in terms of financial and human resources for a department. We identify three venues:

1. applicant as query, faculty members as documents

2. faculty member as query, applicants as documents

3. applicant as query, faculty members as documents – hierarchical

The first two are similar to those proposed in the previous section, but this time the match is done directly with the faculty member, while the third consists of a hierarchical approach, where the match is first performed in regards to the best matching research group, and then the faculty is retrieved from within that group.

Probing further into the behavior of our system and baseline, we plotted the *interpolated precision-recall curve*, averaged over all search queries at a rank k=5. The resulting graphs are shown in Figures 5.5 through 5.7. We observe that similarly to the equivalent variations in Section 5.4.2, the SOP (applicant) query-based retrieval outperforms faculty query-based retrieval under all variations (see Figures 5.5 and 5.6). This is to be expected, since in the first scenario the retrieval is made among 73 faculty members, while in the second scenario, it is made among 304 applicants. Enacting a hierarchical based approach which generates an intermediary mapping to research areas and then retrieves the strongest matching faculty candidates from within the returned areas, achieves a similar performance to the first approach directly mapping to faculty. (see Figure 5.7). As in the previous subsection, the best variation remains *mwe all tf.idf*, with a performance of approximately 40% interpolated precision level, 35% higher than the *unigram all tf.idf* baseline achieving slightly below 5% interpolated precision level. This shows that keywords rather than vocabulary offer a lot more plasticity and bring more value for this task.

Figures 5.8 through 5.10 showcase the performance of the three approaches matching applicants to faculty at different ranks. Here as well, the *SOP query, faculty documents* represents the highest performing use case, retaining its high performance at low k values by achieving a F-score higher than 19% for ranks 2 through 10 using the best performing *mwe all tf.idf* variation, and reaching a maximum of 24.4% for rank 3. The hierarchical system displays a similar performance, as it is able to attain an F-measure above 19% starting at rank 2 as well, but since it is a two step system, it is too inefficient compared to a one step system to motivate its usage. The random baseline accuracy for matching an applicant to a faculty member is 1.4%.

Considering all the use cases, however, we can say that we are able to successfully retrieve research areas and faculty members against SOP queries. Our system always surpasses the baseline by a wide margin, and using the *mwe all tf.idf* variation consistently achieves the best results. This is a great boon for the faculty members and staff members, because instead of manually sifting through hundreds of applications, they can now use our system to screen applicants before starting the laborious manual checking process. Anec-

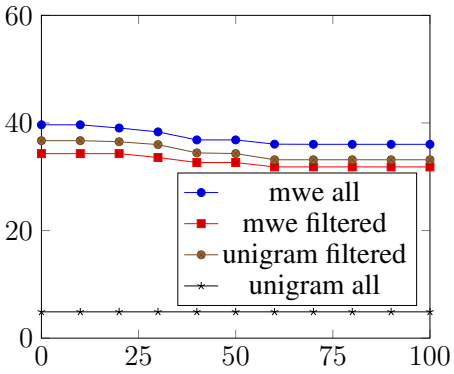

Figure 5.5: Interpolated precision-recall curves (at $k = 5$) showing the "SOP query, faculty documents" approach for matching applicants and faculty, with four variations. X-axis shows the recall level (%), while Y-axis shows the interpolated precision level (%).

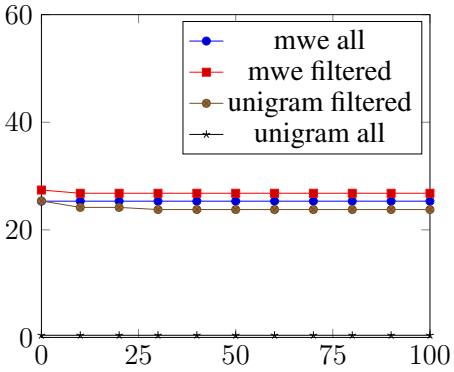

Figure 5.6: Interpolated precision-recall curves (at $k = 5$) showing the "Faculty query, SOP documents" approach for matching applicants and faculty, with four variations. X-axis shows the recall level (%), while Y-axis shows the interpolated precision level (%).

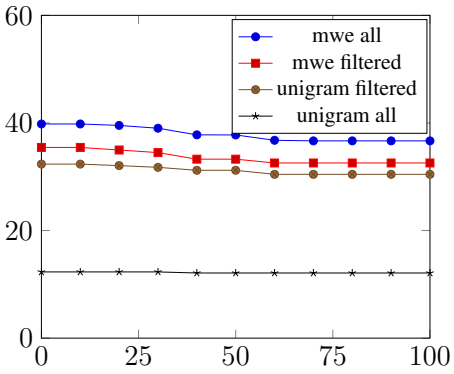

Figure 5.7: Interpolated precision-recall curves (at $k = 5$) showing the "SOP query, faculty documents – hierarchical" approach for matching applicants and faculty, with four variations. X-axis shows the recall level (%), while Y-axis shows the interpolated precision level (%).

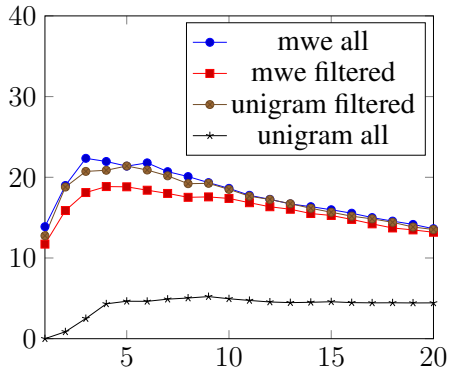

Figure 5.8: F-score curves showing the "SOP query, faculty documents" approach for matching applicants and faculty, with four variations. X-axis shows the Rank, while Y-axis shows the F-score (%).

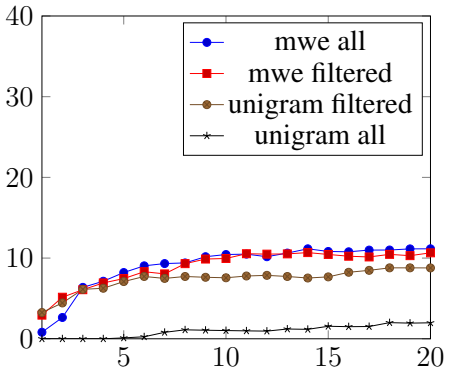

Figure 5.9: F-score curves showing the "Faculty query, SOP documents" approach for matching applicants and faculty, with four variations. X-axis shows the Rank, while Y-axis shows the F-score (%).

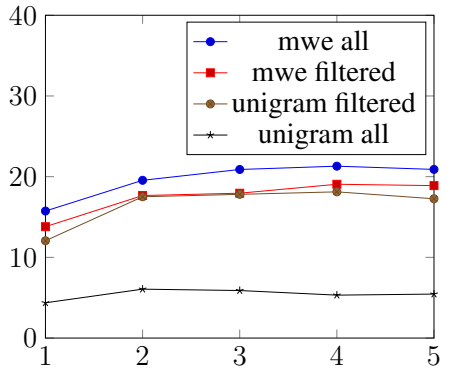

Figure 5.10: F-score curves showing the "SOP query, faculty documents – hierarchical" approach for matching applicants and faculty, with four variations. X-axis shows the Rank, while Y-axis shows the F-score (%).

dotal evidence from faculty members in our department showed that this was indeed the case, and they were happy with the search results produced by our system.

5.5 Conclusion

In this chapter, we introduced a new task – matching graduate applicants with faculty members using text-based features. The problem is complex, given that there are no standard annotated datasets, not much relevant related work, and *content disparity* between the textual materials authored by applicants and those authored by faculty members. We created our own dataset comprising 4,534 papers authored by 73 different faculty members at the Computer Science and Engineering department of a large Midwestern university. We further considered an in-house set of 1,107 statements of purpose, and a set of 788 faculty-applicant pairings constructed manually. Keywords were extracted from papers and SOPs, and a detailed exploratory analysis was performed leading to insights regarding the content depth and subtlety of documents. We obtained encouraging results using standard information retrieval techniques using five different use cases, concluding that keywords offer a significantly better representation (more efficient and better results) compared to bag-of-words variations. Overall, we are able to match students to research groups with an F-score of 62%, while for the more difficult task of matching students to faculty, we are able to achieve a 24% F-score. Our future work includes obtaining more data (especially applicant data), more reliable faculty-applicant annotations, and more sophisticated models that take into account the sparsity of the task.

CHAPTER 6

Using Keyword Extraction to Predict Student Performance: Data-Driven Keyword Extraction on Piazza Forum

Prediction of student performance from their participation in academic forums has received a lot of attention in recent years. However, the question of *whether we can predict student performance from their textual contributions in online forums*, has not been thoroughly investigated. In this chapter, we examine whether we can predict student grade in a large undergraduate course consisting of 600 students. We further examine the intriguing role of keyphrases, stopwords, and LIWC categories [16] in this prediction task. Note that the conventional approach of using forum metadata to predict student performance (more details in Section 2.7) is complicated by the fact that forum metadata can be complex, with spatio-temporal interactions [136], whereas text data is much simpler to deal with, and is also available in abundance.

6.1 Introduction

With an ever-increasing number of students in both undergraduate and graduate courses all across the United States,[1] the question of assessing student performance early in the course, and giving students timely feedback assumes paramount importance. Our goal is to automate, or at least complement the process of manual assessment of performance. In this chapter, we aim to examine whether student performance can be predicted from text-centered signals extracted from posts written by students.

[1] https://nces.ed.gov/fastfacts/display.asp?id=98.

Table 6.1: Grades received by students in a large undergraduate course.

Letter Grade	Numeric Grade (range)	# Students in the Grade Bucket
A	90-100	70
B	80-89	220
C	70-79	69
D	60-69	15
F	30-59	4
Total	**0-100**	**378**

Table 6.2: Statistics of the Piazza data. Average was taken across all students in a particular grade bucket. Numbers in parentheses are standard deviations.

Grade Bucket	Avg # posts	Avg # words	Avg # unique words
A	23.67 (38.41)	964.73 (1299.51)	297.67 (297.72)
B	10.69 (16.92)	426.30 (669.89)	170.47 (164.66)
C	11.45 (17.10)	500.46 (729.87)	189.00 (185.86)
D	13.13 (16.12)	506.73 (589.27)	205.40 (199.06)
overall	15.14 (26.16)	622.79 (948.67)	218.27 (227.62)

We start out by reviewing the recent work on student performance prediction, using non-textual (Section 2.7.1) and textual data (Section 2.7.2). Section 6.2 discusses our dataset, which was constructed around the online participation of students enrolled in an on-campus 600-strong undergraduate course at a US university. Section 6.3 describes the task and how it was framed as a machine learning problem, starting with feature engineering and experimental setup and leading to the results and our findings. Section 6.4 concludes the chapter with contributions and future research directions.

6.2 Dataset

Piazza is an online portal aimed at supporting the classroom setting by providing a centralized space for course materials and outside classroom discussions. It combines elements from the traditional wikis and online forums, allowing course participants to post, view and edit each other's questions, answers, and notes, therefore resulting in increased levels of student engagement.[2] Piazza was established by Pooja Sankar in 2009, opened worldwide to institutions in January 2011, and reached over 330 schools and tens of thousands of students by Summer 2011.

On Piazza, users can publicly (and anonymously, if the head instructor allows it) ask questions, answer questions, and post notes. Each student-authored question, answer, or note can be collaboratively edited by all students and faculty members, while each instruc-

[2] https://en.wikipedia.org/wiki/Piazza_(web_service).

Table 6.3: Statistics of the Piazza data. Average was taken across all posts made by students in a particular grade bucket. Numbers in parentheses are standard deviations.

Grade Bucket	Avg post length (in words)
A	40.75 (38.99)
B	39.89 (34.05)
C	43.71 (100.04)
D	38.58 (31.24)
overall	41.13 (58.21)

tor answer can only be collaboratively edited by other instructors. The platform further allows private questions and private notes. Users are allowed to attach external files to posts, use LaTeX formatting, view a post's edit history, add follow-up questions, respond to them (known as "feedback"), and receive email notifications when new content is added. The interface consists of a dynamic list of posts on the left side of the screen, a central panel for viewing and contributing to individual posts, and an upper bar for account control. Instructors have additional control, such as Piazza statistics and trends. The platform is available on iOS and Android mobile operating systems.

The Piazza metadata is fairly complex, including timestamps, five types of posts (question, student answer, faculty answer, feedback, followup), content privacy, question ratings, and answer endorsements.

We designed our own dataset from the Piazza forum discussions[3] of a large undergraduate course consisting of 600 students in a Midwestern university. 378 students participated in the Piazza discussions. Their grades are shown in Table 6.1. We assigned a letter grade to each grade bucket. As can be observed from Table 6.1, grade bucket "B" is the majority class, with 58.82% of the total number of students (ignoring grade bucket "F").

Statistics on the first four grade buckets are shown in Table 6.2. We note that the majority of students fall under grade bucket "B". Note further from Table 6.2 that grade bucket "A" has the highest number of posts on average, and also the highest number of words and unique words, followed by grade buckets D, C, and B. When we look into Table 6.3, we see that the length of post (on average) is longest for "C" students, followed by "A", "B" and "D" students.

6.3 Task

We seek to predict the academic performance of undergraduate students in a large introductory course on Programming and Data Structures, part of the Computer Science un-

[3] https://piazza.com/.

Table 6.4: Most frequent keywords/keyphrases extracted from Piazza data.

Keyword	Frequency
autograder	80
function	74
error	67
code	64
player	60
list	56
card	53
test cases	46
project	44
problem	43

dergraduate curriculum. The task is framed as a four-class classification problem – by making predictions for grade buckets "A", "B", "C" and "D". We use a supervised learning approach to classify the data, where each instance consists of data pertaining to a given student enrolled in the course, represented as a feature vector; the class label for a student is the grade bucket (s)he falls under.

Preprocessing. We extract all the posts made by students from Piazza XML dumps,[4] and cleaned them by removing spurious HTML tags and markups. We employed several core NLP text processors (included with the Stanford CoreNLP [137] library), such as tokenization, sentence segmentation, part-of-speech tagging, constituency and dependency parsing, named entity recognition and lemmatization.[5]

6.3.1 Features

Below we introduce the feature types considered for the classification task.

- **Word grams:** We use the raw student text (lowercased and without punctuation) to extract word unigrams, bigrams and trigrams. In order to reward more rare tokens and to discount frequent ones, we use term frequency inverse document frequency (*tf.idf*) [138] weighting for the entries found in the corpus, where a document is the total textual contribution of a student in a semester. This is a numeric feature, and has been widely used in text classification tasks [139, 140, 141].

- **Keywords:**

[4]Piazza data is available to instructors upon request, along with all metadata as well as timestamps, in XML format.

[5]Stanford CoreNLP modules were run before lowercasing and removing stopwords.

In order to gauge whether higher achieving students are more specific in their writings, we focus on the specificity of the words employed. More precisely, we extract keywords from Piazza discussions using the state-of-the-art supervised keyword extractor developed recently [134], and trained on academic papers.[6] As can be seen from Table 6.4, the most frequent keywords are related to programming assignments. They are all highly relevant to the Piazza data, and contain important cues on student behavior around coding assignments ("error", "project", "problem", "autograder", "test cases"). The feature weighting used was term frequency (*tf*) [138], to boost the value of repeated keywords; *tf* was normalized by the token count.

- **Common words:** Given the problem of data sparsity in a vectorial space model built on word unigrams across a given student's participation, we also explore modeling the usage of common words. We use a list of the most frequent 5,000 English words[7] computed over the 560 million word Corpus of Contemporary American English (COCA), which is arguably the most accurate word frequency list for American English, as it is balanced for genre, focuses on contemporary English usage, and is very large. We used *tf* weighting for common words, normalized by the token count.

- **Stopwords:** Furthermore, we explore the usage of stopwords in student writings, as stopwords may be considered a measure of an author's stylistic signature [142, 143]. We used *tf* weighting, normalized by the token count, to give preponderance to text with a high stopword density. We experimented with two different lists, one consisting of the most frequent 173 stopwords (allowing us to make a more "barebone" comparison across students), and a more extensive list of 423 stopwords (which should be able to retain more context from a student's participation).[8] Our expectation is that the smaller stopword list is too aggressive in removing student content, even though it allows for denser representations, while the longer list may be optimal in allowing the right amount of content to create dense representation while still retaining vocabulary differences across people. Examples of words that appear in the common word list, but not in the stopword lists (and vice versa), are shown in Table 6.5.

- **Lexical features:** It has been observed in Authorship Attribution literature [68, 69]

[6] It may be argued that Piazza and academic papers are two distinct genres; why is it sound to have the keyword extractor trained on academic papers, and predict on Piazza data? To this question, we respond: although the genres are different, what the supervised keyword extractor learns, is a *mapping* from document to keywords, not the properties of documents themselves. This way, the keyword extraction pipeline works seamlessly across all domains (cf. Chapter 3).

[7] https://www.wordfrequency.info/top5000.asp.

[8] https://www.ranks.nl/stopwords.

Table 6.5: List of words that appear in the common words list, but not in the stopwords lists, and vice versa. Note that all the common words start with "ab"; this is not a coincidence; common words were sorted in lexical (ascending) order, and since there are many of them as compared to stopwords, we see a profusion of words that start with "ab".

Common Words	**Stopwords (short)**	**Stopwords (long)**
abandon	am	areas
ability	aren't	asked
able	can't	asking
abortion	couldn't	asks
abroad	didn't	backed
absence	doesn't	backing
absolute	doing	backs
absolutely	don't	became
absorb	hadn't	becomes
abstract	hasn't	began

that word and character n-grams can capture an author's fingerprint. We wanted to see if these authorial choices are indicative of students' performance in class. Each student was represented as a *tf* weighted instance of character trigrams, normalized by the token count.

- **Syntax features:** Similar to word and character n-grams, POS n-grams and parse trees (constituency and dependency) have also been used in Authorship Attribution to link them to the identity of an author [144, 145]. We therefore also explore the relationship between syntactic features and student performance. Specifically, we use terminal rules (part-of-speech to token) for constituency, dependency heads (e.g., *amod*) for dependency, and POS unigrams and bigrams for parts-of-speech. Further, we use *tf* weighted instance of syntax features, normalized by the total number of times all elements of that feature type appears for a given student.

- **Embeddings:** Word embeddings capture in a very dense vectorial space "hidden" or "latent" dimensions in text data, enabling words that occur in a similar context, to become related and inhabit closer together in this space, while words that do not share context to drift apart. Typically, the larger the size of the corpus on which the embeddings are refined, the more accurate these representations are. Shorter training windows allow functionally and semantically similar words to be better represented, while longer training windows allow for a more thematic clustering of words in the vectorial space. Embeddings can also be seen as a method of dimensionality reduction, similar to LDA [94], LSA [146], or PCA [147], and thus are extremely efficient in algorithmic computations. Since the initial vocabulary for our dataset is 9440

tokens, allowing for very sparse representations, embeddings allow contexts to be represented through a very limited number of dimensions (in our case 300), and their performance is typically significantly higher compared to word-based vectorial space models. While using neural nets for natural language applications was not a tractable proposition before, this changed with the breakthrough word embedding model using a shallow neural net architecture proposed by Mikolov et al. [111], and the fury of neural net models that came after that.

In our evaluations we experiment with two settings. First we focus on using pre-trained word and phrase embeddings using a well-formed genre (such as newswire) released by Google News [111]. The advantage of using such embeddings is that they are typically very accurate, as they were refined on a very large volume of news articles that are not publicly available. However, these embeddings are best able to capture generic word meanings, and they may not be particularly well-suited for a computer science jargon-laden course, where embeddings for words such as "child" or "orphan" may be quite different compared to their generic counterparts. We therefore also experiment with training our own embeddings on the Piazza data, using the skip-gram variation with hierarchical softmax and negative sampling of the word2vec model [111] and a window size of 10, in order to enable deriving embeddings with similar settings to those that were publicly released (i.e. 300 dimensions), and compare their performance.

We represent a student's contribution by performing vectorial summation over the individual embeddings pertaining to component words and normalizing with respect to the number of tokens employed by the student. This allows for the emergence of heavy weighted and light weighted dimensions regardless of the length of a student's contribution. Separate models were obtained for representing a student's writing as unigrams, keywords, common words, and stopwords. We should note that keywords were also modeled using the *word2phrase* embeddings [111], also a skip-gram neural network architecture that extracts meaningful phrases from bigrams using discounted mutual information, and derives phrase based embeddings that are more accurately represented, allowing modeling beyond the sum of the parts.

- **Word categories:** The Roget thesaurus is a widely used English-language thesaurus, created in 1805 by Peter Mark Roget (1779-1869), British physician, natural theologian and lexicographer.[9] It was released to the public on April 29, 1852. The original edition had 15,000 words, and subsequent editions have added many more words

[9] https://en.wikipedia.org/wiki/Roget%27s_Thesaurus.

[148]. It is structured like a tree, where at the first level it is split in approximately 1,000 categories under which all of its words are nested. Once we encounter a word in the thesaurus, we replace it by its heading category. A student instance is weighted by the *tf* of the categories encountered, normalized by the token count. This method acts as a form of supervised dimensionality reduction.

- **Psycho-linguistic features:** We aim to model the psycho-linguistic facets expressed by the students in their writings using the Linguistic Inquiry and Word Count (LIWC) [16] tool. LIWC analyzes the way people employ language through the prism of 73 categories, encompassing standard linguistic dimensions (such as pronouns, articles, verb tense, etc.), psychological processes (such as cognition, affect, social, etc.), personal concerns (work, home, leisure, etc.) and spoken categories (assent, fillers and non-fluencies). Ultimately we want to see if students' psychological and personality characteristics as inferred from text are correlated with their classroom performance.

- **User-centered features.**

  - **Formality:** Text formality has been measured by the percentage difference between nouns, adjectives, articles, prepositions; and pronouns, verbs, adverbs, interjections [124]. It has been observed that as the frequency of nouns, adjectives, prepositions and articles in a given text increases, its formality increases. On the other hand, as the frequency of pronouns, verbs, adverbs and interjections in a given text increases, its formality decreases.

  - **Lexical density:** It is the estimated measure of content per functional (grammatical) and lexical units (lexemes) [125]. Lexical density is used in discourse analysis as a descriptive parameter which varies with register and genre. Spoken texts tend to have a lower lexical density than written ones.[10]

  - **Concreteness:** We want to see whether success in the classroom correlates with the usage of concrete vocabulary. To represent concreteness, we employ the set of words, along with their concreteness scores, available as part of the Free Association Norms Database [126]. There are more than 3,000 words available as part of the database.

- **Spatial Prepositions:** Recent studies have shown prepositions to be an important source of semantic information [129, 130, 131]. We use a lexicon of *spatial preposi-*

[10] https://en.wikipedia.org/wiki/Lexical_density.

tions[11] as a bag-of-words feature. The rationale was to observe if spatial properties of programming constructs ("integer **within** given parameters", "trees that are **beneath** the tree") could be captured in terms of prepositions such as *on*, *in*, *by*, *with*, etc.

- **Metadata features:** Piazza has a rich and complex metadata structure, we use seven features from this structure: number of answers a student received on Piazza, number of feedback texts received by the student, number of followup texts received by the student, number of notes the student posted,[12] number of private posts the student made, number of questions the student made,[13] and number of views the student's posts got. We used metadata features as a comparison point against text features, and to see how much leverage in classification may be obtained from metadata alone, without using text.

- **Time Features:** We have four types of time features; (1) day of the week features: based on the 7 days, ratio of the number of posts in that day to the total number of posts contributed by a student; (2) week features: breaking the semester into weeks, the student's weekly participation (ratio of posts contributed in a given week to the total number of posts); (3) overall semester participation feature: total number of weeks in which the student has posted something; and (4) time of day features: considering morning (5am - 11am), afternoon (11am - 5pm), evening (5pm - 9pm) and night (9pm - 5am), the features are the ratio of posts contributed in these time slots divided by the total number of posts made by a student.

6.3.2 Algorithms

We used three different classification algorithms from the scikit-learn library [149] with default parameter settings:

Support Vector Machine (SVM). SVM is a widely used maximum-margin classifier that works on the principle of kernel functions [90]. The particular kernel function is used to separate the classification space whether in a linear way (a plane) or using a radial basis function. We used LinearSVC – the linear basis function kernel.

Logistic Regression. Logistic Regression is another powerful classification algorithm that has seen wide adoption across many different domains, including but not limited to text

[11] Obtained by combining the two lists at https://owl.english.purdue.edu/owl/resource/594/04/ and http://www.firstschoolyears.com/literacy/sentence/grammar/prepositions/resources/Spatial%20Prepositions%20word%20bank.pdf.

[12] If a note has multiple authors, each author gets a contribution.

[13] If a question has multiple authors, each author gets a contribution.

classification [91]. Here, a sigmoid function is used to map the classification answers into 1 or 0 (where 1 is the correct class, and 0 is the other class).

Decision Tree. Lastly, decision trees are widely used classification algorithms that have the advantage of simplicity, understandability, and robustness [150]. For the decision tree, a nested decision takes place; i.e. at every node some feature-related rule directs the decision to go on one branch or another. Ultimately the leaves in the tree represent class labels.

6.3.3 Experimental Setup

6.3.3.1 Settings

We used three classification algorithms due to their diverse learning methodologies, namely SVM LinearSVC, Logistic Regression, and Decision Tree. These are available from the scikit-learn library [149] and we trained them with default parameter settings. All results presented are obtained as a result of 10-fold cross-validation.

6.3.3.2 Baseline

The unsupervised baseline, given a 4-class (balanced) prediction problem, is 25%. We compare our results with the supervised baseline, which predicts the majority class label for all instances in the test set, achieving an accuracy of 31.25%.

6.3.3.3 Evaluation metrics

The performance metric used in the evaluation is classification accuracy. It is defined as:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN},$$

where TP = True Positive (number of instances that have been correctly classified as belonging to the target class), TN = True Negative (number of instances that have been correctly classified as *not* belonging to the target class), FP = False Positive (number of instances that are not in the target class, but have been incorrectly classified to be so), and FN = False Negative (number of instances that are in the target class, but have been incorrectly classified not to be so).

We compute micro-accuracy by considering all the predicted and the real class values for each instance in the 10 test folds.

6.3.4 Results and Discussion

Below we explore the machine learning evaluation results over three major groupings of features, namely text-derived, non-textual, and hybrid.

6.3.4.1 Text-derived features

Table 6.6 shows the various feature types we evaluated and the results obtained, starting with text-derived features, non-text features, and hybrid text/non-text features. We will discuss each section below.

As part of the text-derived features, we have explored three major groups, namely word representations based on vector space models, word representations using embeddings, and other text-derived features. We notice that across all three classifiers, the bigram word model obtains the highest accuracy: in the case of LinearSVC being 4.91 accuracy points ahead of the next best word-focused representation, for Logistic Regression, surpassing the next best by 0.44 points, and for Decision Tree by 4.91 points. We note that despite tf.idf weighting for unigrams, which should reduce the emphasis on words that are frequently used by all students, there is still too much noise for a strong representation. Focusing on keywords instead yields better results across the three classifiers, accentuating though the problem of matrix sparsity, which ultimately leads to mediocre results. In order to have a denser representation, we explore the modeling ability of common words, starting with the most frequent words (the stopwords short list), to slightly less frequent words (the stopwords long list), to the most frequent 5,000 American English words. We note that while the shortest stopword list of only approximately 150 features is able to surpass the performance of the unigram model encompassing 9440 features across all classifiers, its performance is slightly below that of keyword-based classification (surpassing 2 classifier accuracies out of 3). Expanding the stopword list to 423 features, leads to the second strongest performance across the VSM models explored, second only to bigrams, with only a small drop registered for decision trees (of 0.44%), yet a significant boost for LinearSVC by 3.57% and for Logistic Regression, by 3.13%. Further expanding the classification space to allow for 5,000 features originating from the most common words does not result in significant gains, achieving a drop by 1.78% for LinearSVC, falling below the majority class prediction accuracy for decision trees, and increasing merely by 0.89% for Logistic Regression-based accuracy, while increasing 11.8 folds the dimensionality of the space.

Now we explore the ability of embeddings to model student performance. We first look at the performance of pre-trained embeddings, followed by our own trained embeddings derived on the Piazza data. As we see from Table 6.6, pre-trained embeddings display a

similar performance to word unigrams, but using only 300 dimensions, compared to 9440 for word unigrams. Representing only keywords using the pre-trained embeddings achieves a similar performance to word-based embeddings across the three classifiers. Turning our attention to the embeddings we trained on the Piazza data, we note a boost in performance across the LinearSVC classifier (by 1.78 points compared to pre-trained embeddings), the Logistic Regression classifier (by 5.35 points), and the decision tree classifier (by 8.49 points). This particular combination is the strongest across all embedding variations, falling only 0.4% accuracy points short of the best performing Logistic Regression embedding combination, yet surpassing the next best embedding-based accuracy by 2.68 points for LinearSVC and 1.33 points for Decision Tree. In the case of embeddings, modeling all the words appearing in the context achieves the best results. Trying to scale the emphasis of a given word embedding based on the document frequency of a word proportionately across all latent dimensions produces a drop in accuracy compared to simply representing the entire context through the summation of the underlying word embeddings. Keyword-based embeddings that were trained on the Piazza data display a lower representation ability compared to the generic pre-trained embeddings, probably because sufficient textual data is not available in order to derive a robust model. Phrase-based keywords trained embeddings (using the word2phrase model) show a marginally better performance for LinearSVC and Logistic Regression compared to those obtained using word2vec, but for decision tree, accuracy drops significantly. Between using pre-trained generic embeddings and embeddings refined on the Piazza data to represent student contexts, we see a significant added value from training on in-domain data, by 1.78 points for Linear SVC, by 5.35 points for Logistic Regression, and by 8.49 points for Decision Tree.

In terms of other text-based features, each of the signals considered displays a strong performance for one of the classifiers. Roget categories for example perform well for LinearSVC and Logistic Regression, by 35.27% and 36.61%, respectively, while spatial prepositions (with only 49 features) show the strongest performance for Decision Tree across all the textual features considered. Similarly, LIWC and user-centered features have a very good performance for decision trees across all textual feature types, in particular when considering that they only have 73 and 3 features, respectively. Combining all *other text-based features*, we obtain a better performance than with individual feature types only for logistic regression (by 1.78 points) compared to the top performing Roget categories; other than that, performance is inferior to both word categories and psycho-linguistic categories.

In terms of classifier, we note that the most robust performance is obtained by the LinerSVC model, both in terms of how often it surpasses the other two classifiers, as well as the highest accuracies attained for individual feature types. In the case of VSM features, it

is able to outperform or match the Logistic Regression algorithm for 8 out of 9 evaluations, and the Decision Tree algorithm for 6 out of 9; for Syntax features, it outperforms or matches the Logistic Regression algorithm in 4 out of 5 evaluations, but it performs better than Decision Trees in only 2 out of 5 evaluations; for Embeddings, it outperforms Logistic Regression in 5 out of 9 combinations, and it surpasses or matches decision trees in 9 out of 9 combinations; for other text-based features, LinearSVC performs better than Logistic Regression in 2 out of 5 scenarios compared to logistic regression, and 2 out of 5 scenarios compared to decision trees. For non-text features, LinearSVC performs better in 1 out of 3 evaluations and 2 out of 3 evaluations compared to Logistic Regression and Decision Tree, respectively. We note that decision trees consistently perform poorly on embeddings models, achieving worse results across all trained embeddings evaluations, and they only perform well on pre-trained word embeddings; for pre-trained keyword embeddings, the accuracy matches the baseline majority class accuracy.

Ultimately, looking at text-based features, we note that: (a) boosting keywords / rare words does not help much in this task; (b) unigram embeddings perform better than stopword embeddings whether for short or long stopword lists across all classifiers; (c) pre-trained embeddings for keywords perform better than trained embeddings for keywords, because rare words take more time (and more data) to derive accurate embeddigns; and (d) pre-trained embeddings for words perform worse than trained words, because the forum style is quite different from newswire genre, therefore generic word embeddings are more robust on the same type of data.

6.3.4.2 Non-text features

As a comparison point, we further present the result obtained from seven Piazza Metadata Features. In this experiment, we choose seven relatively simple metadata features (cf. Section 6.3.1), and as shown in Table 6.6, metadata features perform poorly, as only one out of three classifiers is able to surpass the majority class baseline of 31.25%, and by only 2.6 percentage points (using the Logistic Regression algorithm). We note that text-based features almost always achieve a stronger performance, in particular for single feature types. We further observe that despite the small number of features, text-derived features such as user-centered features (three individual features: formality, lexical density and concreteness) have a stronger performance across the board, for LinearSVC by 3.12 percentage points, for Logistic Regression by 0.89 percentage points, and for Decision tree by 9.82 percentage points. This implies that text-derived features are particularly well-positioned to predict student performance.

We also engineered a set of time-based features which exhibit the strongest performance

across all the feature types and across the three classifiers explored. With only 30 features, we obtain overall accuracies over 42%, with LinearSVC reaching the highest accuracy at 46%. This shows that student involvement throughout the semester and during particular days of the week is the best predictor we looked at in terms of classroom performance. Combining both time and metadata features does not result in a stronger performance than time features alone. While our focus was not to exhaustively explore non-text-derived features, these evaluations provide a window into what can be expected when considering metadata related to posts, and how those results compare to what can be gathered from text-derived features alone.

6.3.4.3 Hybrid features

We further explore whether by combining non-text and text-derived features into a hybrid space we can form a stronger representation. The best performance for hybrid features is achieved when combining Roget and time-based features (43.75% for LinearSVC, and 41.52% for Logistic Regression). Combining four *other* text features (Roget, psycholinguistic, user-centered, and spatial prepositions) with unigram embeddings yields the best performance for decision tree (33.48%). It is to be noted that LinearSVC is the best performer among the three classifiers, followed by Logistic Regression and decision tree, respectively. However, the second best performance comes from LinearSVC (33.48%) and decision tree (32.14%) for the combination of four *other* text features, unigram embeddings, and POS unigrams. For Logistic Regression, the second best performance comes from time features combined with *other* text features (40.18%). Among three classifiers, LinearSVC best performance is 10.27 points higher than the second best, whereas for Logistic Regression and decision tree it is 1.34 points.

Compared with text features, the hybrid features (Roget and time features combined) outperform all text features and embedding features – by 3.57 points for LinearSVC, and by 2.68 points for Logistic Regression. For decision tree, however, we observe that the best-performing text features outperform the best-performing hybrid features by 6.25 points.

Among the hybrid feature categories, Roget combined with time-features yields the best results (2 out of 3 classifiers), whereas four *other* text features combined with unigram embeddings yields the second best results (1 best out of 3). Comparing hybrid features with non-text features, we observe that across classifiers, time features outperformed hybrid features (by 2.68 points for LinearSVC; by 2.23 points for Logistic Regression; by 8.48 points for decision trees), whereas hybrid features outperformed metadata features (by 15.62 points for LinearSVC; by 7.59 points for Logistic Regression; by 8.03 points for decision trees). Combination of all non-text features performed better than hybrid features

(by 2.23 points for Logistic Regression; by 6.70 points for decision trees), except for the LinearSVC classifier.

6.3.5 Correlation of student performance with text-based features

We further focus on a qualitative assessment of which words are more closely correlated with classroom-based student performance. As such, we explore differences in keyword usage by students, then we look at psycho-linguistic categories derived from words, and we conclude with stopword usage. For keywords and words, we normalized occurrence frequencies per grade; for LIWC categories, we also normalized categories to which the words were mapped to per grade; this ensures that the size of data available for a given grade will not skew our observations. We compute the Pearson correlation between these normalized frequencies and the student grade, by mapping A to 4, B to 3, C to 2, and D to 1. Pearson correlation takes values from -1 to 1; negative values indicate inverse correlations, while positive values indicate direct correlations. The strength of association is ranked as: small (for values between 0.1 and 0.3), medium (for values between 0.3 and 0.5), and large (for values above 0.5).

6.3.5.1 Keywords

Table 6.7 lists 3 sets of keywords grouped around 1, highly correlated with student performance, 0, not predictive of student performance, and -1, inversely correlated with student performance. A direct correlation between the mentioning of outside help and student performance is noted. For example, high performing students use vocabulary such as *instructor*, *office hours*, *professor* often, ranking in the top 20 keywords associated with performance out of a total of 1,759 keywords. We also note that top performing students' keywords are grouped around tests, i.e. the code is already written and running, and they are focusing on testing it. We note *helper function, test, null, valid, bug* keywords in the top 10 most predictive of performance; at the same time, students that receive lower grades deal with more basic programming issues, related to data structures (*struct*), coding (*namespace, get param, friend class*), resource-based problems (*memory, space*), and actually getting the code to compile (*compiling*).

Among the least correlated keywords to performance, we encounter *extra*, *highest*, *solutions*, *sum* and *private members*, which show a correlation of +/- 0.01.

Table 6.6: Classification performance (10-fold CV Accuracy (%)) for different feature categories and classifiers. Best result is boldfaced. All embeddings are lc (lowercased), 300 dimensions, with window size 10, skip-gram model, and optimization type NSHS (negative sampling with hierarchical softmax). **Boldfaced** values in each column represent the best (within each section), and *italicized* values represent the second-best.

Feature	Note	# Features	LinearSVC	Logistic Regression	Decision Tree
Baseline					
Majority class-based prediction			31.25	31.25	31.25
VSM					
unigrams	tf.idf	9440	33.04	33.04	29.02
bigrams	tf.idf	70090	**40.18**	**37.05**	**38.39**
trigrams	tf.idf	132625	34.38	*36.61*	29.91
keywords	tf norm.	2625	31.70	29.02	*33.48*
5000 most common words	tf norm.	4352	35.27	33.93	32.59
stopwords short list	tf norm.	173	35.27	34.38	26.79
stopwords long list	tf norm.	423	33.93	32.59	27.23
Character ngrams	tf norm.	18811	*37.05*	*36.61*	29.46
Combinations					
keywords + stopwords long	tf norm.	3048	30.80	30.80	34.38
Syntax features					
POS unigrams	tf norm.	45	*34.38*	31.70	24.55
POS bigrams	tf norm.	1405	33.04	*33.04*	**37.05**
Constituency	tf norm.	18737	**34.82**	**34.38**	28.13
Dependency	tf norm.	276	27.68	28.13	*28.57*
Combinations					
POS uni + POS bi + dependency	tf norm.	1726	30.80	28.57	32.14
All	tf norm.	20463	31.70	28.57	36.16
Embeddings					
pre-trained embeddings					
word		300	33.93	32.14	33.93
keywords		300	33.04	33.93	31.25
trained embeddings					
unigrams		300	**40.18**	*38.39*	**35.71**
stopwords long		300	*37.50*	*38.39*	33.48
stopwords short		300	36.61	**38.84**	*34.38*
common words		300	35.71	34.38	28.57
keywords (unigrams,tf)		300	32.59	31.25	27.68
keywords (phrase,tf)		300	34.38	32.59	22.32
unigrams tf.idf		300	35.71	*38.39*	*34.38*
Other Text-based Features					
Word categories (Roget)	tf norm.	1069	**35.27**	**36.61**	31.70
Psycho-linguistic (LIWC)	tf norm.	73	*33.48*	33.48	*36.61*
User-centered features		3	31.25	*34.82*	35.27
Spatial prepositions	tf norm.	49	32.14	32.14	**39.73**
Combinations					
LIWC + user-centered + spatial	tf norm.	125	29.02	33.93	29.91
All	tf norm.	1194	33.04	**38.39**	27.68
Non-text Features					
Metadata features		7	*28.13*	*33.93*	*25.45*
Time features		30	**46.43**	**43.75**	**41.96**
Combinations					
All		37	34.82	**43.75**	40.18
Hybrid Features					
Other text (all) + time features	tf norm.	1224	32.14	*40.18*	25.45
Other text (all) + stopwords (short)	tf norm.	1367	28.57	37.50	26.34
Other text (all) + embedd. (uni.)	tf norm.	1494	32.59	37.95	**33.48**
Other text (all) + embedd. (uni.) + POS unigrams	tf norm.	1539	*33.48*	36.61	*32.14*
Roget + time features	tf norm.	1099	**43.75**	**41.52**	30.36

Table 6.7: Keywords and their Pearson correlation with course performance.

Keyword	Correl	Rank	Reverse Rank
Words most correlated with student performance			
helper function	1.00	1	1760
test	1.00	2	1759
null	1.00	3	1758
valid	1.00	4	1757
characters	1.00	5	1755
hard	1.00	6	1756
bug	1.00	7	1754
running	0.99	8	1753
orders	0.99	9	1751
person	0.99	10	1752
length	0.99	11	1750
member functions	0.99	12	1749
track	0.99	13	1748
instructor	0.99	14	1747
numbers	0.99	15	1746
member	0.99	16	1745
office hours	0.99	17	1744
account	0.98	18	1743
recursive	0.98	19	1742
professor	0.98	20	1741
Words that are not associated with student performance			
extra	0.01	1200	561
highest	0.00	1201	560
solutions	-0.01	1205	557
ugli	-0.01	1206	558
add_card	-0.01	1203	555
sum	-0.01	1202	559
private members	-0.01	1204	556
Words that are inversely correlated with student performance			
wrong	-0.91	1741	20
xcode	-0.91	1742	19
worth	-0.91	1743	18
fraction	-0.92	1744	17
link	-0.92	1745	16
friend class	-0.93	1746	15
score	-0.93	1748	14
partnership	-0.93	1747	13
compiling	-0.94	1749	12
getparam	-0.94	1750	11
memory	-0.95	1751	10
files	-0.96	1752	9
space	-0.96	1753	8
shuffling	-0.97	1754	7
deal	-0.97	1755	6
wager	-0.97	1756	5
struct	-0.98	1757	4
namespace	-0.98	1758	3
specs	-1.00	1759	2
names	-1.00	1760	1

Table 6.8: LIWC categories and their Pearson correlation with course performance. (1) Category, (2) Category Abbreviation, (3) Pearson correlation with student success, (4) Feature rank based on correlation with strong performance, (5) Feature rank based on correlation with weak performance.

Category	**Abbreviation**	**Correl**	**Rank**	**Reverse Rank**
Categories most correlated with student performance				
Cognitive processes - discrepancy	DISCREP	0.98	1	72
Grammar - articles	ARTICLE	0.98	2	71
Grammar - numbers	NUMBER	0.96	3	70
Social processes - overall	SOCIAL	0.95	4	69
Cognitive processes - tentative	TENTAT	0.94	5	68
Social processes - family	FAMILY	0.91	6	67
Relativity - motion	MOTION	0.91	7	66
Personal concerns - leisure	LEISURE	0.85	8	65
Grammar - common adjectives	ADJ	0.8	9	64
Grammar - personal pronoun 3rd pers. sg.	SHEHE	0.78	10	63
Cognitive processes - certainty	CERTAIN	0.76	11	62
Grammar - auxiliary verbs	AUXVERB	0.75	12	61
Time orientation - future focus	FOCUSFUTURE	0.74	13	60
Cognitive processes - overall	COGPROC	0.71	14	59
Perceptual processes - see	SEE	0.7	15	58
Categories that are not associated with student performance				
Affective processes - anxiety	ANX	0.08	36	37
Informal language - assent	ASSENT	0.04	37	36
Relativity - time	TIME	0.02	38	35
Personal concerns - home	HOME	-0.03	39	34
Informal language - overall	INFORMAL	-0.07	40	33
Categories that are inversely correlated with student performance				
Grammar - personal pronouns	PPRON	-0.73	58	15
Affective processes - negative emotion	NEGEMO	-0.76	59	14
Grammar - interrogatives	INTERROG	-0.78	60	13
Informal language - swear words	SWEAR	-0.78	61	12
Drives - power	POWER	-0.79	62	11
Grammar - common adverbs	ADVERB	-0.83	63	10
Drives - risk	RISK	-0.86	64	9
Affective processes - overall	AFFECT	-0.88	65	8
Time orientation - past focus	FOCUSPAST	-0.88	66	7
Grammar - common verbs	VERB	-0.89	67	6
Affective processes - sadness	SAD	-0.89	68	5
Grammar - personal pronoun 1st pers. sg.	I	-0.92	69	4
Drives - reward	REWARD	-0.99	70	3
Social processes - friends	FRIEND	-0.99	71	2
Drives - achievement	ACHIEV	-1.00	72	1

6.3.5.2 Psycho-linguistic features

As indicated in Section 6.3.1, LIWC features indicate the psychological profile of a person based on his/her language use. In this section, we analyze the LIWC categories of students as revealed in terms of their Piazza discussions.

Table 6.8 focuses on the psycho-linguistic characteristics associated with student performance. The sample words we cite for a given category are based on the examples from the LIWC 2015 manual [151]; we will use the following format: LIWC category (*ABBREVIATION*: sample words). We note that 6 out of 7 cognitive processes are highly correlated with performance: discrepancy (*DISCREP*: should, would, could) exhibits a 0.98 correlation, tentative (*TENTAT*: maybe, perhaps, guess), 0.94 correlation, certainty (*CERTAIN*: always, never), 0.76 correlation, cognitive (*COGPROC*: cause, know, ought), 0.71 correlation, and differentiation (*DIFFER*: hasn't, but, else), 0.61 correlation. The only cognitive process that exhibits a low correlation is insight (*INSIGHT*: think, know), at 0.17. Some grammar-related categories are also strongly correlated with performance, among them the usage of articles (*ARTICLE*: a, an, the) at 0.98, numbers (*NUMBER*: second, thousand) at 0.96, common adjectives (*ADJ*: free, happy, long) at 0.8, third person singular personal pronouns (*SHEHE*: she, her, him) at 0.78, and auxiliary verbs (*AUXVERB*: am, will, have) at 0.75, while other grammar categories are inversely correlated with performance, among them, the overall usage of personal pronouns (*PPRON*: I, them, her) at -0.73, interrogatives (*INTERROG*: how, when, what) at -0.78, common adverbs (*ADVERB*: very, really) at -0.83, common verbs (*VERB*: eat, come, carry) at -0.89, and above all, the usage of personal pronouns in the first person singular (*I*: I, me, mine) at -0.92. Overall social processes (*SOCIAL*: signaled by words such as mate, talk, they) and family-focused social processes (*FAMILY*: marked by words such as daughter, father) are very strong markers of performance, with a Pearson correlation above 0.91, while friends-focused social processes (*FRIEND*: triggered by words such as buddy, neighbor) are the least associated with performance at -0.99 correlation. Another interesting aspect is that the time orientation of the student's writings are highly correlated with performance, namely top performers are future focused (*FOCUSFUTURE*: may, will, soon), with a correlation of 0.74, while poor performers are past focused (*FOCUSPAST*: ago, did, talked), with a correlation of 0.88. The relativity - motion category (*MOTION*), encompassing words such as arrive, car, go, is highly correlated with performance at 0.91, while the relativity - time category (*TIME*: exemplified by words such as end, until, season) shows no correlation with performance, at 0.02. Among perceptual processes, see (*SEE*: view, saw) is highly correlated with performance, at 0.7. We note that while informal language - overall shows no correlation with performance, netspeak (*NETSPEAK*: btw, lol, thx) is weakly correlated with poor per-

formance, at 0.16, while swear words (*SWEAR*: fuck, damn, shit) are highly correlated with poor performance (at 0.78). Another particularly interesting aspect we should underline, is that most of the personal drives (i.e. 4 out of 5, with the exception of affiliation), namely power (*POWER*: superior, bully), risk (*RISK*: danger, doubt), reward (*REWARD*: take, prize, benefit) and achieve (*ACHIEV*: win, success, better), are highly correlated with the lack of performance in the classroom, at 0.79, 0.86, 0.99, and 0.99 respectively; affiliation (*AFFILIATION*: ally, friend, social) is the only drive that exhibits a strong correlation with performance at 0.5. Affective processes overall (*AFFECT*: happy, cried) are also strongly correlated with poor performance (at 0.88), with subordinate affective processes such as negative emotion (*NEGEMO*: hurt, ugly, nasty) and sadness (*SAD*: crying, grief), displaying a correlation of 0.76 and 0.89, respectively. Interestingly enough, anger (*ANGER*: hate, kill, annoyed) is very weakly associated with poor performance, at 0.11.

6.3.5.3 Stopwords

Table 6.9 lists stopword usage differences and their correlation with classroom performance. We note that the conjunctions *since, if* are the most correlated stopwords with performance, exhibiting the maximum Pearson correlation of 1, while conjunctions *but, whether* are in the 20th most correlated with poor grades, at 0.99 and 0.86, respectively. Determiners such as *a, the, other* as well as pronouns such as *he, itself* are also most correlated with performance, displaying correlations above 0.94; poor performance is correlated above 0.89 with determiners *this, my, 's*, and pronouns *I, me, who*. The stopwords that show no correlation with performance (ranging between 0.02 and -0.02) are conjugated forms of verbs *to be*, *to turn* and *to use*, as well as adverbs *now* and *last*, and the preposition *in*. An interesting observation is that the comparative form of adjective *high* (i.e. *higher*) is strongly correlated with performance, at 0.97, while the superlative form *highest* shows no correlation (at 0.08).

Similar to the psycho-linguistic features, where we noted a strong correlation between past-focused words (FOCUSPAST) and poor student performance, stopwords exhibit the same trend: in the top 50 most correlated words with performance we have no verb conjugated in the past, while in the bottom 50 we have seven verbs that appear in the past form, with correlations above 0.7, among them, *to work*, *to go*, *to be*, and *to make*. Also, similar to the personal pronouns that were correlated with high performance / poor performance, for stopwords we see the same trend at an individual level, namely third person pronouns such as *itself, he, other* are associated with high performance (correlation higher than 0.94), while first person pronouns usage (*I, me, my*) is associated with low performance (correlation above 0.89).

Table 6.9: Stopwords and their Pearson correlation with course performance.

Stopword	Correl	Rank	Reverse Rank
Words most correlated with student performance			
since	1.00	1	384
if	1.00	2	383
orders	0.99	3	382
a	0.98	4	381
way	0.98	5	380
member	0.98	6	379
the	0.98	7	378
number	0.98	8	377
higher	0.97	9	376
room	0.97	10	375
numbers	0.97	11	374
itself	0.96	12	373
possible	0.96	13	372
least	0.96	14	371
has	0.95	15	370
order	0.95	16	369
he	0.95	17	368
across	0.95	18	366
among	0.95	19	367
other	0.94	20	365
Words that are not associated with student performance			
uses	0.02	223	162
now	0.01	224	161
turned	-0.02	228	157
last	-0.02	225	160
been	-0.02	227	158
in	-0.02	226	159
Words that are inversely correlated with student performance			
whether	-0.86	365	20
work	-0.86	366	19
ends	-0.86	367	18
while	-0.87	368	17
let	-0.88	369	16
large	-0.88	370	15
me	-0.89	371	14
i	-0.89	372	13
know	-0.89	373	12
who	-0.90	374	11
get	-0.90	375	10
fact	-0.90	376	9
s	-0.91	377	8
far	-0.92	378	7
my	-0.95	379	6
went	-0.98	380	5
but	-0.99	381	4
worked	-0.99	382	3
this	-0.99	383	2
anyone	-1.00	384	1

6.4 Conclusion

In this chapter, we looked into the prediction of student performance in a large undergraduate course consisting of 600 students. We obtained the text data students wrote on Piazza forum discussions, and used lexical, syntactic, semantic, embedding, and metadata features to predict their final grade in the class. Among text features, word bigrams were found to be the best predictor of student performance, outperforming unigrams and keywords. Embeddings trained on the Piazza data performed comparably well (40.18% accuracy), almost 9 percentage points better than the majority baseline (31.25%). Among non-text features, a small number of time features performed surprisingly well (46.43%), more than 15 points better than the majority baseline. Among hybrid features, the combination of Roget and time features yielded the best results. We further analyzed the correlation between keywords, stopwords, and LIWC (Linguistic Inquiry and Word Count) categories obtained from student text, and found that there was a systematic difference between categories that were associated with high performers, with low performers, and had no association with performance, respectively.

Future research directions would include collecting more student data from different courses, and other important tasks such as prediction of student anxiety and how it affects their performance.

CHAPTER 7

Conclusion

In Chapter 1 we introduced three questions related to Keyword Extraction (KE). In this chapter, we will revisit each of those questions, and outline our contributions, limitations, and potential future work.

1. **Can effective supervised and unsupervised methods be designed for extracting keywords from textual data, and can the core keyword extraction method be specialized to enable the extraction of important content related to undergraduate students, graduate applicants, and online consumers of products?** We showed in Chapter 3 that effective and efficient supervised and unsupervised KE systems can be built on email data. We further showed that the system, when trained on academic papers, can extract meaningful and relevant keywords from student discussions (Chapter 6). We further showed that Keyword Extraction can be specialized for sentences, and that the method when invested with relevant features, can be used to identify usage expression sentences in consumer product reviews (Chapter 4). The method when constrained using Wikipedia,[1] can also be used to uncover important scientific topics in student statements of purpose (Chapter 5), and be used to predict the performance of students in a large undergraduate course while retaining pertinent concepts taught and discussed in the class (Chapter 6). The limitation of the system is the paucity of data in each task, and the necessity of extensive manual annotation. We, however, are of the opinion (based on evidence presented in Chapters 5 and 6) that the system itself can be used to generate new labeled data, essentially self-training or bootstrapping in a new domain and/or task.

2. **Can the KE methods we designed be used to enable different NLP applications?** We showed that the KE system can be effectively used for (1) extraction of salient content from emails (Chapter 3); (2) identification of important specialized content

[1] https://www.wikipedia.org/.

from product reviews (Chapter 4); (3) extraction of important content from student applications and faculty papers for reliable student-faculty matching (Chapter 5); and (4) extraction of relevant content from student forum discussions for the purpose of predicting academic performance (Chapter 6). Chapter 6 further shows that there are three different types of keywords when it comes to student performance prediction – keywords that are associated with high performance, those that are associated with low performance, and the ones that are not associated with performance. These observations form the basis of a potential system that is both efficient and effective, and that can understand student performance early in the course with a view to designing appropriate intervention strategies.

3. **Can automatically extracted keywords be used to gain deeper insights into different data collections?** We used the keywords produced to understand the effect of in-domain training (Chapters 3 and 4); to understand keyphrase appropriateness and time taken to manually classify documents (Chapter 3) – where time taken reduced drastically at a cost of slightly lower accuracy; to look into the temporal flow of scientific topics (Chapter 5) – where we identified a clear content gap between faculty and graduate applicants; to measure the research diversity, research focus, and content density of faculty members (Chapter 5) yielding very different rankings of faculty; and to discern what topics successful students in a large undergraduate class write about (Chapter 6). We note that high-performing students talk about *instructor*, *office hours*, and *professor* often, and focus on testing the code that is already written; low-performing students grapple with fundamental programming issues related to data structures, coding, resource-based problems, and getting the code to compile.

A future research direction is student *anxiety prediction* from text data. While a reasonable body of literature exists for the prediction of student *stress* [152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163], anxiety has not received equal attention. Other future research directions include forecasting student performance for each individual assignment, and the prediction of student performance for next-term courses. While these are more difficult problems to tackle, we hope to be able to solve them with larger and more nuanced datasets. For example, forecasting student performance for individual assignments would require the assignment due date, assignment grade received by the student, and text data authored by the student within a specific time window (assignment issue date until submission date); whereas prediction of student performance for next-term courses would require students who enrolled in at least two different courses in two successive semesters, their corresponding grades, and the text data authored by those students.

Note that the application of keywords in social media data has several possible future research directions, including but not limited to the populations we examined in this dissertation (graduate applicants, undergraduate students, email authors, and online consumers of products). Keyword Extraction (KE) can be used to identify brand information from social tags [164], to detect events and user interests in data streams [165], for designing product ideas [166], and for reasoning about human emotion [167].

In general, keywords express the important *ideas* and *concepts* present within a document. It is therefore of great interest to be able to generalize Keyword Extraction to take into account the philosophical and epistemological implications of a given text document. In other words: we should be able to answer questions such as:

- What are the main "ideas" present in this document?
- What are the main claims the author(s) draw?
- What are the conclusions, express and implied?
- What are the methods/systems used, to reach those conclusions?
- What, if any, are the limitations of the approach?
- How can those ideas be improved to take into account new, and/or potentially challenging, domains?

Note that each of the above questions can, in theory, be expressed in terms of a Keyword Extraction problem. There are, however, two principal roadblocks that need to be removed for answering the above questions:

1. Diverse data. For a proper exposition of keyword extraction in terms of ideas and facets, data from diverse genres and populations must be gathered, with keywords annotated. This is an expensive and time-consuming effort, but definitely worth it.
2. Diverse annotations. Note that the keywords themselves are only as useful as the annotations. It is therefore of special importance to think about each research question separately, and design the annotation guidelines accordingly. It is important to realize that while diverse data presents one side of the keyword extraction problem, diverse annotation presents another.

An interesting and challenging aspect of Keyword Extraction – in previous studies (cf. Chapter 2), in the present work (Chapters 3 to 6), and in future directions – is feature

engineering. Extensive feature engineering and manual design are often necessary to bring out the best performance in Keyword Extraction [10, 107, 108, 168]. A question arises: since deep learning and neural networks have arisen in recent years that largely obviate the need for manual feature design and engineering, and esp. when such techniques were among the best in a recent Keyword Extraction task [40], what is the need for extensive feature engineering? Why not use neural networks instead?

To this question we respond: manual feature design and engineering are often necessary, (a) to get a "feel" for the task; we often have no clue when we are presented with new data, and/or domains, and/or genres; feature engineering gives us a handle to deal with the inherent uncertainty associated with any dataset. (b) Data scarcity. Most datasets in keyword extraction, esp. in new domains, are small. They are either not large enough, not good enough, or not annotated enough. From [17] we know that large data in Keyword Extraction is not sufficient; we also need many annotators to get a reasonable idea of what is happening under the hood. This greatly complicates the problem. A "kitchen sink" application of neural networks fails to solve this problem. (c) Complementarity. Note that the hand-designed features, and features obtained from neural networks are not necessarily antagonistic. They can be complementary. An example of this is shown in Table 6.6, where we combined the features generated using a shallow neural network (Word2Vec), with manually designed features. Such combinations may offer better performance in new domains, esp. when the dataset is relatively small. (d) Control and interpretability. It is an open secret now that neural network models work almost like a "black box", whose features are virtually opaque to the experimenter. This results in a re-incarnation of the interpretability issue that was so rife with topic models [35]. Manually designed features, on the other hand, provide a degree of certainty, control, transparency, and interpretability that will be indeed very hard to obtain otherwise.

BIBLIOGRAPHY

[1] Fleiss, J. L., "Measuring Nominal Scale Agreement Among Many Raters," *Psychological Bulletin*, Vol. 76, No. 5, 1971, pp. 378–382.

[2] Grineva, M., Grinev, M., and Lizorkin, D., "Extracting Key Terms From Noisy and Multi-theme Documents," *18th International World Wide Web Conference*, April 2009, pp. 661–670.

[3] Tonella, P., Ricca, F., Pianta, E., and Girardi, C., "Using Keyword Extraction for Web Site Clustering," *Web Site Evolution, 2003. Theme: Architecture. Proceedings. Fifth IEEE International Workshop on*, 2003, pp. 41–48.

[4] Rennie, J. D. M. and Jaakkola, T., "Using Term Informativeness for Named Entity Detection," *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, 2005, pp. 353–360.

[5] Ferrara, F., Pudota, N., and Tasso, C., "A Keyphrase-Based Paper Recommender System," *Digital Libraries and Archives*, edited by M. Agosti, F. Esposito, C. Meghini, and N. Orio, Vol. 249 of *Communications in Computer and Information Science*, Springer Berlin Heidelberg, 2011, pp. 14–25.

[6] Hulth, A., "Improved Automatic Keyword Extraction Given More Linguistic Knowledge," *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, Association for Computational Linguistics, Stroudsburg, PA, USA, 2003, pp. 216–223.

[7] Mihalcea, R. and Tarau, P., "TextRank: Bringing Order into Texts," *Proceedings of EMNLP 2004*, edited by D. Lin and D. Wu, Association for Computational Linguistics, Barcelona, Spain, July 2004, pp. 404–411.

[8] Nguyen, T. D. and Kan, M.-Y., "Keyphrase Extraction in Scientific Publications," *Proceedings of the 10th international conference on Asian digital libraries: looking back 10 years and forging new frontiers*, ICADL'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 317–326.

[9] Hasan, K. S. and Ng, V., "Conundrums in Unsupervised Keyphrase Extraction: Making Sense of the State-of-the-Art," *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, Association for Computational Linguistics, 2010, pp. 365–373.

[10] Kim, S. N., Medelyan, O., Kan, M.-Y., and Baldwin, T., "SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles," *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 21–26.

[11] Chen, N.-C., Brooks, M., Kocielnik, R., Hong, S. R., Smith, J., Lin, S., Qu, Z., and Aragon, C., *SparQs: Visual Analytics for Sparking Creativity in Social Media Exploration*, Springer International Publishing, Cham, 2017, pp. 394–405.

[12] Wan, X. and Xiao, J., "Single Document Keyphrase Extraction Using Neighborhood Knowledge," *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, AAAI Press, 2008, pp. 855–860.

[13] Litvak, M. and Last, M., "Graph-Based Keyword Extraction for Single-Document Summarization," *Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*, MMIES '08, Association for Computational Linguistics, Stroudsburg, PA, USA, 2008, pp. 17–24.

[14] Seidman, S. B., "Network structure and minimum degree," *Social Networks*, Vol. 5, No. 3, 1983, pp. 269–287.

[15] Batagelj, V. and Zaversnik, M., "An O(m) Algorithm for Cores Decomposition of Networks," *CoRR*, Vol. cs.DS/0310049, 2003.

[16] Tausczik, Y. R. and Pennebaker, J. W., "The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods," *Journal of Language and Social Psychology*, 2010.

[17] Chuang, J., Manning, C. D., and Heer, J., ""Without the Clutter of Unimportant Words": Descriptive Keyphrases for Text Visualization," *ACM Trans. Comput.-Hum. Interact.*, Vol. 19, No. 3, Oct. 2012, pp. 19:1–19:29.

[18] Csomai, A. and Mihalcea, R., "Investigations in Unsupervised Back-of-the-Book Indexing," *FLAIRS Conference*, 2007, pp. 211–216.

[19] Jiang, X., Hu, Y., and Li, H., "A Ranking Approach to Keyphrase Extraction," *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, ACM, 2009, pp. 756–757.

[20] Li, Z., Zhou, D., Juan, Y.-F., and Han, J., "Keyword Extraction for Social Snippets," *Proceedings of the 19th international conference on World wide web*, WWW '10, ACM, New York, NY, USA, 2010, pp. 1143–1144.

[21] Tomokiyo, T. and Hurst, M., "A Language Model Approach to Keyphrase Extraction," *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18*, Association for Computational Linguistics, 2003, pp. 33–40.

[22] Mihalcea, R. and Csomai, A., "Wikify!: Linking Documents to Encyclopedic Knowledge," *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, ACM, New York, NY, USA, 2007, pp. 233–242.

[23] Page, L., Brin, S., Motwani, R., and Winograd, T., "The PageRank Citation Ranking: Bringing Order to the Web," *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, 1998, pp. 161–172.

[24] Kleinberg, J. M., "Authoritative Sources in a Hyperlinked Environment," *J. ACM*, Vol. 46, No. 5, Sept. 1999, pp. 604–632.

[25] Boudin, F., "A Comparison of Centrality Measures for Graph-Based Keyphrase Extraction," *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Asian Federation of Natural Language Processing, Nagoya, Japan, October 2013, pp. 834–838.

[26] Liu, F., Pennell, D., Liu, F., and Liu, Y., "Unsupervised Approaches for Automatic Keyword Extraction Using Meeting Transcripts," *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 620–628.

[27] Wang, R., Liu, W., and McDonald, C., "Corpus-independent generic keyphrase extraction using word embedding vectors," *Software Engineering Research Conference*, Vol. 39, 2014.

[28] Bougouin, A., Boudin, F., and Daille, B., "Keyphrase Annotation with Graph Co-Ranking," *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2945–2955.

[29] Florescu, C. and Caragea, C., "A Position-Biased PageRank Algorithm for Keyphrase Extraction," *AAAI*, 2017, pp. 4923–4924.

[30] Li, C., Liu, Y., and Zhao, L., "Using External Resources and Joint Learning for Bigram Weighting in ILP-Based Multi-Document Summarization," *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Denver, Colorado, May–June 2015, pp. 778–787.

[31] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016.

[32] Aquino, G. O. and Lanzarini, L. C., "Keyword Identification in Spanish Documents using Neural Networks," *Journal of Computer Science & Technology*, Vol. 15, 2015, pp. 55–60.

[33] Wang, Y. and Zhang, J., "Keyword extraction from online product reviews based on bi-directional LSTM recurrent neural network," *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Dec 2017, pp. 2241–2245.

[34] Horita, K., Kimura, F., and Maeda, A., "Automatic Keyword Extraction for Wikification of East Asian Language Documents," *International Journal of Computer Theory and Engineering*, Vol. 8, No. 1, 2016, pp. 32–35.

[35] Lauscher, A., Nanni, F., Fabo, P. R., and Ponzetto, S. P., "Entities as topic labels : combining entity linking and labeled LDA to improve topic interpretability and evaluability," *IJCol - Italian journal of computational linguistics*, Vol. 2, No. 2, 2016, pp. 67–88, Online-Ressource.

[36] Paramonov, I., Lagutina, K., Mamedov, E., and Lagutina, N., "Thesaurus-Based Method of Increasing Text-via-Keyphrase Graph Connectivity During Keyphrase Extraction for e-Tourism Applications," *Knowledge Engineering and Semantic Web*, edited by A.-C. Ngonga Ngomo and P. Křemen, Springer International Publishing, 2016, pp. 129–141.

[37] Chen, J., Zheng, J., and Yu, H., "Finding Important Terms for Patients in Their Electronic Health Records: A Learning-to-Rank Approach Using Expert Annotations," *JMIR Medical Informatics*, Vol. 4, No. 4, 2016.

[38] Siddiqui, T., Ren, X., Parameswaran, A., and Han, J., "FacetGist: Collective Extraction of Document Facets in Large Technical Corpora," *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, ACM, New York, NY, USA, 2016, pp. 871–880.

[39] Chen, N., Hoi, S. C., Li, S., and Xiao, X., "Mobile App Tagging," *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, ACM, New York, NY, USA, 2016, pp. 63–72.

[40] Augenstein, I., Das, M., Riedel, S., Vikraman, L., and McCallum, A., "SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications," *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Association for Computational Linguistics, Vancouver, Canada, August 2017, pp. 546–555.

[41] Turney, P. D., "Learning Algorithms for Keyphrase Extraction," *Information Retrieval*, Vol. 2, No. 4, May 2000, pp. 303–336.

[42] Goodman, J. and Carvalho, V. R., "Implicit Queries for Email," *CEAS*, 2005.

[43] Dredze, M., Wallach, H. M., Puller, D., and Pereira, F., "Generating Summary Keywords for Emails Using Topics," *Proceedings of the 13th International Conference on Intelligent User Interfaces*, IUI '08, ACM, New York, NY, USA, 2008, pp. 199–206.

[44] Laclavík, M. and Maynard, D., "Motivating Intelligent E-mail in Business: An Investigation into Current Trends for E-mail Processing and Communication Research," *2009 IEEE Conference on Commerce and Enterprise Computing*, July 2009, pp. 476–482.

[45] Kassarjian, H. H., "Personality and Consumer Behavior: A Review," *Journal of marketing Research*, 1971, pp. 409–418.

[46] Robertson, T. S. and Myers, J. H., "Personality Correlates of Opinion Leadership and Innovative Buying Behavior," *Journal of Marketing Research*, 1969, pp. 164–168.

[47] Tucker, W. T. and Painter, J. J., "Personality and product use," *Journal of Applied Psychology*, Vol. 45, No. 5, 1961, pp. 325.

[48] Sparks, D. L. and Tucker, W. T., "A Multivariate Analysis of Personality and Product Use," *Journal of Marketing Research*, Vol. 8, No. 1, 1971, pp. 67–70.

[49] Dolich, I. J., "Congruence Relationships Between Self Images and Product Brands," *Journal of Marketing Research*, Vol. 6, No. 1, 1969, pp. 80–84.

[50] Govers, P. C. M. and Schoormans, J. P. L., "Product personality and its influence on consumer preference," *Journal of Consumer Marketing*, Vol. 22, No. 4, 2005, pp. 189–197.

[51] Dave, K., Lawrence, S., and Pennock, D. M., "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews," *Proceedings of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, May 20-24, 2003*, 2003, pp. 519–528.

[52] Hu, M. and Liu, B., "Mining Opinion Features in Customer Reviews," *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*, 2004, pp. 755–760.

[53] Popescu, A.-M. and Etzioni, O., "Extracting Product Features and Opinions from Reviews," *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA, 2005, pp. 339–346.

[54] Ding, X., Liu, B., and Yu, P. S., "A Holistic Lexicon-Based Approach to Opinion Mining," *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, ACM, New York, NY, USA, 2008, pp. 231–240.

[55] Wu, Y., Zhang, Q., Huang, X., and Wu, L., "Phrase Dependency Parsing for Opinion Mining," *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Singapore, August 2009, pp. 1533–1541.

[56] Chao, Y., Wang, Z., Mihalcea, R., and Deng, J., "Mining Semantic Affordances of Visual Object Categories," *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 4259–4267.

[57] Resnik, P., "Selectional Preference and Sense Disambiguation," *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How*, Washington, DC, 1997, pp. 52–57.

[58] Brockmann, C. and Lapata, M., "Evaluating and Combining Approaches to Selectional Preference Acquisition," *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 1*, EACL '03, Association for Computational Linguistics, Stroudsburg, PA, USA, 2003, pp. 27–34.

[59] Erk, K., "A Simple, Similarity-based Model for Selectional Preferences," *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Association for Computational Linguistics, Prague, Czech Republic, June 2007, pp. 216–223.

[60] Pantel, P., Bhagat, R., Coppola, B., Chklovski, T., and Hovy, E., "ISP: Learning Inferential Selectional Preferences," *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, Association for Computational Linguistics, Rochester, New York, April 2007, pp. 564–571.

[61] Bergsma, S., Lin, D., and Goebel, R., "Discriminative Learning of Selectional Preference from Unlabeled Text," *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Honolulu, Hawaii, October 2008, pp. 59–68.

[62] Van de Cruys, T., "A Neural Network Approach to Selectional Preference Acquisition," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, October 2014, pp. 26–35.

[63] Chambers, N. and Jurafsky, D., "Template-Based Information Extraction without the Templates," *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Portland, Oregon, USA, June 2011, pp. 976–986.

[64] Cheung, J. C. K., Poon, H., and Vanderwende, L., "Probabilistic Frame Induction," *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Atlanta, Georgia, June 2013, pp. 837–846.

[65] Chen, Y., Wang, W. Y., and Rudnicky, A. I., "Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing," *2013*

IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013, 2013, pp. 120–125.

[66] Gupta, S. and Manning, C., "Analyzing the Dynamics of Research by Extracting Key Aspects of Scientific Papers," *Proceedings of 5th International Joint Conference on Natural Language Processing*, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, November 2011, pp. 1–9.

[67] Tsai, C.-T., Kundu, G., and Roth, D., "Concept-Based Analysis of Scientific Literature," *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, CIKM '13, ACM, New York, NY, USA, 2013, pp. 1733–1738.

[68] Juola, P., "Authorship Attribution," *Found. Trends Inf. Retr.*, Vol. 1, No. 3, Dec. 2006, pp. 233–334.

[69] Stamatatos, E., "A survey of modern authorship attribution methods," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 60, No. 3, March 2009, pp. 538–556.

[70] Koppel, M., Schler, J., and Argamon, S., "Computational methods in authorship attribution," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 60, No. 1, Jan. 2009, pp. 9–26.

[71] Rosen-Zvi, M., Griffiths, T., Steyvers, M., and Smyth, P., "The Author-topic Model for Authors and Documents," *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, UAI '04, AUAI Press, Arlington, Virginia, United States, 2004, pp. 487–494.

[72] Singh, A., Rose, C., Visweswariah, K., Chenthamarakshan, V., and Kambhatla, N., "PROSPECT: A System for Screening Candidates for Recruitment," *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, ACM, New York, NY, USA, 2010, pp. 659–668.

[73] Li, H., "A Short Introduction to Learning to Rank," *IEICE Transactions*, Vol. 94-D, No. 10, 2011, pp. 1854–1862.

[74] Elbadrawy, A., Polyzou, A., Ren, Z., Sweeney, M., Karypis, G., and Rangwala, H., "Predicting Student Performance Using Personalized Analytics," *Computer*, Vol. 49, No. 4, Apr 2016, pp. 61–69.

[75] Sweeney, M., Lester, J., Rangwala, H., and Johri, A., "Next-Term Student Performance Prediction: A Recommender Systems Approach," *EDM*, 2016.

[76] Mueen, A., Zafar, B., and Manzoor, U., "Modeling and Predicting Students' Academic Performance Using Data Mining Techniques," *International Journal of Modern Education and Computer Science*, Vol. 11, 11 2016, pp. 36–42.

[77] Qiu, J., Tang, J., Liu, T. X., Gong, J., Zhang, C., Zhang, Q., and Xue, Y., "Modeling and Predicting Learning Behavior in MOOCs," *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, ACM, New York, NY, USA, 2016, pp. 93–102.

[78] Xu, B. and Yang, D., "Motivation Classification and Grade Prediction for MOOCs Learners," *Intell. Neuroscience*, Vol. 2016, Jan. 2016, pp. 4:4–4:4.

[79] Meier, Y., Xu, J., Atan, O., and van der Schaar, M., "Predicting Grades," *IEEE Transactions on Signal Processing*, Vol. 64, No. 4, Feb 2016, pp. 959–972.

[80] Widyahastuti, F., Riady, Y., and Zhou, W., "Prediction Model Students' Performance in Online Discussion Forum," *Proceedings of the 5th International Conference on Information and Education Technology*, ICIET '17, ACM, New York, NY, USA, 2017, pp. 6–10.

[81] Cen, L., Ruta, D., Powell, L., Hirsch, B., and Ng, J., "Quantitative approach to collaborative learning: performance prediction, individual assessment, and group composition," *International Journal of Computer-Supported Collaborative Learning*, Vol. 11, No. 2, Jun 2016, pp. 187–225.

[82] Sander, P., "Using Learning Analytics to Predict Academic Outcomes of First-year Students in Higher Education," Tech. rep., University of Oregon, 2016.

[83] Romero, C. and Ventura, S., "Educational data science in massive open online courses," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 7, No. 1, 2017, pp. n/a–n/a.

[84] Rendle, S., "Factorization Machines with libFM," *ACM Trans. Intell. Syst. Technol.*, Vol. 3, No. 3, May 2012, pp. 57:1–57:22.

[85] Elbadrawy, A., Studham, R. S., and Karypis, G., "Personalized Multi-Regression Models for Predicting Students' Performance in Course Activities," *Proceedings of the 5th International Learning Analytics & Knowledge conference*, LAK '15, 2015.

[86] Dillenbourg, P., *Collaborative Learning: Cognitive and Computational Approaches*, Advances in learning and instruction series, Elsevier Science & Technology Books, 1999.

[87] Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K., "Extreme learning machine: Theory and applications," *Neurocomputing*, Vol. 70, No. 1, 2006, pp. 489 – 501, Neural Networks.

[88] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J., *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA, 1984.

[89] MITx and HarvardX, "HarvardX-MITx Person-Course Academic Year 2013 De-Identified dataset, version 2.0," 2014.

[90] Cristianini, N. and Shawe-Taylor, J., *An Introduction to Support Vector Machines And Other Kernel-based Learning Methods*, Cambridge University Press, New York, NY, USA, 2000.

[91] Hosmer, D. W. and Lemeshow, S., *Applied Logistic Regression*, John Wiley and Sons, 2000.

[92] Chen, Y., Yu, B., Zhang, X., and Yu, Y., "Topic Modeling for Evaluating Students' Reflective Writing: A Case Study of Pre-service Teachers' Journals," *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, LAK '16, ACM, New York, NY, USA, 2016, pp. 1–5.

[93] Shermis, M. D. and Burstein, J. C., editors, *Automated Essay Scoring: A Cross Disciplinary Perspective*, Lawrence Erlbaum Associates, Hillsdale, NJ, 2003.

[94] Blei, D. M., Ng, A. Y., and Jordan, M. I., "Latent Dirichlet Allocation," *J. Mach. Learn. Res.*, Vol. 3, March 2003, pp. 993–1022.

[95] Liu, Z., Huang, W., Zheng, Y., and Sun, M., "Automatic Keyphrase Extraction via Topic Decomposition," *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 366–376.

[96] Lee, S. and Kim, H.-j., "News Keyword Extraction for Topic Tracking," *Proceedings of the 2008 Fourth International Conference on Networked Computing and Advanced Information Management - Volume 02*, NCM '08, IEEE Computer Society, Washington, DC, USA, 2008, pp. 554–559.

[97] Yih, W.-t., Goodman, J., and Carvalho, V. R., "Finding Advertising Keywords on Web Pages," *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, ACM, New York, NY, USA, 2006, pp. 213–222.

[98] Klimt, B. and Yang, Y., "Introducing the Enron Corpus," *First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, 2004.

[99] Clear, J. H., "The British National Corpus," *The digital word*, edited by G. P. Landow and P. Delany, MIT Press, Cambridge, MA, USA, 1993, pp. 163–187.

[100] Lahiri, S., "Complexity of Word Collocation Networks: A Preliminary Structural Analysis," *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, Gothenburg, Sweden, April 2014, pp. 96–105.

[101] Finkel, J. R., Grenager, T., and Manning, C., "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling," *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, Association for Computational Linguistics, Stroudsburg, PA, USA, 2005, pp. 363–370.

[102] Phan, X.-H., "CRFTagger: CRF English POS Tagger," 2006.

[103] Csomai, A. and Mihalcea, R., "Linguistically Motivated Features for Enhanced Back-of-the-Book Indexing," *ACL*, edited by K. McKeown, J. D. Moore, S. Teufel, J. Allan, and S. Furui, The Association for Computer Linguistics, 2008, pp. 932–940.

[104] Loza, V., Lahiri, S., Mihalcea, R., and Lai, P.-H., "Building a Dataset for Summarization and Keyword Extraction from Emails," *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, edited by N. C. C. Chair), K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, European Language Resources Association (ELRA), Reykjavik, Iceland, may 2014.

[105] Hasan, K. S. and Ng, V., "Automatic Keyphrase Extraction: A Survey of the State of the Art," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Baltimore, Maryland, June 2014, pp. 1262–1273.

[106] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H., "The WEKA data mining software: an update," *SIGKDD Explor. Newsl.*, Vol. 11, No. 1, Nov. 2009, pp. 10–18.

[107] Pianta, E. and Tonelli, S., "KX: A Flexible System for Keyphrase eXtraction," *Proceedings of the 5th International Workshop on Semantic Evaluation*, Association for Computational Linguistics, Uppsala, Sweden, July 2010, pp. 170–173.

[108] Berend, G. and Farkas, R., "SZTERGAK : Feature Engineering for Keyphrase Extraction," *Proceedings of the 5th International Workshop on Semantic Evaluation*, Association for Computational Linguistics, Uppsala, Sweden, July 2010, pp. 186–189.

[109] Berend, G., "Opinion Expression Mining by Exploiting Keyphrase Extraction," *Proceedings of 5th International Joint Conference on Natural Language Processing*, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, November 2011, pp. 1162–1170.

[110] Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., and Nevill-Manning, C. G., "KEA: Practical Automatic Keyphrase Extraction," *Proceedings of the fourth ACM conference on Digital libraries*, DL '99, ACM, New York, NY, USA, 1999, pp. 254–255.

[111] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J., "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 3111–3119.

[112] Pagliardini, M., Gupta, P., and Jaggi, M., "Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features," *CoRR*, Vol. abs/1703.02507, 2017.

[113] Boom, C. D., Canneyt, S. V., Demeester, T., and Dhoedt, B., "Representation learning for very short texts using weighted word embedding aggregation," *Pattern Recognition Letters*, Vol. 80, 2016, pp. 150–156.

[114] Arora, S., Liang, Y., and Ma, T., "A simple but tough-to-beat baseline for sentence embeddings," 2016.

[115] Klein, D. and Manning, C. D., "Accurate Unlexicalized Parsing," *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Sapporo, Japan, July 2003, pp. 423–430.

[116] Chen, D. and Manning, C., "A Fast and Accurate Dependency Parser using Neural Networks," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, October 2014, pp. 740–750.

[117] Flesch, R., "A new readability yardstick," *Journal of applied psychology*, Vol. 32, No. 3, 1948, pp. 221.

[118] Farr, J. N., Jenkins, J. J., and Paterson, D. G., "Simplification of Flesch Reading Ease Formula," *Journal of applied psychology*, Vol. 35, No. 5, 1951, pp. 333.

[119] Senter, R. J. and Smith, E. A., "Automated readability index," Tech. rep., DTIC Document, 1967.

[120] Kincaid, J. P., Fishburne Jr., R. P., Rogers, R. L., and Chissom, B. S., "Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel," Tech. rep., DTIC Document, 1975.

[121] Coleman, M. and Liau, T. L., "A computer readability formula designed for machine scoring," *Journal of Applied Psychology*, Vol. 60, No. 2, 1975, pp. 283.

[122] Gunning, R., *The technique of clear writing*, McGraw-Hill New York, 1968.

[123] McLaughlin, G. H., "SMOG grading: A new readability formula," *Journal of reading*, Vol. 12, No. 8, 1969, pp. 639–646.

[124] Heylighen, F. and Dewaele, J.-M., "Formality of Language: definition, measurement and behavioral determinants," *Interner Bericht, Center "Leo Apostel", Vrije Universiteit Brüssel*, 1999.

[125] Ure, J., "Lexical density and register differentiation," *Applications of Linguistics*, 1971, pp. 443–452.

[126] Nelson, D. L., McEvoy, C. L., and Schreiber, T. A., "The University of South Florida word association, rhyme, and word fragment norms." 1998.

[127] Levin, B., *English Verb Classes And Alternations: A Preliminary Investigation*, The University of Chicago Press, 1993.

[128] Strapparava, C. and Valitutti, A., "WordNet-Affect: an Affective Extension of WordNet," *In Proceedings of the 4th International Conference on Language Resources and Evaluation*, 2004, pp. 1083–1086.

[129] Srikumar, V. and Roth, D., "Modeling Semantic Relations Expressed by Prepositions," *Transactions of the Association for Computational Linguistics*, Vol. 1, 2013, pp. 231–242.

[130] Schneider, N., Srikumar, V., Hwang, J. D., and Palmer, M., "A Hierarchy with, of, and for Preposition Supersenses," *Proceedings of The 9th Linguistic Annotation Workshop*, June 2015, pp. 112–123.

[131] Schneider, N., Hwang, J. D., Srikumar, V., Green, M., Suresh, A., Conger, K., O'Gorman, T., and Palmer, M., "A Corpus of Preposition Supersenses," *Proceedings of the 10th Linguistic Annotation Workshop*, Association for Computational Linguistics, Berlin, Germany, Aug. 2016.

[132] Wu, Z. and Palmer, M., "Verb Semantics and Lexical Selection," *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, Association for Computational Linguistics, Stroudsburg, PA, USA, 1994, pp. 133–138.

[133] Buciluǎ, C., Caruana, R., and Niculescu-Mizil, A., "Model Compression," *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, ACM, New York, NY, USA, 2006, pp. 535–541.

[134] Lahiri, S., Mihalcea, R., and Lai, P.-H., "Keyword Extraction from Emails," *Natural Language Engineering*, Vol. 23, No. 2, 2017, pp. 295–317.

[135] Smyth, B. and McClave, P., "Similarity vs. Diversity," *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR '01, Springer-Verlag, London, UK, UK, 2001, pp. 347–361.

[136] Rotabi, R. and Kleinberg, J. M., "The Status Gradient of Trends in Social Media," *Proceedings of the Tenth International Conference on Web and Social Media, Cologne, Germany, May 17-20, 2016.*, 2016, pp. 319–328.

[137] Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D., "The Stanford CoreNLP Natural Language Processing Toolkit," *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Association for Computational Linguistics, Baltimore, Maryland, June 2014, pp. 55–60.

[138] Salton, G. and Buckley, C., "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, Vol. 24, No. 5, 1988, pp. 513–523.

[139] Kumaran, G. and Allan, J., "Text Classification and Named Entities for New Event Detection," *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, ACM, New York, NY, USA, 2004, pp. 297–304.

[140] Lan, M., Tan, C.-L., and Low, H.-B., "Proposing a New Term Weighting Scheme for Text Categorization," *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI '06, AAAI Press, 2006, pp. 763–768.

[141] Ko, Y., "A Study of Term Weighting Schemes Using Class Information for Text Classification," *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, ACM, New York, NY, USA, 2012, pp. 1029–1030.

[142] R., A., R., S., Suresh, V., and Murty, M. N., "Stopwords and Stylometry : A Latent Dirichlet Allocation Approach," *NIPS Workshop on Applications for Topic Models: Text and Beyond*, Whistler, Canada, 2009.

[143] Menon, R. and Choi, Y., "Domain Independent Authorship Attribution without Domain Adaptation," *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, RANLP 2011 Organising Committee, Hissar, Bulgaria, September 2011, pp. 309–315.

[144] Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., and Chanona-Hernández, L., "Syntactic Dependency-Based N-grams as Classification Features," *Advances in Computational Intelligence*, edited by I. Batyrshin and M. G. Mendoza, Springer, Berlin, Heidelberg, 2013, pp. 1–11.

[145] Kaster, A., Siersdorfer, S., and Weikum, G., "Combining text and linguistic document representations for authorship attribution," *SIGIR workshop: stylistic analysis of text for information access*, 2005.

[146] Landauer, T. K. and Dumais, S. T., "Latent semantic analysis," *Scholarpedia*, Vol. 3, No. 11, 2008, pp. 4356.

[147] F.R.S., K. P., "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Vol. 2, No. 11, 1901, pp. 559–572.

[148] Jarmasz, M., "Roget's Thesaurus as a Lexical Resource for Natural Language Processing," *CoRR*, Vol. abs/1204.0140, 2012.

[149] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[150] Rokach, L. and Maimon, O., *Data Mining With Decision Trees: Theory and Applications*, World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2nd ed., 2014.

[151] Pennebaker, J. W., Boyd, R. L., Jordan, K., and Blackburn, K., "The development and psychometric properties of LIWC2015," Tech. rep., The University of Texas at Austin, 2015.

[152] Arip, M. A. S. M., Kamaruzaman, D. N., Roslan, A., Ahmad, A., and Rahman, M. R. A., "Development, Validity and Reliability of Student Stress Inventory (SSI)," *The Social Sciences*, Vol. 10, No. 7, 2015, pp. 1631–1638.

[153] Feldt, R. C., "Development of a Brief Measure of College Stress: The College Student Stress Scale," *Psychological Reports*, Vol. 102, No. 3, 2008, pp. 855–860, PMID: 18763455.

[154] Stallman, H. M. and Hurst, C. P., "The University Stress Scale: Measuring Domains and Extent of Stress in University Students," *Australian Psychologist*, Vol. 51, No. 2, 2016, pp. 128–134.

[155] Dahlin, M., Joneborg, N., and Runeson, B., "Stress and depression among medical students: a cross-sectional study," *Medical Education*, Vol. 39, No. 6, 2005, pp. 594–604.

[156] Darling, C. A., McWey, L. M., Howard, S. N., and Olmstead, S. B., "College student stress: the influence of interpersonal relationships on sense of coherence," *Stress and Health*, Vol. 23, No. 4, 2007, pp. 215–229.

[157] Gadzella, B. and Baloğlu, M., "Confirmatory Factor Analysis and Internal Consistency of the Student-life Stress Inventory," *Journal of Instructional Psychology*, Vol. 28, 01 2001, pp. 84–94.

[158] Barker, S. B., Barker, R. T., McCain, N. L., and Schubert, C. M., "A Randomized Cross-over Exploratory Study of the Effect of Visiting Therapy Dogs on College Student Stress Before Final Exams," *Anthrozoös*, Vol. 29, No. 1, 2016, pp. 35–46.

[159] Torres, J. B. and Solberg, V. S., "Role of Self-Efficacy, Stress, Social Integration, and Family Support in Latino College Student Persistence and Health," *Journal of Vocational Behavior*, Vol. 59, No. 1, 2001, pp. 53 – 63.

[160] Beaumont, E., Durkin, M., Hollins Martin, C. J., and Carson, J., "Measuring relationships between self-compassion, compassion fatigue, burnout and well-being in student counsellors and student cognitive behavioural psychotherapists: a quantitative survey," *Counselling and Psychotherapy Research*, Vol. 16, No. 1, 2016, pp. 15–23.

[161] no Sierra, J. O., Fonseca-Pedrero, E., Aritio-Solana, R., and de Luis, E. C., "Stress assessment during adolescence: Psychometric properties and measurement invariance of the Student Stress Inventory-Stress Manifestations across gender and age," *European Journal of Developmental Psychology*, Vol. 13, No. 5, 2016, pp. 529–544.

[162] Gadzella, B. M., "Student-Life Stress Inventory: Identification of and Reactions to Stressors," *Psychological Reports*, Vol. 74, No. 2, 1994, pp. 395–402, PMID: 8197278.

[163] Kokkinos, C. M., Stavropoulos, G., and Davazoglou, A., "Development of an instrument measuring student teachers' perceived stressors about the practicum," *Teacher Development*, Vol. 20, No. 2, 2016, pp. 275–293.

[164] Nam, H., Joshi, Y. V., and Kannan, P. K., "Harvesting Brand Information from Social Tags," *Journal of Marketing*, Vol. 81, No. 4, 2017, pp. 88–108.

[165] Shi, L. L., Liu, L., Wu, Y., Jiang, L., and Hardy, J., "Event Detection and User Interest Discovering in Social Media Data Streams," *IEEE Access*, Vol. 5, 2017, pp. 20953–20964.

[166] Jeong, B., Yoon, J., and Lee, J.-M., "Social media mining for product planning: A product opportunity mining approach based on topic modeling and sentiment analysis," *International Journal of Information Management*, 2017.

[167] Li, X., Wang, Z., Gao, C., and Shi, L., "Reasoning human emotional responses from large-scale social and public media," *Applied Mathematics and Computation*, Vol. 310, 2017, pp. 182–193.

[168] Lopez, P. and Romary, L., "HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID," *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 248–251.