UNIVERSITY OF MINNESOTA

This is to certify that I have examined this copy of Master's thesis by

Anagha Krishna Kulkarni

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Dr. Ted Pedersen

_____

Name of Faculty Adviser

_____

Signature of Faculty Advisor

_____

Date

GRADUATE SCHOOL

Unsupervised Context Discrimination and Automatic Cluster Stopping

A THESIS

SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL

OF THE UNIVERSITY OF MINNESOTA

BY

Anagha Krishna Kulkarni

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

July 2006

Acknowledgments

I would like to thank my advisor Dr. Ted Pedersen who has among many other things introduced me to the intriguing field of Natural Language Processing, has taught me how to methodically and ethically conduct research and has given me a great role model. I could not have made it this far without his endless patience and guidance.

I am grateful to Dr. Ron Regal and Dr. Hudson Turner for raising interesting issues and questions, for thorough reviews of my thesis report and for kindly accepting to be on my committee on a relatively short notice.

The NLP@UMD's group meetings have re-instilled my belief in the benefits of well-focused discussions and here I would like to express my appreciation towards the group members Mahesh Joshi, Saiyam Kohli, Jason Michelizzi, Apurva Padhye and Prath, for their interesting and encouraging feedbacks and finally to Ted for organizing and facilitating these meetings.

I would like to thank Dean Riehl and Dr. Carolyn Crouch for generously funding my travel to instrumental events like conferences, workshops and tutorials.

This work has been supported by a National Science Foundation Faculty Early CAREER Development Award (#0092784).

This work was carried out in part using hardware and software provided by the University of Minnesota Supercomputing Institute.

This work would have been beyond imagination had it not been for the eternal support and encouragement that I have received from my husband, Mahesh and from my parents, whom I can never thank enough!

|| Om Shri Sairam ||

Contents

List of Figures

List of Tables

Abstract

Context discrimination is the task of separating a set of written contexts into different groups based on their mutual similarity or dissimilarity. More specific instances of this task are problems such as name discrimination, word sense discrimination, and email clustering. Name discrimination seeks to distinguish between different entities that share the same name. For example, George Bush - the 41st President of the USA, versus George Bush - the 43rd. In word sense discrimination the task is to disambiguate the occurrences of a word with multiple meanings. For example, the word *tape* can refer to a measuring device, a fastening piece of cloth or paper, etc. The goal of email clustering is to group together messages that discuss similar topics.

In this thesis we approach each of these problem and more generally the context clustering problem by adapting and extending the unsupervised word sense discrimination techniques proposed by Purandare and Pedersen [2004]. One of our main contributions is the development of three cluster-stopping measures: PK2, PK3 and the Adapted Gap Statistic. Given a collection of contexts it is a non-trivial task for a user to determine how many different clusters might exist. Hence, we automate the process of predicting the optimal number of clusters for any given dataset by considering the properties of the raw data. We find that our measures can be successfully applied to each of the above problems with equal ease. Of the three cluster-stopping measures, PK3 shows the highest agreement when compared to manually determined clusters.

1 Introduction

One of the fortes of the human brain is its ability to group entities based on the features that the brain either recognizes or decides to be of importance. Such grouping or *clustering* of entities helps us in multiple ways:

- it makes large volumes of data digestible
- it allows generalization of properties and rules
- it facilitates deduction of patterns and trends

Because of such properties, automated clustering of data finds applications in many fields such as marketing, finance, census and education.

In this thesis we apply clustering to the domain of natural language processing. We use clustering to find similar contexts in written text where we define a context as any unit of text such as sentence, paragraph or an entire document. Given a set of such contexts we cluster these such that similar contexts are grouped together in the same cluster while dissimilar contexts are separated into different clusters. The applications of this idea are many. For instance, given a set of web-pages one might want to group them based on the similarity of their contents so as to better facilitate the perusal of their contents. Or, one might wish to do the same with a collection of email messages to group together the related threads of correspondences.

Another interesting application of clustering techniques is the task of word sense discrimination where the goal is to distinguish between the usage of different meanings of an ambiguous word in different contexts. For example, given a set of sentences each containing the word *tape* the task would group these sentences such that the ones that use the sense of *cloth or paper used to fasten things* are grouped together while the ones that refer to the *recording device* are grouped together into another cluster and the ones that refer to the *measuring instrument* form the third group of sentences. In short, the goal is to separate out the contexts using different senses of an ambiguous word into different clusters.

A more specific manifestation of this problem that we investigate in this thesis is the task of proper name discrimination. The problem here concerns people, places or organizations that share the same name. For instance, the city name *Duluth* can refer to the city in the state of Minnesota, USA or can also refer to the city in the state of Georgia, USA. Or, the name *George Miller* can refer to the *Professor from the Princeton University - WordNet father* or to the *Australian movie director* or to a *Congressman from the USA*. This problem of name ambiguity is even more severe for the World Wide Web (WWW). As the web presence of people, places, organizations increases, name ambiguity problems become even more acute. For example, a Google search for the name *Michael Collins* provides web-pages related to more than four different people in the first 50 results[1].

While each of the above tasks can be approached in multiple ways, we chose clustering primarily because it is an *unsupervised machine learning* method. A method is said to be *unsupervised* if it does not require compiled knowledge resources or manually annotated/labeled data. Rather, these methods try to extract information exclusively from the given raw data. The advantages of using these type of methods will become evident once we discuss the *supervised* methods. Traditionally, training a machine to perform any human-like activity has involved creating rules and hand-crafting database of sub-activities, cause-effect relationships etc. As one can guess, such systems are regarded as the *supervised* systems. Although these systems provide good accuracy for the tasks they are crafted for, they suffer from the knowledge acquisition bottleneck. For instance, word sense disambiguation is the task of assigning each instance of an ambiguous word with a sense from an existing inventory of senses. Now if a language specific resource such as a sense-inventory or a dictionary is created to assist in this task, the applicability of such resource is limited to the language it has been crafted for and thus cannot be used with a different language. Secondly, it is also require a regular maintenance to keep up with additions of new meanings to words across time. As such, *supervised* systems cannot easily scale to different tasks or domains.

Another great advantage of the unsupervised nature of our approach is its ability to remain language-independent. In our approach we do not make use of any form of external language dependent knowledge resource such as machine readable dictionaries or ontologies, nor do we use language dependent utilities such as part-of-speech taggers or parsers. As a result our method is completely

[1] As on May 5, 2006

language independent and can be applied to any language where the tokenization is based on a separating white space. We have demonstrated (Pedersen et al. [2006b], Pedersen et al. [2006a]) the language independent nature of our method by applying it to the task of name discrimination across several different languages other than English, such as Romanian, Spanish and Bulgarian.

In this thesis we extend and generalize the methods proposed by Purandare and Pedersen [2004] for the task of unsupervised word sense discrimination. They base their approach on the methods proposed by Schütze [1998] and Pedersen and Bruce [1997]. The philosophy underlying all these methods is a hypothesis proposed by Miller and Charles [1991] where they state that *any two words are semantically similar to the extent that their contextual representations are similar*. For instance, all the occurrences of *George Miller* which occur with *Princeton University* or *WordNet* can be expected to refer to the *Professor* whereas the ones co-occurring with *Australia* or *Mad Max* are highly likely to refer to the *Movie Director*.

The following section highlights the contributions of this thesis to the research on name discrimination, email clustering and context discrimination in general.

1.1 Theory

- In this thesis we have proposed and implemented four *cluster stopping* measures that automate the process of choosing the appropriate number of clusters for any given data and thus eliminates the dependency of our approach on users knowledge about the data at hand.
- Based on the agreement shown between the predictions made by a cluster stopping measure and the *given* number of clusters we find that our cluster stopping measures can be ranked in the following order: 1st :- $PK3$, 2nd :- $PK2$ and 3rd :- $Adapted\ Gap\ Statistic$
- As a pre-requisite of the cluster stopping measure Adapted Gap Statistic, we have proposed and implemented an algorithm to generate a random matrix given the column and the row marginals.
- From our experimental results we conclude that if large data has clearly distinct senses to be discriminated then first order context representation is the best option whereas for small data with fine-grained/subtle sense distinctions second order context representation is a better

choice. As a result, we find that first and second order context representation are more of a complimentary pair rather than a competing pair of parameter settings.

- We find that for our approach performing Singular Value Decomposition does not really buy any improvement which is goes against the conventional wisdom and thus is a surprising finding.

- We conclude that often the clustering methods of Repeated Bisection and Agglomerative Clustering are equivalent for our approach. However, with small amounts of data and for subtle ambiguities Repeated Bisection is a better choice.

- We have observed from experimental results that discriminating the subtle senses of ambiguities is a non-trivial task as compared to the task of discriminating relatively distinct senses.

1.2 SenseClusters Related

Pedersen and Purandare had developed a freely available open source package called SenseClusters[2] which implemented their methodology of unsupervised word sense discrimination. We started with version 0.55 of SenseClusters and since then have released several enhanced versions. The current version, which was also used to conduct all the experiments reported in this thesis, is 0.91, released on June 17.

- We have extended the previous methodology and SenseClusters to support *headless* contexts, that is, contexts which do not contain any *head* word to be disambiguated but instead the complete contexts are to be analyzed and clustered. Email clustering is an example of *headless* clustering.

- We have further extended the discrimination capabilities of our methodology by adding support for clustering individual words and as a result finding sets of related words.

- Once the given contexts are clustered we now also assign each discovered cluster a label which serves as an indicator of the underlying entity for the cluster.

[2] http://senseclusters.sourceforge.net

- We have enhanced SenseClusters to be now capable of applying Singular Value Decomposition (SVD) to the first order context representation, which was previously viewed as just for the second order context representation.

1.3 Utilities

- We have crafted a data creation utility to create data for the name discrimination problem. This utility is capable of generating SENSEVAL-2 formatted files containing artificially created name ambiguity from originally unambiguous names.
- We have also created another data creation utility for *headless* datasets which converts contents of each input file into a context to be clustered.
- We have implemented and distributed a solution proposed by Munkres [1957] to the classical assignment problem. We use this in the evaluation phase of our methodology to resolve a significant bottleneck.

2 Background

In this section we provide the background about techniques and measures that are fundamental to our work.

2.1 Clustering

Clustering is one of the primary methods of unsupervised learning. The expectation is to find and learn the patterns present in the given vectors that represent the contexts (context-vectors), based exclusively on the similarity or dissimilarity that a clustering algorithm is able to find amongst these vectors. A comprehensive review of clustering can be found in Jain et al. [1999].

Clustering algorithms are broadly classified as Hierarchical, Partitional and Hybrid.

Hierarchical clustering is further classified into the bottom-up agglomerative and the top-down divisive techniques. Agglomerative clustering starts with singleton clusters where each context-vector is placed in a separate cluster. At every stage two clusters which are most similar to each other are merged together. This can be repeated until all the context-vectors are combined into a single cluster. The divisive clustering works in exactly the opposite direction - from one cluster to multiple cluster, splitting clusters at every stage based on their dissimilarity. Thus clustering of data using Hierarchical techniques proceeds in stages.

Unlike Hierarchical clustering, Partitional techniques cluster the data in just one stage. K-means is one of the widely used Partitional clustering techniques. The clustering mechanism of K-means is based on the centroids/means of the clusters. K-means starts by choosing k random context-vectors as the centroids and then assigns the remaining context-vectors to the nearest centroid. The centroids of clusters thus formed are re-calculated. This process of assigning context-vectors to the nearest centroid is repeated until the centroids do not change and thus k clusters are obtained from the dataset.

Hierarchical techniques have quadratic time complexity whereas the Partitional techniques have linear time complexity. Although Partitional techniques are computationally more efficient than the Hierarchical clustering it has been traditionally believed that Hierarchical techniques produce

better solutions than Partitional. However, Zhao and Karypis [2002] have found that contrary to the common belief about inefficiency of the Partitional algorithms, they out-perform other clustering algorithms for large corpora. They attribute this poor performance of Hierarchical clustering to the merging errors that occur at the early stages of the clustering and get multiplied along the way. The merging errors usually get introduced in the early stages because of the nature of hierarchical clustering which lacks the global view of the data.

Hybrid clustering algorithm are partitional methods that produce hierarchical clustering solutions using repeated bisections. The intention is to take advantage of the global view of the partitional algorithms but also to reduce the instability induced because of the initial k random centroids. Specifically, instead of partitioning the data directly into k partitions, the data is partitioned into 2 (bisected) clusters at each stage. Thus the initial one cluster containing all the context-vectors is partitioned into two clusters. Then the larger of the two is further partitioned into two clusters and this can continue until each of the context-vectors are partitioned into its own separate cluster or $k - 1$ times if k clusters are expected from the data.

2.2 Criterion Functions

Criterion function is a term that is typically used to refer to the different metrics that clustering algorithms use to try and optimize the quality of the clustering solution. The criterion functions that are utilized in this work are described in detail below.

Criterion functions are classified into the following three categories:

1. internal (e.g., $I1$ and $I2$)
2. external (e.g., $E1$)
3. hybrid (e.g., $H1$ and $H2$)

The *internal* type of criterion functions, as the name suggests, take an intra-cluster (within) view of the clustering process and are thus defined only on the context-vectors that are present inside each cluster. In other words, the *internal* criterion functions do not take into account the context-vectors

that have been assigned to other clusters and thus only capture how the context-vectors in any given cluster are related together. The criterion functions under this type that we study are referred as $I1$ and $I2$.

The *external* criterion functions take an inter-cluster (between) view and try to find clusters which are as different or dissimilar from each other as possible. Unlike *internal*, the *external* criterion functions capture how context-vectors in a cluster are related with the complete set of context-vectors. We study one external criterion function referred to as $E1$.

In the formulation of the final type of criterion functions - *hybrid*, both *internal* as well as *external* criterion functions used. As a result, *hybrid* criterion functions strive to propose clustering solution which maximize intra-cluster similarity and simultaneously minimize the inter-cluster similarity. It is interesting to note that if clustering a context-vector in one specific cluster results in improvement in one of the criterion functions but degradation in the other criterion function then whether the context-vector is assigned to that cluster is decided by comparing the improvement to be gained with the degradation to be borne. Two criterion functions, namely $H1$ and $H2$, from this category are described below.

2.2.1 Internal Criterion Function - $I1$

$I1$ is a maximization function defined over the sum of the average pairwise similarities between the context-vectors assigned to each cluster, weighted by the size of each cluster. The formulation of the same is given in Formula 1 where the similarity between context-vectors d_i and d_j is measured using the cosine function and the size of each of the k clusters is represented as n_r.

$$maximize\ I1 = \sum_{r=1}^{k} n_r \left(\frac{1}{n_r^2} \sum_{d_i, d_j \in S_r} cos(d_i, d_j) \right) \tag{1}$$

2.2.2 Internal Criterion Function - $I2$

$I2$ is a maximization function which favors a clustering solution that maximizes the sum of the similarity of each context-vector in a cluster with the cluster centroid/mean. Thus $I2$ is formulated

as shown in Formula 2.

$$maximize\ I2 = \sum_{r=1}^{k} \sum_{d_i \in S_r} cos(d_i, C_r) \tag{2}$$

2.2.3 External Criterion Function - $E1$

$E1$ is a minimization function as shown in Formula 3, trying to minimize the similarity (cosine function) between the centroid of each cluster (C_r) and the centroid of the entire set (C), weighted by the size of each cluster (n_r).

$$minimize\ E1 = \sum_{r=1}^{k} n_r\ cos(C_r, C) \tag{3}$$

2.2.4 Hybrid Criterion Function - $H1$

$H1$ is the ratio of an *internal* ($I1$) and *external* ($E1$) criterion function. As we can see $H2$ is directly proportional to a maximization function and inversely proportional to a minimization function and thus itself is a maximization function.

$$maximize\ H1(m) = \frac{I1(m)}{E1(m)} \tag{4}$$

2.2.5 Hybrid Criterion Function - $H2$

$H2$ is formulated along similar lines as $H1$, the difference being the criterion function used in the numerator of the ratio - $I2$ instead of $I1$. For similar reasons as those for $H1$, $H2$ is also a maximization function.

$$maximize\ H2(m) = \frac{I2(m)}{E1(m)} \tag{5}$$

Table 1: A contingency table for the word pair "fine wine" in a hypothetical corpus.

	$wine$	$\overline{wine}$	
$fine$	$n_{11} = 15$	$n_{12} = 91$	$n_{1+} = 106$
$\overline{fine}$	$n_{21} = 4$	$n_{22} = 92,696$	$n_{2+} = 92,700$
	$n_{+1} = 19$	$n_{+2} = 92,787$	$n_{++} = 92,806$

2.3 The Log-likelihood Ratio

The Log-likelihood ratio (Dunning [1994]) is a statistical measure of association between a pair of words occurring in a set of documents such as a collection of web pages, e-mails or instances of an ambiguous word. It assigns a score of association to a pair of words that co-occur with each other. This score of association is a measure of how significant the co-occurrence is, as opposed to being a random co-occurrence (happening by chance). The log-likelihood ratio for word co-occurrences is based on a contingency table, such as the one shown in Table 1. This contingency table shows the frequency counts of co-occurrence of the word pair "fine wine" in a hypothetical corpus. $n_{11} = 15$ represents the number of times these words occur together in the corpus. $n_{12} = 91$ represents the number of times the word "fine" *does not* occur with the word "wine". Similarly $n_{21} = 4$ represents the number of times the word "wine" *does not* occur with the word "fine". Finally, n_{22} represents the total count of words other than "fine" and "wine" in the corpus. The marginal totals $n_{1+} = 106$ and $n_{+1} = 19$ represent the individual frequency of occurrence of the words "fine" and "wine" respectively in the entire corpus, and the total number of words in the corpus is $n++ = 92,806$.

Given such a contingency table (as in Table 1), the log-likelihood measure of association (symbolically G^2) of the word pair that is represented by the table is:

$$G^2 = \sum_{i=1,2;j=1,2} 2 * n_{ij} * log \frac{n_{ij}}{e_{ij}}$$

where $e_{i,j}$ is the randomly *expected* count in the cell i, j of the contingency table based on chance, which is calculated using the marginal totals and the total word count. That is,

$$e_{ij} = \frac{n_{i+} * n_{+j}}{n_{++}}$$

Intuitively one can see that the log-likelihood ratio compares the actual frequency counts in the contingency table to the randomly expected frequency counts, and the higher this ratio the better is the significance of the word pair.

It can be shown that the log-likelihood measure as defined above is asymptotically distributed with a Chi-squared distribution (χ^2) with one degree of freedom. Intuitively, given the marginal frequency counts in the contingency table, filling in any one particular value n_{ij} immediately decides all the other n_{ij} values. A log-likelihood *test* of association can therefore be performed to ascertain the significance of a word pair based on the confidence thresholds of a χ^2 distribution with one degree of freedom. In particular, a log-likelihood score greater than 3.841 shows 95% confidence in the significance of a word pair and a log-likelihood score greater than 6.635 shows 99% confidence in the significance of a word pair.

2.4 Cluster Purity

As the name suggest, cluster purity measures how pure or impure a cluster and thus the clustering solution is. A cluster is said to be pure if all the context-vectors present in it belong to the same category/sense. If a cluster contains context-vectors from multiple categories then the cluster is assumed to belong to the category with majority context-vectors. The purity for a cluster S_r containing n_r context-vectors is defined as:

$$P(S_r) = \frac{1}{n_r}\, max(n_r^i) \tag{6}$$

where n_r^i is the number of context-vectors from the category i that are present in the cluster S_r.

Based on this definition the purity of an entire clustering solution is defined as the weighted sum of the individual cluster purities:

$$Purity = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \tag{7}$$

3 Methodology

We briefly describe our approach to clustering similar contexts, and then the subsequent sections elaborate upon the individual phases. While we build upon work by Purandare and Pedersen [2004], Purandare [2004], this thesis adds to that significantly (Kulkarni and Pedersen [2005b], Kulkarni and Pedersen [2006], Pedersen and Kulkarni [2006b], Pedersen and Kulkarni [2006a]) which will be pointed out as we proceed.

Also note that the methodology described below has been implemented and is available as a freely distributed open source package called SenseClusters[3], which internally uses a few other packages for various tasks. Each of these packages will be referenced at appropriate places in the description.

The goal here is, given a corpus containing set of contexts (sentences, paragraphs, documents), to cluster the contexts based on their similarity.

We start with the lexical feature extraction phase. Lexical features are short textual entities like single words or word-pairs or in other words, are formed out of lexicals found in the given corpus and do not contain any grammatical information. These lexical features can be considered as the representatives of the underlying corpus or also as the entities which can translate a corpus to an abstract level. With SenseClusters these features can be *unigrams*, which are high frequency individual words, or *bigrams*, which are ordered pairs of words, or *co-occurrences*, which are unordered pairs of words or *target-co-occurrences* which are unordered pairs of words in which one of the words is the word to be disambiguated, also referred to as a *target-word* or a *head word*. These features can be extracted from a separate set of feature selection data (which is not clustered) or from the data that is to be clustered. These features can be selected by simple ranking of their frequency of occurrence or by using statistical tests of association. SenseClusters integrates the N-gram Statistics Package (NSP)[4] to extract raw n-grams which SenseClusters translates into lexical features as described in Section 3.1.

The next step is to represent each context in terms of the features extracted in the earlier step. This translates each context into a vector representation. As shown in the Figure 1 one can choose to do

[3] http://senseclusters.sourceforge.net

[4] http://www.umn.edu/home/tpederse/nsp.html

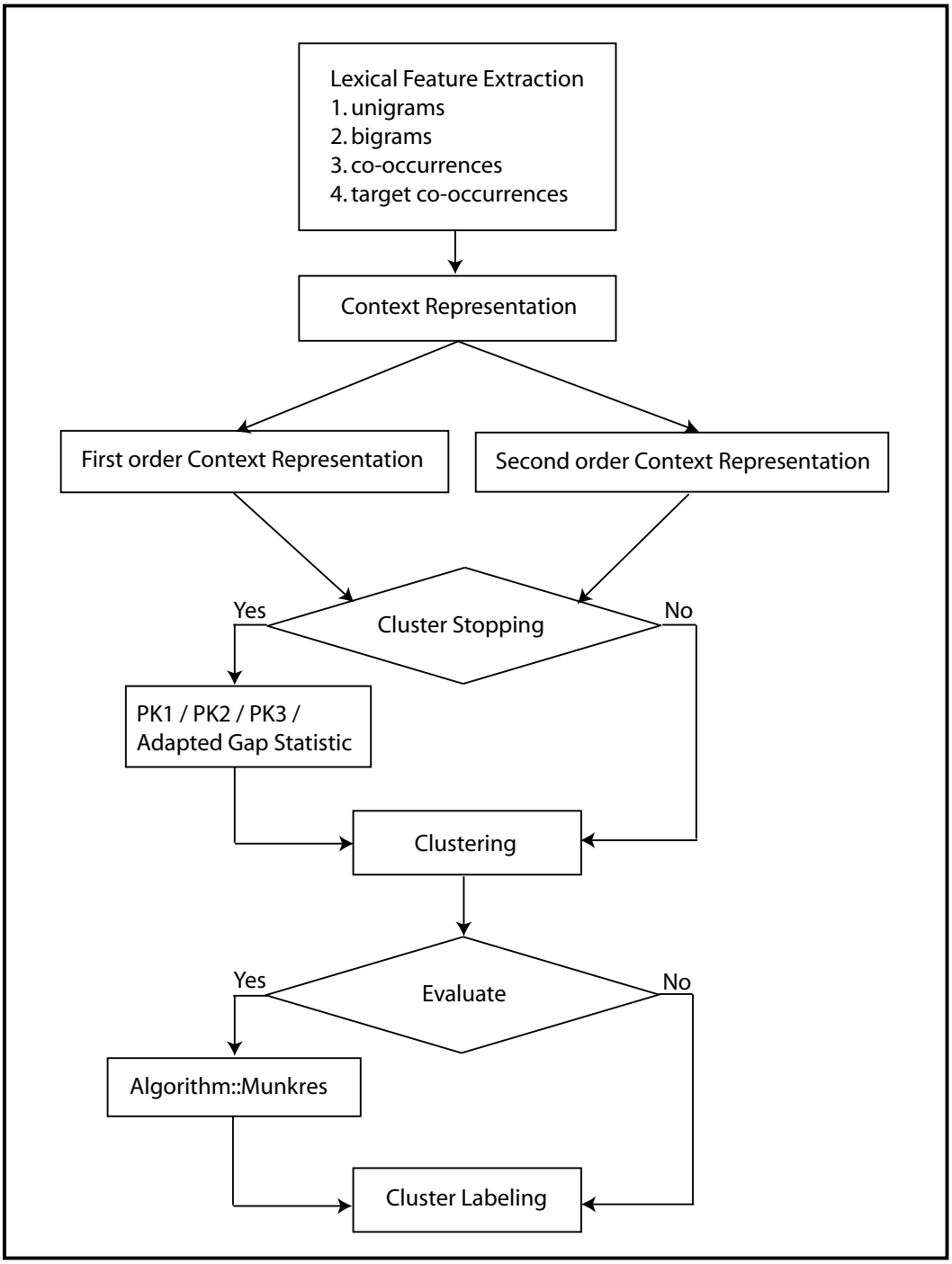

Figure 1: Flowchart showing high-level design of our approach

this using either first order context representation or second order context representation. The exact details about these representations are described later in Section 3.2. Every context for name discrimination or word sense discrimination experiment consists of a *head word* and the surrounding words/sentences. While for email clustering or document clustering, the complete email or document is represented as a context without any *head* word and thus we refer to this form of context as *headless*! It is important to note that we introduce *headless* clustering in this thesis and it is thus one of our contributions towards this work of clustering similar contexts. We have also further extended the *headless clustering* to *word clustering* where we try to form clusters of related words. Such clustering has a potential to bring out various kinds of relations among words like antonymy, synonymy, "is a" etc. However, *word clustering* is not pursued in this thesis.

Once each context (either *headed* or *headless*) is represented as a vector, all these vectors are clustered such that the intra-cluster similarity for each cluster is maximal while the inter-cluster similarity is minimal. Further details about clustering are given in the Section 2.1. Clustering algorithms typically ask for the number of clusters into which the contexts should be separated. A user can choose to specify the number of clusters he expects from the dataset but such knowledge is rare for a user to have and thus we have introduced four measures, each of which tries to predict the appropriate number of clusters for a dataset based purely on the properties exhibited by the dataset. We refer to the process of predicting the number of clusters for a dataset as *Cluster Stopping*. Details about this process are described in Section 3.3. Clustering of contexts into either the specified or predicted number of clusters follows. SenseClusters utilizes CLUTO[5], a suite of various highly efficient clustering algorithms for its clustering requirements.

Further each cluster thus formed is assigned a list of significant words referred to as its label. These words are picked from the contents of the cluster and thus these cluster labels summarize the contents of the cluster and provide an easy automated way to identify the entity that the cluster represents. See Section 3.6 for details.

Next we elaborate on each step of our methodology.

[5]http://www-users.cs.umn.edu/∼karypis/cluto

3.1 Feature Identification

Feature Identification/extraction is the first step and an important building block of our methodology. The lexical features identified in this phase determine how accurately the contexts will be represented and thus the extent to which the similarity between the contexts can be captured.

These lexical features can be specified to be selected from the test data or from separate raw data which is not to be clustered but is only used for feature identification and thus referred to as the feature selection corpus. Note that in either case, the feature selection process is completely unsupervised, i.e., no manually annotated information is used, nor is any form of external knowledge resource used, nor are the answer keys referred when using test data with answer keys.

SenseClusters supports four different types of lexical features and each one of them captures slightly different information from the others. The supported features are unigrams, bigrams, co-occurrences and target co-occurrences. Unigrams are the simplest of the features. These are individual words that occur in the corpus more often than some cut-off frequency. Thus unigrams represent the class of single words from the feature selection corpus that occur frequently enough to clear the basic test of noise elimination. Bigrams are ordered pairs of words, thus "World History" and "History World" would be two separate bigrams. The word-pairs can optionally have intervening words between them in the actual corpus which are ignored while forming the bigram. The number of the intervening words that are allowed between the word-pair is decided by the specified window size. For example in the phrase "World and its ancient History" the intervening words "and its ancient" are ignored while selecting the bigram "World History" and the window size must have been set to at least 5. In other words, the window size should be computed as the acceptable number of intervening words + 2 (for the 2 words in the bigram). Co-occurrences are unordered word-pairs, thus "World History" and "History World" will be two occurrences of the same co-occurrence feature. Co-occurrences can also have intervening words in the actual corpus. With co-occurrences too the window size is used to specify the acceptable number of intervening words. By retaining the positional information for bigrams we expect to capture phrases or collocations whereas with co-occurrences we identify the word-pairs that tend to occur together. Finally the target co-occurrences are the unordered pairs of words where one of the words is the target-word. As a result this feature type can be used only with *headed* context or more specifically with name

discrimination or word sense discrimination datasets and cannot be used with *headless* contexts like email datasets. This feature is based on the motivation that the words in the immediate vicinity of the ambiguous word are more likely to help characterize the true underlying entities and also are less likely to be complete noise.

Although one can choose to use all the identified features, we typically select and use only the significant features with the intention of filtering the noisy features. Since our approach is completely unsupervised we cannot use supervised methods of finding significant features like information-gain and thus instead we rely on eliminating noisy features using various techniques described below and then hoping that what are left are good features. One of the most generic and basic methods for lexical feature filtering is to use a *stoplist*. A *stoplist* is a list of words for a language that fall under the category of functional words i.e. belong to the set of closed-words. Such words tend to co-occur with all/many context words and thus result into very noisy and useless features. We typically make use of *OR* stoplisting with all the types features, except for *unigram features*. In *OR* stoplisting, if a feature contains one or more words from the stoplist then the feature is dropped from the set of candidate features. The simplest unsupervised method of selecting (hopefully) significant features is to rank them according to the frequency of their occurrence in the corpus and then select the *n* most frequently occurring features. This is the only technique that can be used with unigram feature types. For bigrams and co-occurrences more complex techniques - statistical tests of association like log-likelihood ratio, mutual information, point-wise mutual information etc. can be used. These techniques help identify the significant word-pairs - these word-pairs that occur together more often than expected by chance.

3.2 Context Representation

Once we have the set of identified lexical features we proceed to represent each context in terms of these feature either directly or indirectly. SenseClusters can represent contexts using first order context representation or second order context representation.

3.2.1 First order context representation

The first order representation is based upon the technique that Pedersen and Bruce [1997] developed. In this representation we create a matrix with each context represented by a row and each selected feature represented by a column. The value at cell [x,y] in this matrix represents the frequency of occurrence of the yth feature in the xth context. This matrix tends to be a large sparse matrix with mostly binary values. The sparsity of the matrix is due to the fact that the number of features present in any given context is generally small compared to the size of the identified feature set. The binary nature is exhibited if the contexts are relatively small, in which case, any open class word or word-pair would rarely occur more than once in a context.

Singular Value Decomposition (SVD) is a technique from Linear Algebra which we use to achieve dimensionality reduction. SVD can be optionally used to reduce the dimensionality of the above described matrix and thus to reduce the sparsity. If used, SVD also has a effect of smoothing values. We maintain the post-SVD column dimension of the matrix to a minimum of 10% of the actual column dimensions or 300. Thus if the original column dimensions were more than 3000 then the matrix is reduced to 300 columns but if the original column dimensions were less than 3000 then the reduced column dimensions are 10% of the actual size. Previous to this thesis, one could not apply SVD to a context by feature matrix generated using the first order context representation, however now we have added this functionality and SVD can be easily used with both first and second order context representations.

Irrespective of whether SVD was applied to the matrix or not, each row of the matrix is a vector representing the context at the row. When SVD is applied to a matrix the rows of the resultant matrix can be looked at as the reduced context-vectors.

3.2.2 Second order context representation

Our second order context representation is adapted from Schütze [1998]. We start by representing the identified bigram or co-occurrence features in a word by word matrix format where first word of the feature is represented across the row, the second word across the column and the cell values are either their co-occurrence frequencies or the statistical scores of test of associativity. Note that this

matrix does not incorporate any information from test data in it. Also note that the unigram feature type cannot be used in this representation because as we have seen this representation requires the features to be composed of word-pairs. SVD can be employed to the word by word matrix for dimensionality reduction and smoothing of the values. Each row of the matrix, either with or without SVD can be interpreted as the word-vector for the word it represents. Every word-vector carries the information about the co-occurrence pattern of the word, that is, this word-vector gives the information about which other words this word occurs with. These word-vectors are used to create the second order context representation. A context-vector is created by averaging the word-vectors for the words that occur in the context i.e., the word-vectors for all the words that occur in a given context are looked up in the word-vector matrix and are averaged together to create a second order context representation for the context.

In either the first or second order context representation, for datasets with *headed* contexts one can restrict the words that form a context. For example, if originally the context size was 101 (head-word + 50 words to the left + 50 words to the right) then one can reduce the context to, lets say, 21 words (head-word + 10 left words + 10 right words). If using a first order context representation, this implies that instead of 101 words now only 21 words will represent the context in the context-feature matrix. For a second order context representation, it implies that now only 20 word-vectors will contribute towards the formation of the context-vector instead of 100 word-vectors. We refer to this as restricting the test-scope and have shown the improvement gained by its use in Kulkarni [2005] and Pedersen et al. [2005]. This is based on the theory that a word positionally nearer to the target-word is more likely to be related to the target-word than a word farther from it and using only such words leads to a reduction in the noise.

On similar lines, the scope for the feature selection data can also be restricted. We refer to this as restricting the training-scope. To be able to restrict the training scope, the feature selection data needs to be of *headed* type. When the training-scope is restricted, the content for choosing the features is reduced to the resulting smaller contexts. This can be looked at as a method of feature selection, or from the opposite perspective, as feature pruning.

Once the contexts are represented in vector format we proceed to find the number of clusters that these context-vectors naturally separate out into.

3.3 Predicting k via Cluster Stopping

Please note that Cluster Stopping was not supported previously and thus is one of the major contributions in this work.

To be able to optimally cluster the generated context-vectors we need to know the appropriate number of clusters (k) that should be used to group the vectors. If the user has a priori knowledge of how many clusters to expect from the data then we can proceed to cluster the context-vectors into those many number of clusters. However such knowledge is rare for any user to have. Keeping this in mind we have designed and implemented four measures which automate the process of predicting the optimal number of clusters from the context-vectors. We refer to this as the cluster stopping process, and thus the measures as the cluster stopping measures. SenseClusters integrates each of these four measures and provides the option to specify either one or a group or all of them to predict the k value. If more than one measure is specified then context-vectors are iteratively clustered into number of clusters predicted by each of the selected measures and thus in such cases multiple sets of results are generated. Also note that all the four cluster stopping measures are independent of the clustering method used, i.e., they can be used with almost any type of clustering method - partitional, hierarchical or hybrid.

Each of the four cluster stopping measures are described in subsequent sections and what follows is a description common to all the four measures.

We formulate our cluster stopping measures using the criterion functions (crfun) described in Section 2.2.

Each of the criterion functions encode and exhibit the properties of data in their own way, and this is the primary motivation for choosing these functions as the building blocks of our measures. Despite their differences, the values of internal and hybrid crfuns exhibit a typical pattern, which we refer to as the *knee* where the crfun values increase as the number of clusters m increase but only until some m value beyond which the crfun values remain steady. This pattern indicates that we were gaining some improvement by clustering the data into more and more clusters until some point beyond which adding more clusters does not provide any improvement. Similarly the external crfuns values generate *elbow* shaped patterns for any data that has more than one cluster. Figure

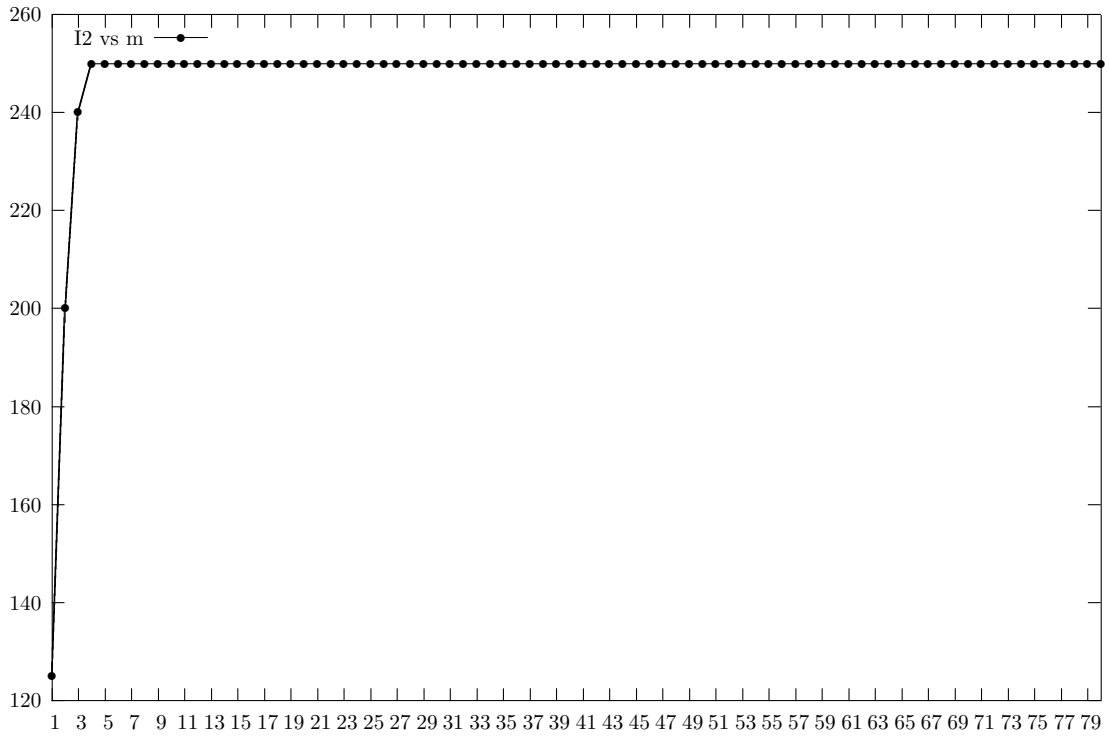

Figure 2: Criterion function ($I2$) vs. m for contrived data

2 shows one such sharp *knee* generated by the $I2$ values computed for a contrived data containing 80 instances which we know would be best clustered into 4 clusters. This contrived data consists of context-vectors which are either identical or are orthogonal to each other and thus exhibit clear similarity or clear distinction. The plot in Figure 2 can be read as: the internal similarity ($I2$) for the generated m clusters increases as the m value goes from 1 to 4, however no increase or advantage is observed by clustering the data into more than 4 clusters. This is precisely the point which most cluster stopping measures try to locate - the point at which no further improvement is seen - the start of the plateau.

If we were to formulate a cluster stopping measure based only on such contrived data then a trivial and yet effective cluster stopping measure, when using maximization criterion functions like internal and hybrid, would be to choose an m value for which the crfun(m) is maximum. Extending on those lines, for the minimization criterion functions like external crfun the cluster stopping measure would

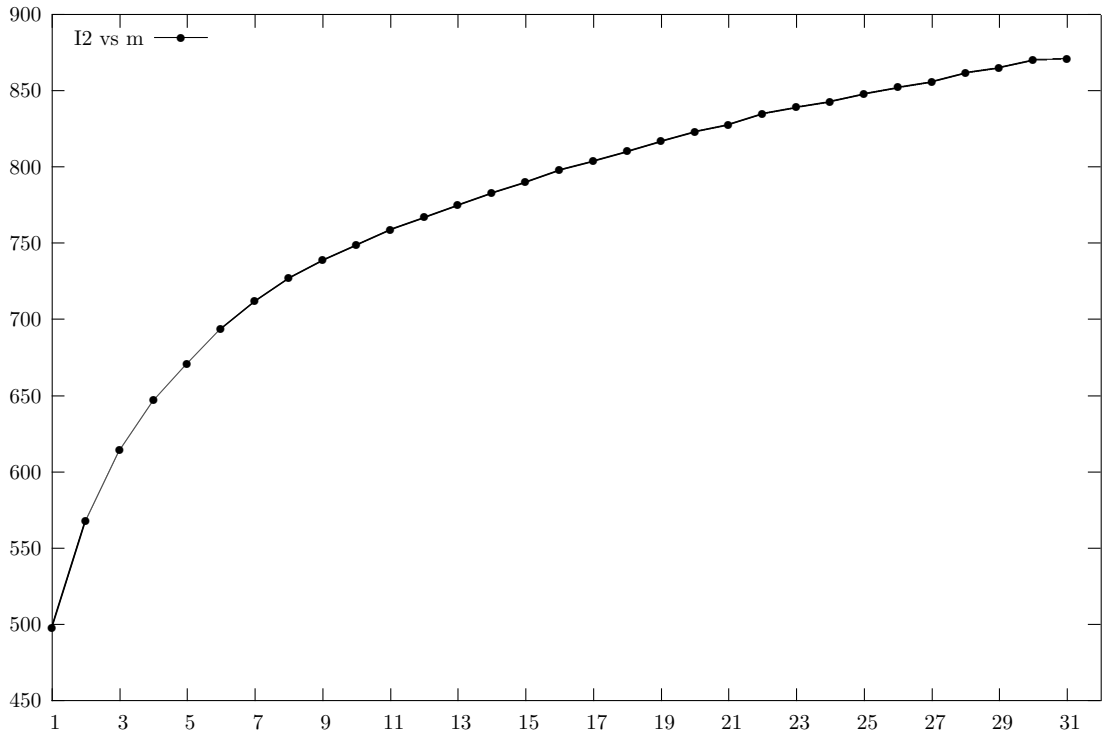

Figure 3: Criterion function ($I2$) vs. m for real data

be to choose an m for which the crfun(m) is minimum. However, as always the contrived examples and formulation based on contrived data do not scale up to the real data and the real problems. For example, Figure 3 shows a plot generated with $I2$ crfun when applied to a real dataset containing 900 contexts with the optimal number of clusters being 4. As one can observe, the is no *knee*, unlike the contrived data. In other words it is not clear where the improvement has stopped or where the plateau has begun and thus locating one specific point which would cluster the data optimally becomes a non-trivial task.

Given the above observations about the disparity among the properties shown by contrived data and real data, henceforth we will look only at real data and its properties.

We have observed that iteratively clustering the data into m clusters, where m is set to 1 through $\#contexts$, for finding the k value is highly inefficient and unnecessary. Instead, we repeatedly cluster the data only until we observe at least a small amount of improvement (for maximization

crfuns) or decrease (for minimization) in consecutive crfun values. We realize this by computing the difference between the consecutive crfun values and comparing this difference with what we refer as *delta* value. We set this *delta* value to the smallest possible non-zero value with the precision of the crfun value. For example, if the crfun value is 0.873 then the *delta* value would be set to 0.001, if crfun value is 0.53 then the *delta* would be set to 0.01 and for crfun value of 290, *delta* will be set to 1. In Figure 3 we can see this mechanism at work - the vectors iteratively get clustered only until m=31 because the difference between the $I2(30)$ and $I2(31)$ is less than the *delta* value of 1. We refer to this m value at which the difference becomes smaller (larger for minimization functions) than the *delta* value as the *deltaK* value. Thus for Figure 3 the *deltaK* value is set to 31.

3.3.1 PK1

Our very first cluster stopping measure, $PK1$, was inspired by Mojena [1977], who observes the *knee* and *elbow* patterns (although without giving them these names) exhibited by the criterion values in the realm of hierarchical clustering. He infers from this observation that if the criterion values for two consecutive m values are significantly different then it indicates that the clustering should not be stopped yet. In other words we say that we look for the start of the plateau.

We formulate the $PK1$ measure as follows with the aim of finding the crfun value which is on the cusp of the regions formed by the set of significantly differing crfun values and the set of significantly not-differing crfun values.

$$PK1(m) = \frac{crfun(m) - mean(crfun[1..deltaK])}{stdev(crfun[1..deltaK])} \tag{8}$$

The above formula is similar to that used to convert a Normal distribution to a Standard Normal distribution or Z distribution. Via the $PK1$ measure we are trying to find a z-score (PK1(m)) in either of the tails of the distribution that would correspond to the appropriate k value. With this technique although we can narrow down to a set of candidate k values, without having to manually set any threshold, selecting and recommending a specific k value is not possible without a threshold value. We have empirically found that a threshold value of -0.7 works best with our domain and data. For maximization functions, if m is the first value for which the crfun(m) value is greater than

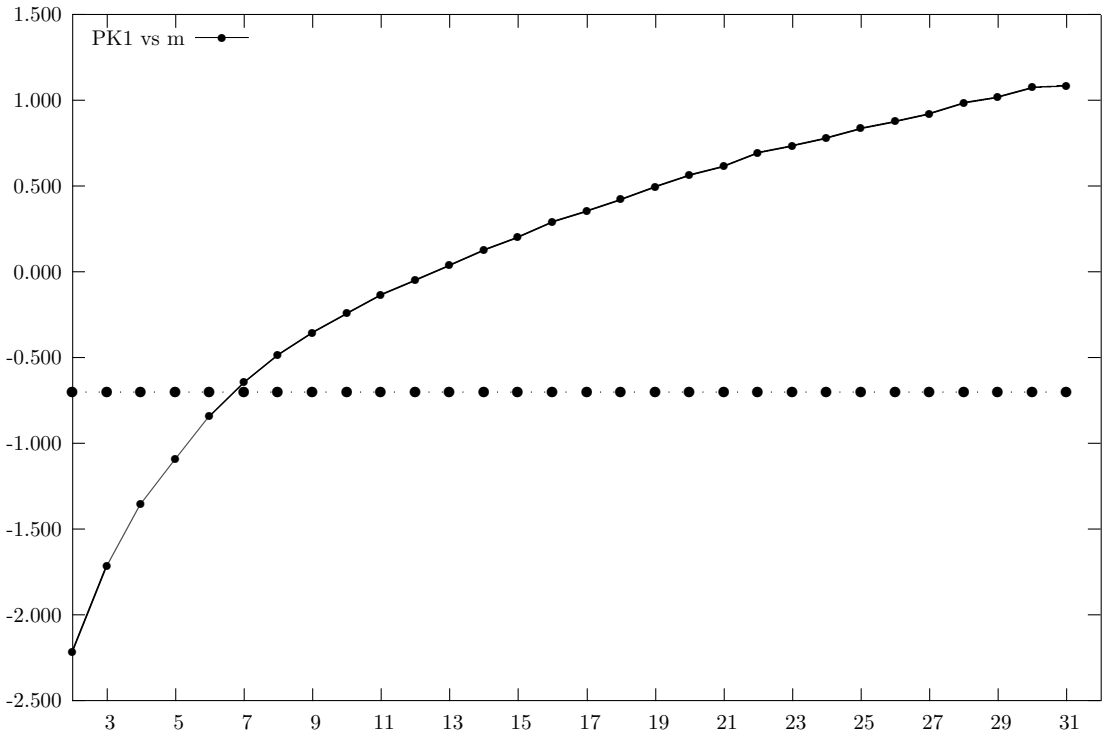

Figure 4: $PK1$ vs. m

-0.7 then $m-1$ is selected as the k value and for minimization functions, if m is the first value for which crfun(m) is less than -0.7 then $m-1$ is selected as the k value. Keeping in mind that the threshold value of -0.7 might not be the appropriate threshold value for every domain and dataset, SenseClusters provides a facility to specify and thus use a threshold value other than -0.7.

Figure 4 plots the $PK1$ scores calculated for the $I2$ values shown in Figure 3. The horizontal line across -0.7 indicates the applied threshold value. As the plot shows, $I2(m) = -0.6073$ is the first crfun value that is greater than the threshold value of -0.7 and thus $m-1=6$ is the selected k value.

Note that of the four cluster stopping measures, $PK1$ is the only measure that requires the user to set a threshold, however we do realize that expecting a user to set such a threshold value could be even more esoteric than expecting the user to already know the number of clusters.

3.3.2 PK2

The $PK2$ measure and the next cluster stopping measure, $PK3$, were inspired by looking at various $I2\ vs\ m$ plots and observing that the pattern or the *knee* shape of these plots can be captured by comparing consecutive crfun values. As a result we formulated $PK2$ as a ratio of two consecutive crfun values, as shown in formula 9. The $PK2$ scores indicate the improvement gained by going from $m-1$ clusters to m clusters.

$$PK2(m) = \frac{crfun(m)}{crfun(m-1)} \tag{9}$$

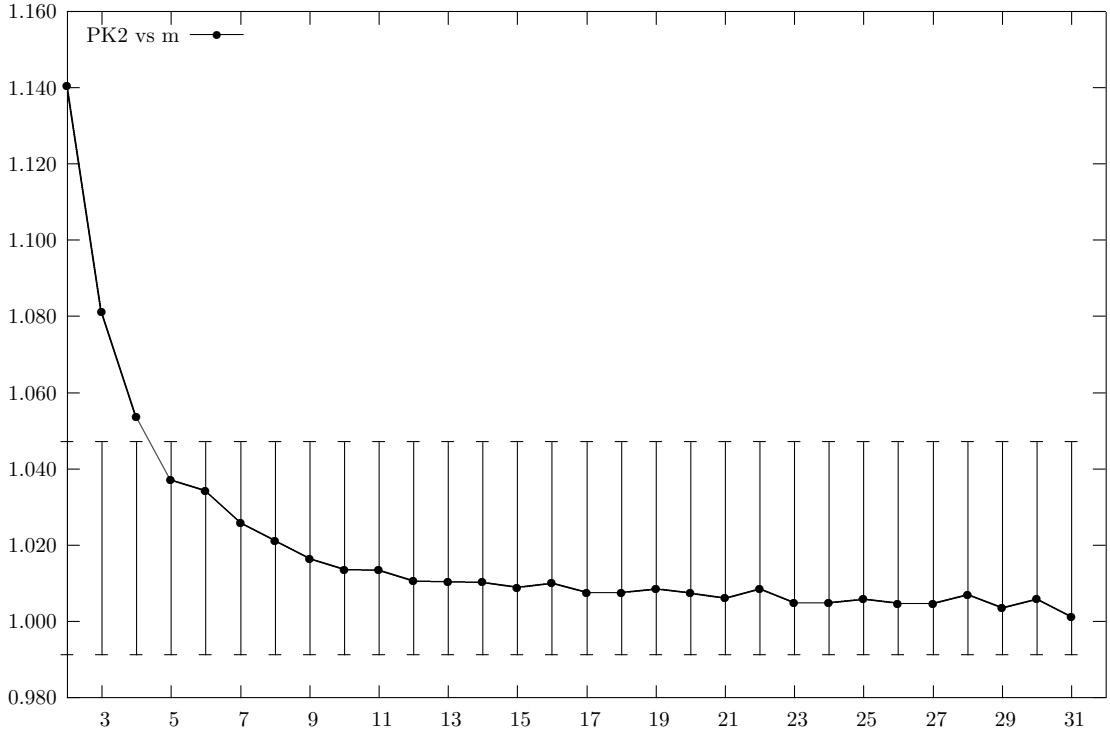

Figure 5: $PK2$ vs. m

The m values at which this ratio is approximately 1 are all candidate k values because the ratio value of 1 indicates that the improvement was negligible which implies that the points are on the plateau. To choose one of the candidate k values we calculate the mean and the standard deviation

of the $PK2$ scores and pick the m value which has a score that is closest from outside the interval defined by one standard deviation. We adopt this selection procedure to ensure that the selected k value is a point neither on the rising edge of the knee nor on the plateau region but is right on the knee. $PK2$ is similar in spirit with Hartigan [1975].

Figure 5 shows the plot of $PK2$ values versus the m values for the $I2$ values from the Figure 3. As expected, the shape of the plot indicates that the gain obtained by going from $m-1$ clusters to m clusters is decreasing as m increases - exhibiting the law of diminishing returns. The vertical bars across the plot indicate the interval defined by one standard deviation. The crfun(m)=1.0537 value at $m=4$ is nearest from the outside to the interval of one standard deviation and thus 4 is selected as the k value.

3.3.3 PK3

$PK3$ came as a natural extension of $PK2$. Here we look at three consecutive crfun values instead of two consecutive crfun values. The primary motivation for looking at an additional crfun value, was that looking at three consecutive values simultaneously helped identifying a knee (an angle) more accurately than looking only at two values.

We formulate $PK3$ as:

$$PK3(m) = \frac{2 \times crfun(m)}{crfun(m-1) + crfun(m+1)} \tag{10}$$

Given this formulation, a triplet of consecutive crfun values for which $PK3$ score is greater than or equal to 1 implies that the three points are linear, either on the rising edge of the knee or on the plateau. To select a point that is on neither of these plain, i.e., but is right on the *knee*, we use a similar strategy as for $PK2$ - we calculate the mean and the standard deviation of $PK3$ scores and select the m value for which the $PK3$ score is greater than 1 and is closest from outside the interval of one standard deviation. A sample plot of $PK3$ scores for the $I2$ values from the Figure 3 is shown in Figure 6. The k selection criterion of one standard deviation selects 4 as the optimal number of clusters for the underlying dataset, which as we know is the true or expected number of clusters for this data.

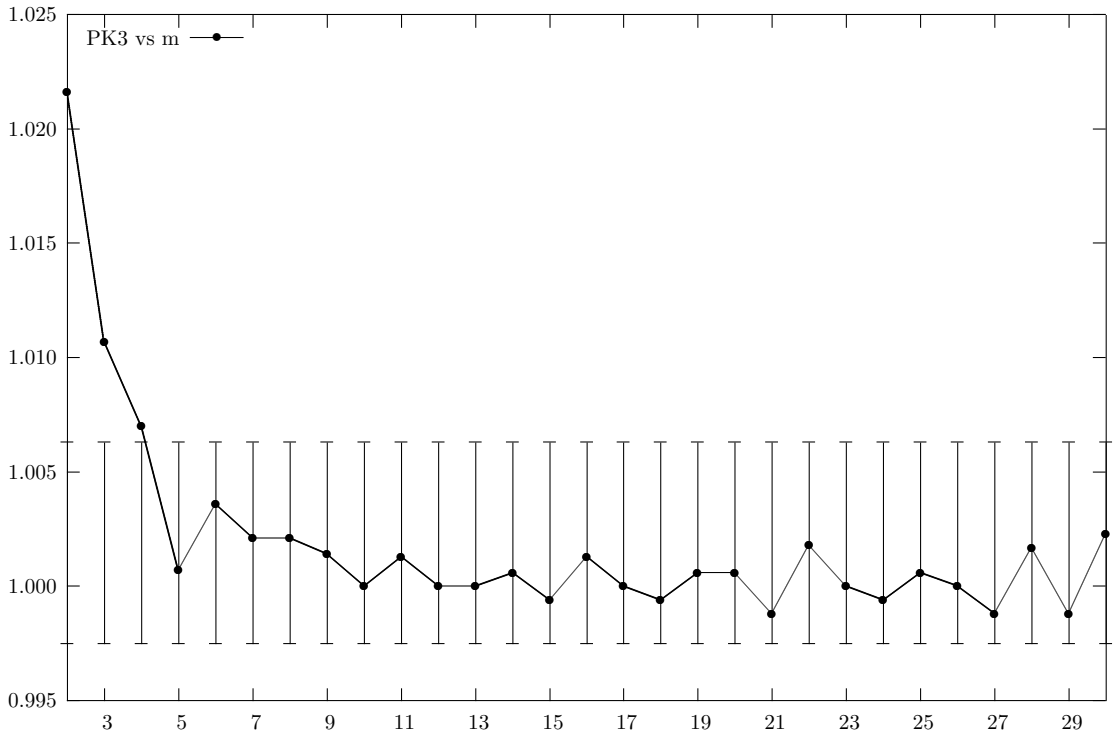

Figure 6: $PK3$ vs. m

3.3.4 Adapted Gap Statistic

Our final cluster stopping measure is based on a method proposed by Tibshirani et al. [2001] called the Gap Statistic. We first briefly describe the original Gap Statistic and then elaborate on our adapted version of the Gap Statistic.

Gap Statistic uses *within cluster dispersion* as its primary metric. The *within cluster dispersion*, or in short, the *error* values for any dataset when clustered into m clusters iteratively, exhibits a pattern which is often referred to as the *elbow* because of its shape. This shape can be read as showing that the *error* value is high initially and then it decreases as the m value increases until some m value beyond which it remains constant. This *elbow* can be looked at as a mirror image of the *knee* pattern that we have described earlier which also implies that the aim is to locate the m value for which the *error* does not decrease any further and remains constant. Gap Statistic uses distance measures like Euclidean, Squared Euclidean, Manhattan etc. to formulate this *within cluster dispersion* that is the

error measure.

The main idea in Gap Statistic is to standardize the plot of this *error* metric generated for a dataset by comparing it with an expected plot under appropriate null reference distribution. The adopted null model is the case of single cluster ($k = 1$) which is rejected in favor of k clusters (where $k > 1$) if sufficient evidence is exhibited by the data. As one can see this is similar to the hypothesis testing idea, where the null hypothesis is that the dataset has no pattern. In other words, the appropriate number of clusters for the dataset is one and the alternative hypothesis is that the dataset has some pattern in it and should be clustered into k ($k > 1$) clusters.

The idea for adapting the original Gap Statistic sprung from two observations:

1. The *criterion functions* exhibit properties similar to the ones exhibited by the *error* measure.
2. Using the *criterion functions* instead of the *error* made the complete process exponentially faster and efficient.

Thus in the adapted version of Gap Statistic we standardize the plot generated by the *criterion function* values by comparing it with an expected plot under null reference model of one cluster. The details about how we generate a null reference model are given in the following paragraph.

We start with a matrix of context-vectors, that is, a matrix containing contexts along the rows and features along the columns. The requirement is to generate a null reference matrix (*no pattern matrix*) based on the given context-vectors matrix. This implies generating a random matrix that obeys the following constraints:

1. The generated matrix should be of the same dimensions as the given context-vector matrix.
2. The generated matrix should be drawn from the marginal distributions of the given context-vector matrix.

The marginal distributions of the context-vector matrix are: the context distribution along its rows, and the feature distribution along its columns.

In the discussion below we assume a matrix representation of the dataset where the features are by the columns and the contexts are by the rows. To generate a null reference model with the properties

described above we generate random values for the cells of the reference data matrix while holding the row and column marginals to the same values as that of the test matrix. In other words, given the row and column marginals for the test matrix we generate random values for the reference matrix such that the final row and column marginals of the reference matrix would also sum-up to the same value. The high level two step algorithm that we use for this requirement is as follows:

For each cell while traversing the matrix in row-major interpretation:

1. Generate random number based on constraints (using the algorithm that follows)
2. Reduce row and column marginals by the generated random number

End for

The step 1 above for generating a random number for a cell c(i,j) based on constraints is broken into the following algorithm:

1. Find the range i.e., the lower and the upper threshold on the random number to be generated

1a. The upper threshold is set to the minimum value between the row marginal value for row i and column marginal value for the column j

1b. If the upper threshold is set to 0 then set the lower threshold to 0 too else sum together the column marginals for all the columns past the current column. This sum gives the total of the column marginals yet to be satisfied beyond the current column. Subtract this sum from the current row marginal to compute the lower bound on the random number.

2. Generate random number between the range defined by the lower and upper threshold (inclusive).

We have further adapted this algorithm to support real valued random number generation and also to support negative valued marginals. The latter case arises when Singular Value Decomposition is applied to the test matrix and the resulting reduced matrix contains negative values and thus negative marginals.

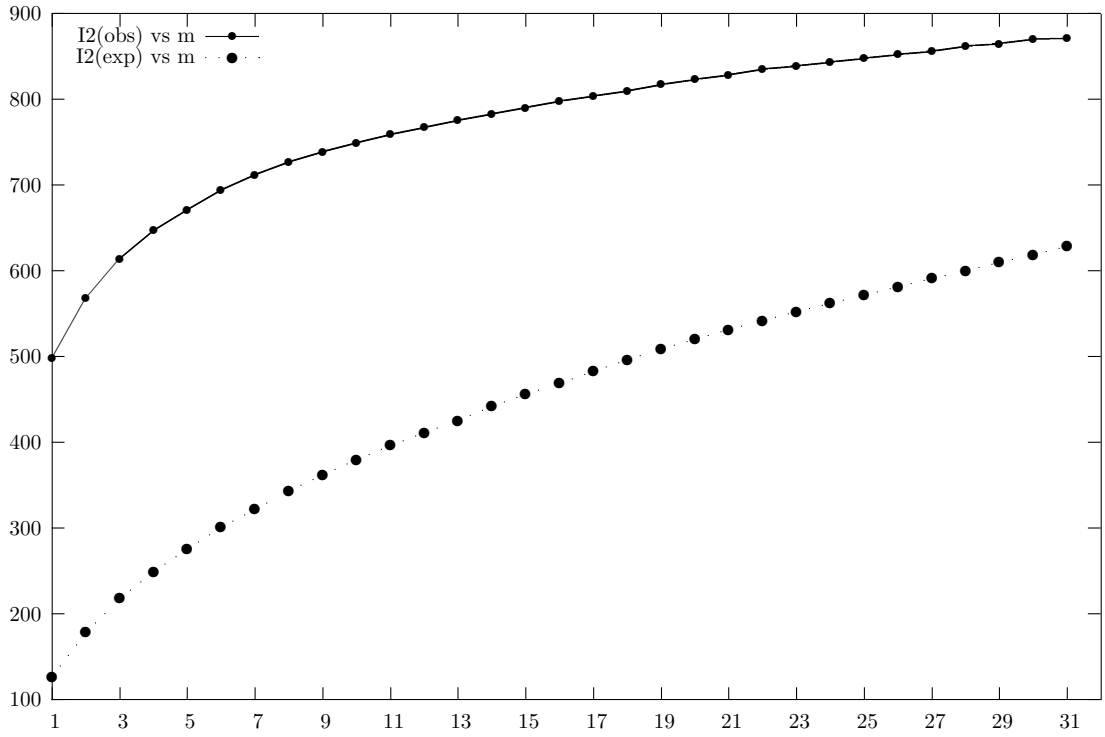

Figure 7: Observed and Expected vs. m

Once a reference matrix is generated based on the test matrix marginals using the algorithm described above we proceed to iteratively cluster it into m clusters, where $m = 1..deltaK$ and record the criterion function values at each iteration. The plot of such crfun[$1..deltaK$] values for the previously described dataset with expected number of clusters as 4 is shown in Figure 7 with a broken line. The plot with full lines is a re-plotting of the crfun plot for the test data shown in Figure 3 on a different scale. Although not very easily perceivable we can see that the plot for test data exhibits a slight *knee* pattern while the plot for the reference data is almost linear or monotonically increasing. If we compute the difference of the *gap* between these two plots at each m value then the plot of resulting values would look like the one shown in Figure 8. As we can see in this Figure 8 that comparing these two plots brings out the pattern present in the test dataset and we can see a clear peak at $m = 4$ which would be selected at the predicted k value for this dataset by the Adapted Gap Statistic.

We have packaged together the above functionality of generating a random matrix based on the provided marginals in a Perl module named Algorithm::RandomMatrixGeneration[6] and have made it freely available on CPAN[7].

3.4 Context Clustering

Once the contexts are represented in vector format and the number of clusters (k) to expect from the set of vectors is known, the next step is to cluster these vectors into k clusters. We seamlessly integrate and use a suite of clustering algorithms called CLUTO[8] for context clustering.

We have found that for our domain a hybrid clustering algorithm called *repeated bisection - rb* and its refinement *rbr* generally performs the best. Details about different types of clustering algorithms are covered in Section 2.1

[6] http://search.cpan.org/dist/Algorithm-RandomMatrixGeneration/

[7] http://www.cpan.org/

[8] http://www-users.cs.umn.edu/∼karypis/cluto

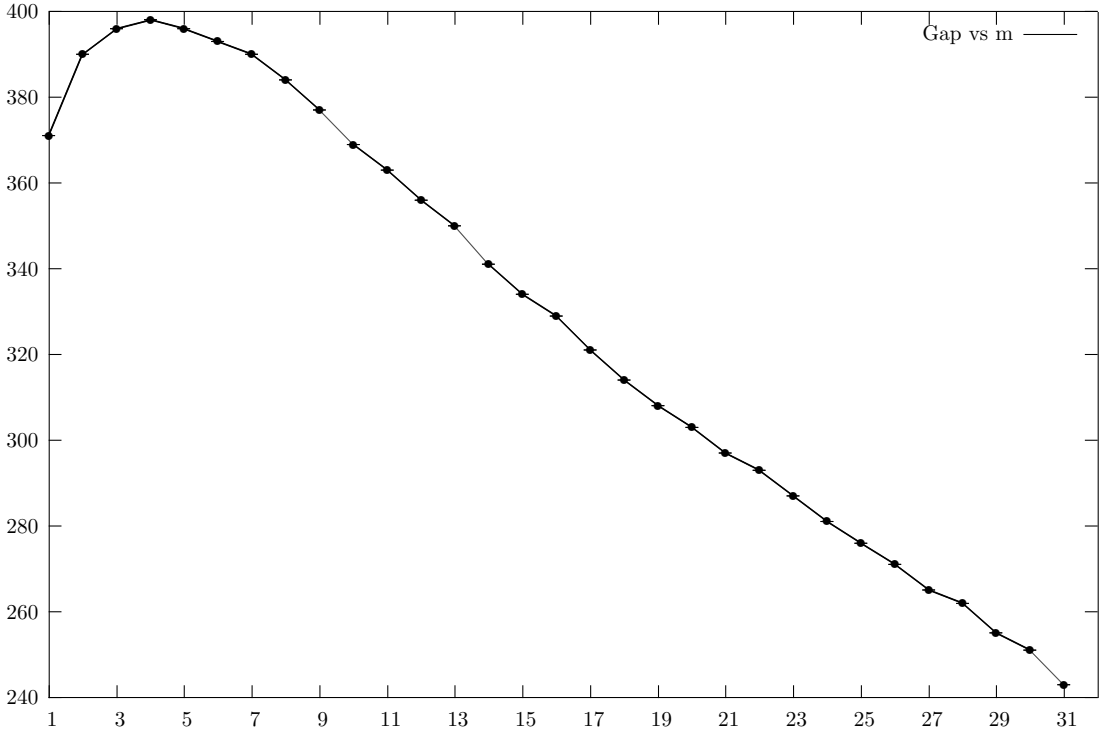

Figure 8: Gap vs. m

Table 2: Sense Assignment Matrix (3-way)

	Congressman	**Professor**	**Director**	**T**
C0	42	0	38	80
C1	86	0	2	88
C2	89	12	17	118
T	217	12	57	

3.5 Evaluation

If the true or expected grouping of the contexts is provided with the test data, in other words, if the answer key is available for the test data then SenseClusters can provide a completely automated evaluation of a clustering solution obtained by an experiment.

Before we can perform evaluation, each of the generated clusters should be assigned one of the senses. That is, once we have the contexts grouped in k clusters, we try to associate each cluster with one of the n distinct senses present in the answer key such that the number of contexts present in the true or expected clusters is maximized. Table 2 shows a representation of the problem for a dataset containing contexts for three different people with the same name *George Miller* - a movie director, a professor and a congressman. In this representation the clusters are along the rows and the distinct senses are by the columns and the goal is to re-arrange the rows and columns such that the sum of the values along the diagonal is maximized. The row and column with label *T* provide the total of row or column respectively. This representation is often referred to as the *confusion matrix*. For Table 2 the re-arrangement that would satisfy our criterion of maximizing the sum of values along the diagonal is shown in Table 3.

The simplest way to automate this re-arrangement so as to satisfy the criterion is to use the brute-force method - listing all possible sets of associations between the clusters and the senses and choosing the one that maximizes the correct grouping of contexts. However as number of clusters (k) and number of sense (n) increase, the brute-force method quickly becomes very inefficient. Thus we needed some solution other than brute-force which would be easily scalable. We decided on

Table 3: Rearranged Sense Assignment Matrix (3-way)

	Director	**Congressman**	**Professor**	**T**
C0	38	42	0	80
C1	2	86	0	88
C2	17	89	12	118
T	57	217	12	

using one of the most ingenious solutions to the classical assignment problem proposed by Munkres [1957]. Although originally proposed for the assignment problem - assigning n jobs to m workers such as to minimize the total cost - we can easily use this algorithm for our purpose because the requirement of both the tasks - to optimize (either minimize or maximize) assignment of one set to the other - is comparable.

Typically, this task is performed over a matrix - rows representing the jobs or in our case the senses/entities and the columns representing the workers or in our case clusters. We have extended the original algorithm proposed by Munkres for square matrices to rectangular matrices using a trick - of padding extra zeroes where required and thus transforming a rectangular matrix to a square matrix. We have put together all this in a re-usable and distributable form of Perl Module called Algorithm::Munkres[9] which has been released on CPAN[10].

When the number of created clusters is greater than the number of senses then the number of clusters which are extra have to be discarded. Of all the clusters the clusters which would minimize the effect of this cast away on the overall performance of the system are removed. On the other hand if the number of senses is greater than number of clusters than only few of the senses can be associated with the available clusters and thus the assignment is again geared towards maximizing the performance of the system.

We report evaluation in terms of precision, recall and F-measure (F1). Precision, as shown in the following formula, is the ratio of number of correctly clustered contexts and the number of contexts

[9] http://search.cpan.org/dist/Algorithm-Munkres/

[10] http://www.cpan.org/

that were attempted for clustering.

$$Precision = P = \frac{\#contexts\ correctly\ clustered}{\#contexts\ attempted\ for\ clustering} \tag{11}$$

While the recall measure is formulated as the ratio of number of contexts correctly clustered and the total number of contexts present in the test data.

$$Recall = R = \frac{\#contexts\ correctly\ clustered}{total\ \#contexts\ present\ in\ the\ test\ dataset} \tag{12}$$

We use the F1 variation of the F-measure which is formulated as a harmonic mean of precision and recall.

$$Fmeasure = F1 = \frac{2\ *\ P\ *\ R}{P\ +\ R} \tag{13}$$

For a baseline we use the F-measure obtained by assigning the *majority sense* to all the contexts. In other words, we group all the contexts in the dataset into one cluster and find the best F-measure score that we would get in such uni-cluster scenario. Thus if a dataset contains contexts for an ambiguous word with two senses and if the distribution of the senses is balanced in the dataset then the baseline will be 0.5, but if the distribution is skewed - one of the sense has 30 contexts while the other sense has 70 contexts then the baseline will be 0.7 for such a dataset. As such, this baseline captures the performance one can expect for a dataset without performing any kind of categorization and thus provides the lowest threshold value to measure the effectiveness of any method.

3.6 Cluster Labeling

We have attempted to address the commonly faced problem of identifying the underlying entity that a cluster represents without having to manually examine the cluster contents.

Once the context-vectors are separated into clusters we assign each of these clusters a list of bigrams which we refer to as a label for the cluster. The purpose of assigning these labels is to summarize the contents of the clusters in terms of the most significant words that occur in the cluster and thus

help one identify the actual underlying entity. The labels are classified into two types specifically *Descriptive* and *Discriminating*. The *Descriptive* labels are the top N bigrams and *Discriminating* labels are the top N bigrams which are unique to the cluster. The *Descriptive* labels capture the main concept or entity of the cluster whereas the *Discriminating* clusters try to convey the contents that separates one cluster from another cluster.

Each cluster is composed of various contexts grouped under that cluster by the clustering algorithm; these contexts together form a corpus for our label generation process. Similar to the feature selection process, we identifying the bigrams by ranking them either on their frequency of occurrence or their statistical scores. The top N bigrams are picked as the *Descriptive* bigrams while the top N unique bigrams are picked as *Discriminating* labels.

Although this simple technique is almost always very effective in identifying the underlying entity we would like to clarify that these labels do not consist of well-formed grammatical sentences or phrases but are simply lists of word-pairs.

4 Experimental Data

We use four different genres of experimental data to highlight the different applications and aspects of our methodology. We refer to these four genres as:

1. NameConflate data
2. Email data
3. Word Sense Disambiguation data
4. Web data

Each of these genres is described in detail below.

4.1 NameConflate Data

Name discrimination is about trying to identify and separate the distinct entities that share the same name. To test our methods for this problem we needed appropriate test data, that is, data containing name ambiguity. Although such raw data could be found in abundance, especially on World Wide Web, annotated data was almost non-existent. Creating such data implied a humongous amounts of manual labor. The concept of pseudo/artificial ambiguity came to our aid. Using this concept one can artificially create ambiguity by masking each occurrence of two or more unambiguous words and thus make them look alike. For example, given a corpus one could mask each occurrence of the words, *heaven* and *hell* with *earth* and thus make *heaven* and *hell* "look" alike - as the *earth*! Such pseudo ambiguity has often been used in word sense disambiguation research, although its use has been questionable for two reasons: firstly because the different senses of naturally ambiguous words typically tend to be closely related while the artificially created ambiguous words might not exhibit this property; and secondly given that most words are polysemous the words chosen to create artificial ambiguity cannot be guaranteed to be totally unambiguous themselves unless each occurrence is individually examined. Fortunately artificial ambiguity when applied to the create proper name ambiguity does not suffer, or at least offers a work around for each of these objections. Firstly one can easily control the relatedness/disparity among the underlying entities by choosing

appropriate names. For example, one could choose to mask/conflate the names *Johnny Cash* and *Elvis Presley* to create closely related ambiguity or one could choose to mask/conflate *Johnny Cash* and *Sonia Gandhi* to create a highly disparate ambiguity. The second issue is avoided by the fact that given a proper name one can usually be quite certain about how many entities share the same name, if any.

SenseClusters requires the test data to be in SENSEVAL-2[11] format, which divides the data into a set of contexts. Each of these contexts consists of one occurrence of the ambiguous name/word along with the surrounding words. The size of the surrounding data can range from few words to complete document. The true identity/sense (the key) of the ambiguous name/word in each context is stored in a separate answer tag which facilitates automatic evaluation of methodologies.

To achieve all of this we have crafted a Perl script, nameconflate.pl[12], which takes unambiguous names to be conflated, corpora to be used, the maximum size of each context and creates a SENSEVAL-2 formatted dataset containing the artificially created name-ambiguity for the specified names. We have made this script capable of handling plain text and Gigaword formatted input corpora, the latter being more efficient. The unit used to specify the maximum expected surrounding data is - count of words. Thus if one wishes to create contexts with at most 25 words before and 25 words after the conflated name then the window size of 25 should be specified. One of the latest enhancements to this script has been its capability to conflate variants of the unambiguous names instead of just one form of the names. For example, one can now specify to conflate variants of the string *Johnny Cash* such as *Mr. Cash, J. Cash, JRC* etc. One of the motivations behind adding this functionality comes from the following observation - the long form of a name is typically spelled out only once at the start of the writing and later on a shorter version of the name or shorter variants are used. By including such variants we test the methodology for both one-to-many (one name referring to multiple entities) and many-to-one (multiple names referring to same entity) both. To support the name-variants functionality we make use of Perl's regular-expressions.

For our experimental datasets the window option of the nameconflate script was set to 50, which implies that for any context in any of the datasets the maximum number of words to the left or to

[11] http://www.senseval.org/

[12] http://www.d.umn.edu/∼kulka020/kanaghaName.html

the right of the conflated name can be 50. Thus the maximum size of any context can be 101 words (conflated-word + 50 left words + 50 right words).

The source of the raw data that we use for creating our name-conflated datasets is the second edition of the English Gigaword Corpus[13] as distributed by the Linguistic Data Consortium. This consists of newswire text from five international news services, namely:

- Agence France Press English Service (AFE)
- Associated Press Worldstream (APW)
- Central News Agency of Taiwan (CNA)
- The New York Times (NYT)
- The Xinhua News Agency (XIN)

Here we use The New York Times' portion of the corpus that appeared from January 2002 to December 2004 and which consists of approximately one billion words.

We have created 13 name-conflated datasets, each of which can be categorized based on the following two criterion:

1. the number of names that were conflated together,
2. the disparity among the conflated names.

Varying the number of conflated names simulates the cases in which more than two entities share the same name and thus tests the methodology for not just 2-dimensional but for n-dimensional scenarios.

The categorization of the 13 datasets based on the first criterion, the number of conflated names, is given in Table 4.

Experimenting with varying degrees of disparity among the conflated names examines the cases where the underlying entities for an ambiguous name have either acute or obtuse differences. The

[13] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2005T12

Table 4: Categorization of the 13 name-conflated datasets based on the number of conflated names

N-way	Conflated Names
2-way	Bill Gates & Jason Kidd
	Bill Gates & Steve Jobs
	Serena Williams & Richard Boucher
	Mark McGwire & Sammy Sosa
3-way	Bill Gates & Jason Kidd & Dalai Lama
	Bill Gates & Steve Jobs & Larry Ellison
	Serena Williams & Richard Boucher & Michael D. Eisner
	Mark McGwire & Sammy Sosa & Randy Johnson
4-way	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless
	Bill Gates & Steve Jobs & Larry Ellison & Paul Allen
	Mark McGwire & Sammy Sosa & Randy Johnson & Mike Piazza
5-way	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless & Beatles
6-way	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless & Beatles & Peter Jackson

dataset created by conflating the names *Bill Gates*, founder of Microsoft Corporation and *Jason Kidd*, a National Basketball Association (NBA) player is an example of highly disparate dataset because the contexts in which each of them are discussed are almost always mutually exclusive. The opposite case is of entities which have many overlapping characteristics. For example, the dataset created by conflating *Bill Gates* and *Steve Jobs*, co-founder of Apple Computer, Inc. and Pixar Animation Studios would be a good representative of this case. With these two cases of name ambiguity - underlying entities highly unrelated or related - we aim to capture and simulate the possibilities of disparity that can be found in real data.

The categorization of the 13 datasets based on the second criterion, the disparity among the conflated names, is given in Table 5.

Table 6 gives tabularized details for each NameConflate dataset such as the number of names con-

Table 5: Categorization of the 13 name-conflated datasets based on the disparity among the conflated names

Disparity	**Conflated Names**
Distinct	Bill Gates & Jason Kidd
	Serena Williams & Richard Boucher
	Bill Gates & Jason Kidd & Dalai Lama
	Serena Williams & Richard Boucher & Michael D. Eisner
	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless
	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless & Beatles
	Bill Gates & Jason Kidd & Dalai Lama & Verizon Wireless & Beatles & Peter Jackson
Subtle	Bill Gates & Steve Jobs
	Bill Gates & Steve Jobs & Larry Ellison
	Bill Gates & Steve Jobs & Larry Ellison & Paul Allen
	Mark McGwire & Sammy Sosa
	Mark McGwire & Sammy Sosa & Randy Johnson
	Mark McGwire & Sammy Sosa & Randy Johnson & Mike Piazza

flated (N-way: first column), the names that were conflated along with their distribution (N) and percentage distribution (P) in second column and the total number of contexts per dataset (T) in the last column. As we can see from the Table 6, the distribution of names in any given dataset is more or less naturally balanced, with the exception of names like *Dalai Lama* and *Peter Jackson* which have significantly fewer number of contexts (295 and 245 respectively) than the other names in the dataset. Although, we could have artificially balanced the distribution of names for the datasets, we have not done so. The reason being that one cannot always expect balanced ambiguity in real world data. Secondly we have also tried to include some variety in the total number of contexts. For example, within the 2-way category, the total number of contexts vary from 400+ to 2000+. The purpose behind doing this is to test the effect of varying population, if any.

Table 6: Details about NameConflate datasets

N-way	Conflated Names (N)(P)	T
2-way	Serena Williams (230)(46.94) & Richard Boucher (260)(53.06)	490
	Bill Gates (372)(50.34) & Steve Jobs (367)(49.66)	739
	Bill Gates (636)(44.57) & Jason Kidd (791)(55.43)	1427
	Mark McGwire (1128)(53.66) & Sammy Sosa (974)(46.34)	2102
3-way	Serena Williams (230)(31.81) & Richard Boucher (260)(35.96) & Michael D. Eisner (233)(32.23)	723
	Bill Gates (372)(34.03) & Steve Jobs (367)(33.58) & Larry Ellison (354)(32.39)	1093
	Bill Gates (636)(36.94) & Jason Kidd (791)(45.93) & Dalai Lama (295)(17.13)	1722
	Mark McGwire (1128)(30.70) & Sammy Sosa (974)(26.51) & Randy Johnson (1572)(42.79)	3674
4-way	Bill Gates (372)(24.20) & Steve Jobs (367)(23.88) & Larry Ellison (354)(23.03) & Paul Allen (444)(28.89)	1537
	Bill Gates (636)(29.44) & Jason Kidd (791)(36.62) & Dalai Lama (295)(13.66) & Verizon Wireless (438)(20.28)	2160
	Mark McGwire (1128)(22.52) & Sammy Sosa (974)(19.45) & Randy Johnson (1572)(31.39) & Mike Piazza (1334)(26.64)	5008
5-way	Bill Gates (636)(23.04) & Jason Kidd (791)(28.66) & Dalai Lama (295)(10.69) & Verizon Wireless (438)(15.87) & Beatles (600)(21.74)	2760
6-way	Bill Gates (636)(21.16) & Jason Kidd (791)(26.32) & Dalai Lama (295)(9.82) & Verizon Wireless (438)(14.58) & Beatles (600)(19.97) & Peter Jackson (245)(8.15)	3005

4.2 Email Data

To generate datasets for email clustering, or in general to create *headless* datasets, we wrote a utility in Perl which we refer to as *Text2headless*. This script creates SENSEVAL-2 formatted dataset from

a set of input file. As the name of the utility implies, it is used to generate contexts without any specific word/name to be disambiguated. This utility transforms the contents of each provided input file into a SENSEVAL-2 formatted context. Given a set of files and a set of directories to this utility it would read each of the specified files and would recursively traverse the directories to find files and read them. While reading the file contents it removes all the newline characters and empty/blank lines. If the file was directly specified, then the file-name is stored as the answer key, and if directory was specified, then the directory name is stored as the answer key for every file found under that directory. Given its functionality, one can see that this utility can also be used for generating datasets for document clustering experiments quite easily.

For email clustering experiments we have created 13 datasets with the *Text2headless* script described, using 20 Newsgroups dataset[14] as the source data. The 20 Newsgroup dataset is a collection of approximately 20,000 USENET postings manually categorized into 20 different groups such as *comp.graphics*, *rec.sport.hockey* etc.

Using the text2headless script we combined all the emails from two or more categories into one SENSEVAL-2 formatted dataset where each email is transformed into a context and its original category is retained in a separate answer tag. For example, every email under the categories *comp.graphics* and *talk.politics.mideast* was transformed into a context, the category was stored away into an answer tag (for evaluation purposes) and all these contexts were put together in SENSEVAL-2 format to give the first dataset shown in Table 7.

Again, each of the 13 datasets can be categorized based on:

1. the number of newsgroups that were combined,
2. the disparity among the newsgroups that were combined.

The motivation for selecting such categorization is the same as that described for the nameconflate dataset categorization.

The categorization of the 13 email datasets based on the first criterion, and the number of newsgroups combined, is given in Table 7.

[14] http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 7: Categorization of the 13 email datasets based on the number of newsgroups combined

N-way	Combined Newsgroups
2-way	comp.graphics & talk.politics.mideast
	rec.sport.hockey & soc.religion.christian
	sci.crypt & sci.electronics
	sci.electronics & soc.religion.christian
	talk.politics.guns & talk.politics.mideast
3-way	rec.sport.hockey & soc.religion.christian & misc.forsale
	sci.crypt & sci.electronics & sci.med
	sci.electronics & soc.religion.christian & talk.politics.guns
	talk.politics.guns & talk.politics.mideast & talk.politics.misc
4-way	rec.sport.hockey & soc.religion.christian & misc.forsale & talk.politics.guns
	sci.crypt & sci.electronics & sci.med & sci.space
	sci.electronics & soc.religion.christian & talk.politics.guns & alt.atheism
	talk.politics.guns & talk.politics.mideast & talk.politics.misc & talk.religion.misc

The categorization of the 13 email datasets based on the second criterion, and the disparity among the combined newsgroups, is given in Table 8.

The columns in Table 9 have the same interpretation as that described earlier for Table 6. As we can see from the Table 9, almost all of the newsgroups used to create the datasets have 900+ emails, except for talk.politics.misc (775 emails), alt.atheism (800 emails) and talk.religion.misc (628 emails).

4.3 Word Sense Data

To test our methods for the more generic task of word sense discrimination we experiment with datasets associated with four ambiguous words:

1. *line*

Table 8: Categorization of the 13 email datasets based on the disparity among the combined newsgroups

Disparity	**Combined Newsgroups**
Distinct	comp.graphics & talk.politics.mideast
	rec.sport.hockey & soc.religion.christian
	rec.sport.hockey & soc.religion.christian & misc.forsale
	rec.sport.hockey & soc.religion.christian & misc.forsale & talk.politics.guns
	sci.electronics & soc.religion.christian
	sci.electronics & soc.religion.christian & talk.politics.guns
	sci.electronics & soc.religion.christian & talk.politics.guns & alt.atheism
Subtle	sci.crypt & sci.electronics
	sci.crypt & sci.electronics & sci.med
	sci.crypt & sci.electronics & sci.med & sci.space
	talk.politics.guns & talk.politics.mideast
	talk.politics.guns & talk.politics.mideast & talk.politics.misc
	talk.politics.guns & talk.politics.mideast & talk.politics.misc & talk.religion.misc

2. *hard*
3. *serve*
4. *interest*

We have used the cleaned and SENSEVAL-2 formatted versions of these datasets distributed by Dr. Ted Pedersen[15].

The dataset for the noun *line* was created and introduced by Leacock et al. [1993]. It contains 4,146 contexts, each of which consists of the sentence in which the word *line* occurred along with one or more prior sentences. Each of these contexts were manually tagged with one of the six WordNet senses of *line*: *product, phone, text, cord, division* and *formation*.

[15]http://www.umn.edu/home/tpederse/data.html

Table 9: Details about the email datasets

N-way	Combined Newsgroups (N)(P)	T
2-way	talk.politics.guns(364)(49.19) & talk.politics.mideast(376)(50.81)	740
	comp.graphics(389)(50.85) & talk.politics.mideast(376)(49.15)	765
	sci.crypt(396)(50.19) & sci.electronics(393)(49.81)	789
	sci.electronics(393)(49.68) & soc.religion.christian(398)(50.32)	791
	rec.sport.hockey(399)(50.06) & soc.religion.christian(398)(49.94)	797
3-way	talk.politics.guns(364)(34.67) & talk.politics.mideast(376)(35.81) & talk.politics.misc(310)(29.52)	1050
	sci.electronics(393)(34.02) & soc.religion.christian(398)(34.46) & talk.politics.guns(364)(31.52)	1155
	sci.crypt(396)(33.42) & sci.electronics(393)(33.16) & sci.med(396)(33.42)	1185
	rec.sport.hockey(399)(33.61) & soc.religion.christian(398)(33.53) & misc.forsale(390)(32.86)	1187
4-way	talk.politics.guns(364)(27.98) & talk.politics.mideast(376)(28.90) & talk.politics.misc(310)(23.83) & talk.religion.misc(251)(19.29)	1301
	sci.electronics(393)(26.64) & soc.religion.christian(398)(26.98) & talk.politics.guns(364)(24.68) & alt.atheism(320)(21.69)	1475
	rec.sport.hockey(399)(25.73) & soc.religion.christian(398)(25.66) & misc.forsale(390)(25.15) & talk.politics.guns(364)(23.47)	1551
	sci.crypt(396)(25.08) & sci.electronics(393)(24.89) & sci.med(396)(25.08) & sci.space(394)(24.95)	1579

The datasets for the adjective *hard* and for the verb *serve* were created by Leacock et al. [1998]. The *hard* dataset consists of 4333 contexts and each of the contexts was manually tagged with one of the three WordNet senses: *not easy, not soft(metaphoric)* and *not soft(physical)*. While the *serve* dataset consists of 4,378 contexts with 4 WordNet senses: *supply with food, hold an office, function as something* and *provide a service*.

The noun - *interest* dataset, created by Bruce and Wiebe [1994] consists of 2,368 contexts, each manually tagged with one of six senses: *readiness to give attention, quality of causing attention to be given to, activity etc. that one gives attention to, advantage, a share in a company or business* and *money paid for the use of money*.

The details about each of these datasets are given in Table 10 where the first column specifies the granularity of the ambiguity (N-way), the second column gives the ambiguous word, the third column lists the various senses of the word that are present in the dataset, the last but one column gives the distribution of each sense in terms of counts (N) and percentages (P), and the last column provides the total number of contexts in the datasets.

4.4 Web Data

Our final genre of datasets is created from the data that we have collected from the World Wide Web (WWW). We selected the five ambiguous names given in Table 11 and collected the *content data*, that is the actual data without the meta-data, from the first 50 search results given by Google. Additionally we also crawled each page one level down to include *content data* found on the html pages which were present in the same web-domain as the parent page. We have used an exact and complete match for the Google search of the names. More specifically, we have used double-quoted strings of names for the search. For example, we used *"George Miller"* as the search string for the name *George Miller*, which guarantees that each of the documents returned by Google for this search string contains at least one occurrence of the ambiguous name in the specified format.

Once this raw content data was collected we manually cleaned the data to remove strings like "[IMAGE]" and to discard data which consisted merely of a list of names without any meaningful surrounding content.

Next we chopped the data into contexts of at most 101 words each containing the ambiguous name and the surrounding content pivoted around the name. Note that here along with creating contexts for exact matches of the ambiguous names we have also created contexts based on different variations of the name. For instance, we have created contexts for strings like "Mr. Miller", "Dr. Miller", "G. Miller" etc. After such contexts, each containing some form of the ambiguous names are cre-

Table 10: Details about the word sense disambiguation datasets

N-way	Word	Senses	N (P)	T
3-way	HARD	not soft (physical)	376 (8.68)	4333
		not soft (metaphoric)	502 (11.59)	
		not easy	3455 (79.74)	
4-way	SERVE	hold an office	439 (10.03)	4378
		supply with food	853 (19.48)	
		provide a service	1272 (29.05)	
		function as something	1814 (41.43)	
6-way	LINE	text	349 (8.42)	4146
		product	373 (9.00)	
		phone	376 (9.07)	
		formation	404 (9.74)	
		cord	429 (10.35)	
		division	2218 (53.50)	
	INTEREST	quality of causing attention to be given to	11 (0.46)	2368
		activity etc. that one gives attention to	66 (2.79)	
		advantage	178 (7.52)	
		readiness to give attention	361 (15.24)	
		a share in a company or business	500 (21.11)	
		money paid for the use of money	1252 (52.87)	

ated we proceed to manually annotate each of the context. We annotate each context based on the underlying entity that is being discussed in it. The list of entities for each of the five names is given under the column titled *Entities* in the Table 11. The next column provides the distribution of each entity in a dataset (*N*) and the numbers in the brackets (*P*) are again distribution but now in percentage values. The last column *T* gives the total number of contexts present in each dataset. Given the dynamic nature of WWW and thus of the search results, we have documented the dates on which these searches were made. These dates are given in Table 11 right below each name. As we see,

the distribution of each of the datasets is very skewed - one of the entities in each dataset is very dominant, that is, has higher web-presence and higher Google ranking than all the other entities that share the same name.

Table 11: Details about the Web datasets

N-way	Name	Entities	N (P)	T
2-way	SARAH CONNOR	Fictional character from the movie - Terminator	41 (27.33)	
	(May 29, 2006)	German pop singer	109 (72.67)	150
	RICHARD ALSTON	Australian Senator	71 (28.74)	
	(May 4, 2006)	British Choreographer	176 (71.26)	247
3-way	GEORGE MILLER	Professor from Princeton University	12 (4.20)	
	(May 1, 2006)	Australian Movie Director	57 (19.93)	
		Congressman	217 (75.87)	286
4-way	TED PEDERSEN	TV series writer	10 (3.00)	
	(May 1, 2006)	Son of a famous Captain	25 (7.50)	
		Book Writer	43 (12.92)	
		Professor at University of Minnesota	255 (76.58)	333
	MICHAEL COLLINS	NASA Astronaut	17 (4.74)	
	(May 5, 2006)	Professor at University of Wisconsin Madison	32 (8.91)	
		Professor from MIT	41 (11.42)	
		Irish freedom fighter	269 (74.93)	359

Table 12: List of different parameters and settings that we experiment with.

Parameter	**Parameter Settings**
Context-representation and Feature type	order1 & unigrams
	order1 & bigrams
	order2 & bigrams
Test Scope	complete
	7 words
SVD	with
	without
Criterion function	I2
	E1
	H2
Clustering algorithm	agglo
	rbr
Cluster stopping measure	all
	none

5 Experimental Results

SenseClusters provides various user configurable parameters and parameter settings for lexical features, context representation, clustering, cluster stopping etc., described in Section 3. In this thesis we experiment with a few of these variations which are summarized in Table 12. The reasons and hypotheses behind investigating these parameters or combinations of parameters are described in the discussions to follow.

Apart from the variations listed in Table 12 we use the following experimental settings across all the experiments. Each of the following settings try to improve the feature set by (hopefully) pruning noisy features.

- OR Stoplisting (described in Section 3.1)

Table 13: Table summarizing the number of experiments conducted.

Genre	Sub-genre	#datasets	#parameter-settings	Total
NameConflate Data	**Distinct**	7	144	1008
	Subtle	6	144	864
Email Data	**Distinct**	7	72	504
	Subtle	6	72	432
Word Sense Disambiguation Data	-	4	144	576
Web Data	-	5	144	720
			Total	4104

- Frequency cutoff of 2
- Log-likelihood as test of association with score cutoff of 3.841 (95% certainty)

We try each of the parameter settings listed in Table 12 with each of the datasets under the genres - nameconflate, web and WSD. The number of experiments that we conducted is summarized in Table 13. For the datasets from email genre we experiment with each of the parameter settings from Table 12 except for the two variations on the test-scope option. Since the email datasets are of type *headless*, that is, since the contexts do not contain any *head* word, restricting the scope around the *head* word for the test contexts is not possible and thus the complete context is always used to create a representation for each email context.

With these numerous experiments we wish to address various questions and issues, each of which we discuss in the following sub-sections. In most of the issues below the goal is to compare two settings, that is, to compare the performance based on some metric, obtained while using the two given settings and to find if we can see any patterns emerging. To achieve this we use *scatter plots* to observe the association between the two settings. We plot one setting along the x-axis and the second along the y-axis. If most of the points in a scatter plot are along the diagonal then we can conclude that there isn't any difference in the performance when using either of the settings. However if majority of the points are concentrated in the bottom-right corner of the scatter plot then

it indicates that the setting along the x-axis provides a better performance as compared to the setting along the y-axis. Similarly if majority of the points are concentrated in the upper-left corner then one can conclude that the setting along the y-axis performs better as compared to the x-axis setting.

5.1 Order1 and unigrams versus Order2 and bigrams

The aim of this comparison between first and second order context representation is to re-confirm and generalize the findings of Purandare and Pedersen [2004]. For WSD data, they observe that in spite of its simple representation the first order context representation is able to capture enough information if large enough feature selection data is available. However, the merit of second order representation can be seen when a relatively small feature selection data is available.

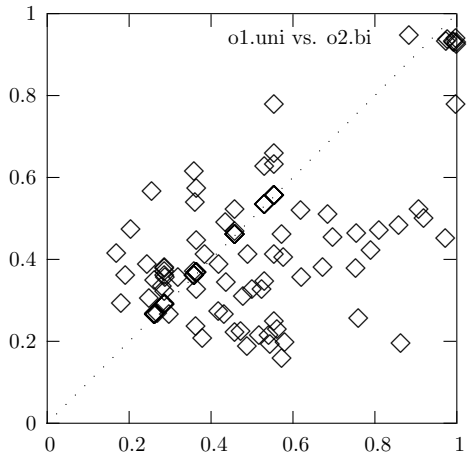

Figure 9: Comparison between order1 & unigram vs. order2 & bigram, (F-measure), (Gap), (Nameconflate-Distinct)

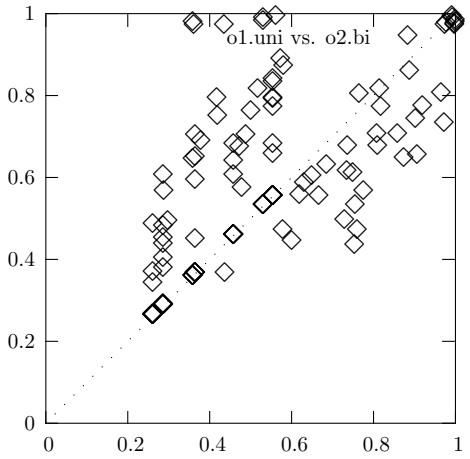

Figure 10: Comparison between order1 & unigram vs. order2 & bigram, (purity), (Gap), (Nameconflate-Distinct)

Figure 9 shows our results comparing order1 and unigram features with order2 and bigram features for the distinct category of nameconflate dataset. Please note that henceforth in this subsection order1 implies order1 with unigram features and order2 implies order2 with bigram features. The x-axis corresponds to the results obtained using the first order context representation and the y-axis corresponds to the results obtained using the second order context representation. The performance

metric used in this plot is the F-measure and the Adapted Gap Statistic was used to predict the number of clusters to be generated. The Figure 10 presents similar comparison using purity (described in Section 2.4) as the performance metric.

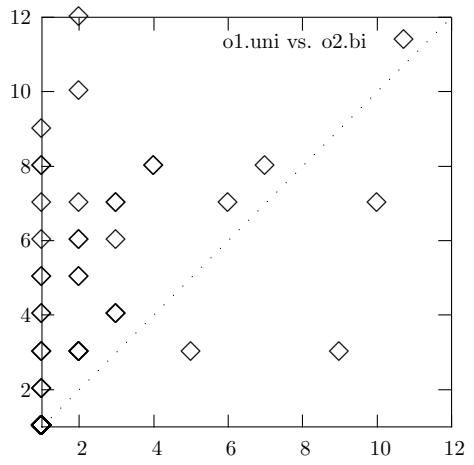

Figure 11: Comparison between order1 & unigram vs. order2 & bigram, (#clusters predicted by Adapted Gap Statistic), (Nameconflate-Distinct)

The first impression that one gets from these plots is that they exhibit contradicting patterns - that order1 performs better than order2 when evaluating based on F-measure while order2 is better than order1 when evaluating using purity. However, one needs to be careful here to note that if we look at the Figure 11 we can see that the Adapted Gap Statistic measure has often over-predicted the number of clusters when using order2 context representation. Consequently the clustering algorithm divides the data into clusters with very fine-grained distinctions and as a result the purity of such clusters typically is higher than clusters with coarse-grained distinctions however the F-measure suffers due to the extra fragmentation of the contexts.

Now if we look at similar plots (Figures 12 and 13) for the datasets from the nameconflate-subtle category then another interesting conclusion can be drawn. Contrary to the previous observation of order1 being superior to order2, here we see that in majority of the cases the results obtained using

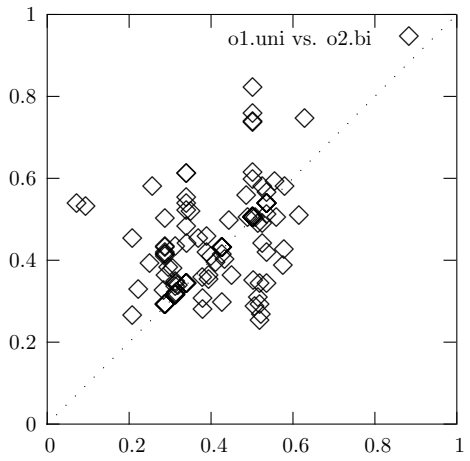

Figure 12: Comparison between order1 & unigram vs. order2 & bigram, (F-measure), (Gap), (Nameconflate-Subtle)

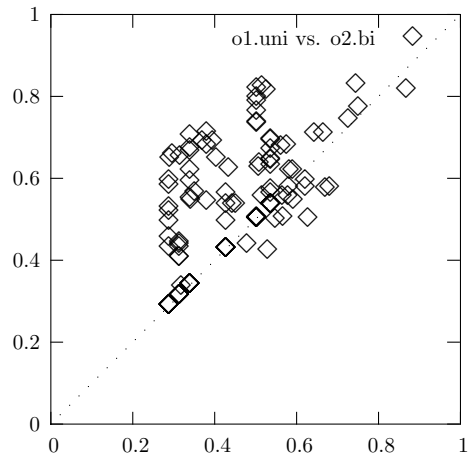

Figure 13: Comparison between order1 & unigram vs. order2 & bigram, (purity), (Gap), (Nameconflate-Subtle)

order2 representation are better than the results obtained using order1 representation.

We hypothesize that the reason for this disparity is the stark difference in the definitions and thus properties of the datasets from the two categories of distinct and subtle. As described in Section 4, the datasets under the subtle category of nameconflate were created by conflating highly related names. For example, the names *Bill Gates* and *Paul Allen*, both founders of Microsoft Corporation, were conflated to create one such dataset. Typically finding discriminating features in such datasets is a non-trivial task, especially for first order representation which relies on surface-features. However, the capability of second order context representation to capture indirect relationships between words and thus contexts is able to bring out discriminating features even from such subtle datasets. On the other hand, with distinct type of datasets, rich features are directly available at the surface level and thus trying to find extra information through indirect relations tends to bring in unwanted noise. Given these observations, we conclude that if one has large data with clearly distinct senses then order1 context representation is the best option whereas for small data with fine-grained sense distinctions order2 is a better choice. As such, order1 and order2 together provide a complementary pair of context representation.

These trends and observations about the appropriate applicability of these two context representations form guidelines for our future experiments and applications. We conclude from our results that the observations by Purandare and Pedersen [2004] are generally applicable and thus can be re-confirmed.

It is interesting to note that although we have observed a comparable performance by order1 and order2 in Kulkarni and Pedersen [2005a], this might be due to the comparatively smaller dataset and limited number of senses per ambiguous name.

5.2 Without SVD versus With SVD

There were multiple motivations for investigating the issue of whether Singular Value Decomposition (SVD) imparts any strengths to our approach. Firstly, until now we had not been able to settle this issue conclusively based on our previous experiments Pedersen et al. [2005]. Secondly, Schütze [1998] reports of observing positive effects of using SVD in his algorithm of *context-group discrimination*. Thirdly, Wiemer-Hastings [1999] reports about the ability of SVD to "do more in less" while using Latent Semantic Analysis (LSA). More specifically, he reports that although the results of LSA with SVD are not significantly better than those without SVD, there is a great computational advantage to operating on SVD reduced matrices rather than the humongous unreduced matrices.

Figures 14, 15, 16 and 17 show the scatter plots comparing the performance of Email-Distinct datasets and WSD datasets, without and with SVD, once with F-measure as the metric and next with cluster purity values as the metric. The cluster stopping measure of PK3 was used in these experiments to predict the appropriate number of clusters for each dataset.

We can clearly see from these plots that our methods perform better without SVD. Although these plots show results only for the datasets from the email-distinct and WSD genre, this trend was observed for datasets from the other genres too. The only exceptions were datasets from the email-subtle and nameconflate-subtle genre. In case of these datasets the performance of both the settings - with and without SVD - was comparable, neither was a clear winner.

Given the closeness of our work to that of Schütze one would expect comparable findings, but we think there are several possible explanations for the observed difference between our findings

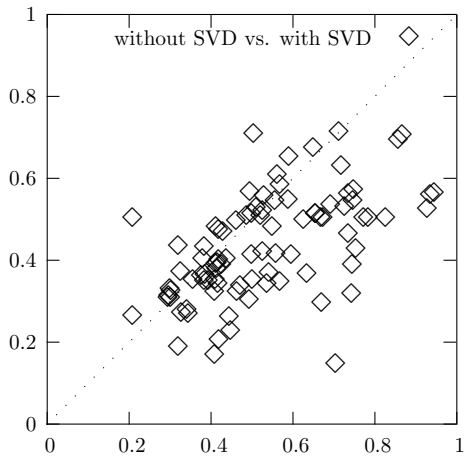

Figure 14: Comparison between without and with SVD, (F-measure), (PK3), (Email-Distinct)

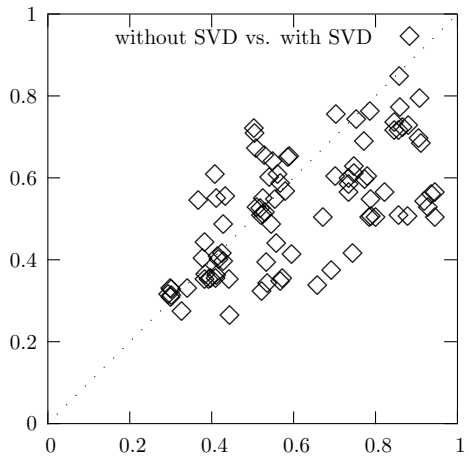

Figure 15: Comparison between without and with SVD, (purity), (PK3), (Email-Distinct)

and his findings with respect to SVD. Firstly, almost all the datasets in Schütze's study contain ambiguous words with only two senses. More importantly, he uses a separate and a bigger feature selection corpus (*training corpus*) while we use the test data as the feature selection corpus. As a result, we think that the model generated by our approach is already well-pruned compared to the model created by Schütze from a different corpus which might have redundancy and noise, which is eliminated by SVD, thus boosting the performance. As noted by Wiemer-Hastings [1999] LSA like techniques tend to be more effective with longer text than with shorter, and we typically deal with shorter contexts rather than longer, and this might be another reason for the behavior we observe.

5.3 Repeated Bisection versus Agglomerative Clustering

Zhao and Karypis [2002] recommend a hybrid clustering algorithm, Repeated Bisection, over partitional and agglomerative clustering algorithms, and in this comparison we seek to verify this for our domain and data.

Figures 18, 19 and 20 show our results for the datasets from the Web genre. The cluster stopping measure of PK2 was used in these experiments.

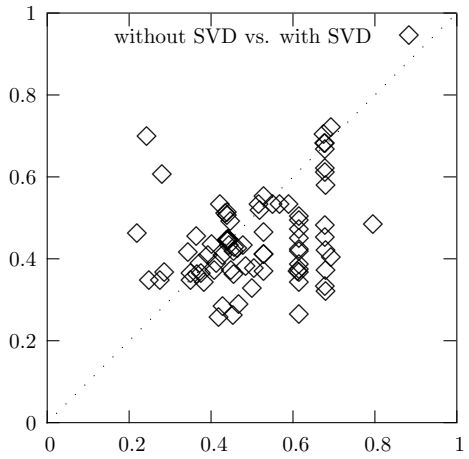

Figure 16: Comparison between without and with SVD, (F-measure), (PK3), (WSD)

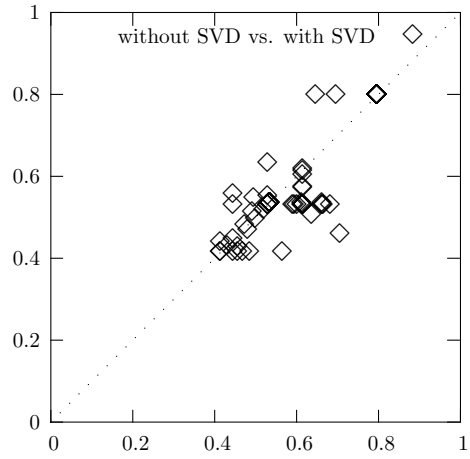

Figure 17: Comparison between without and with SVD, (purity), (PK3), (WSD)

We can see from the F-measure and purity plots that most of the results are concentrated around the diagonal, and thus our methods seem to be performing equally well with both the clustering algorithms that we experimented with.

One exception that we observe is shown in Figures 21 and 22. These results are for the datasets from the nameconflate-subtle category when the cluster stopping measure of PK2 was used. One can arguably claim from these plots, especially the F-measure plot, that the repeated bisections (rbr) clustering algorithm is able to perform better with these datasets than agglomerative clustering does. Zhao and Karypis [2002] report that the primary strength of repeated bisection over agglomerative algorithms is its ability to avoid the merging errors in the early stages that are frequently the cause of poor performance of the agglomerative methods. We think that this might be the reason for the trend that we observe for the nameconflate-subtle datasets. Given the nature of the subtle datasets and the non-trivial task of discriminating them, minor errors in the early stages of clustering can have considerable impact on the overall performance.

Here again we reconfirm and generalize the findings of Purandare and Pedersen [2004] that an improvement can be gained by using the repeated bisections method of clustering when a small amount of data is available.

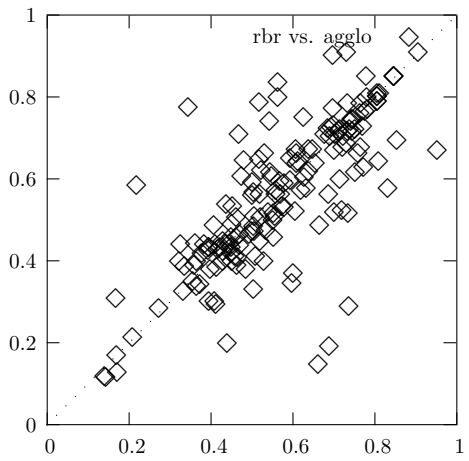

Figure 18: Comparison between Repeated Bisection and Agglomerative Clustering, (F-measure), (PK2), (Web)

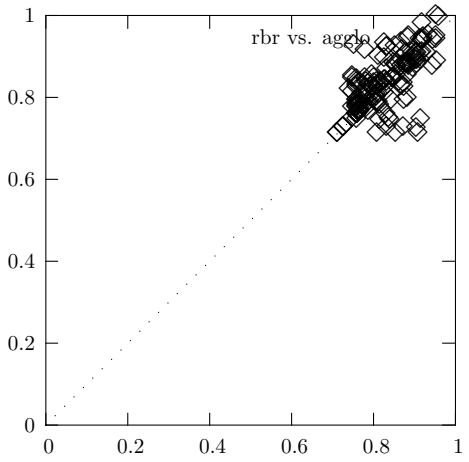

Figure 19: Comparison between Repeated Bisection and Agglomerative Clustering, (purity), (PK2), (Web)

5.4 Full test-scope versus Limited test-scope

We have previously observed in Pedersen et al. [2005] that setting the scope parameters in inversely proportional relation performed well. More specifically, we found that using small test-scope and large train-scope and vice-versa resulted into better performance. However from our current set of experiments it is not clear if we can report any such pattern emerging from the results. Figures 23 and 24 show comparison between the results obtained when using full test-scope (x-axis) versus limited test-scope (y-axis) of 7 words on both sides of the head word. A complete training-scope has been used in both the experiments. Also, the number of clusters was not predicted by any cluster stopping measure but was set by us to an appropriate value.

The only exception to this behavior of comparable performance between full and limited test-scope is the nameconflate-distinct category. It appears from Figures 25 and 26 that for the datasets from this category, full test-scope along with full training-scope performs better than limited test-scope of 7 words and full training-scope.

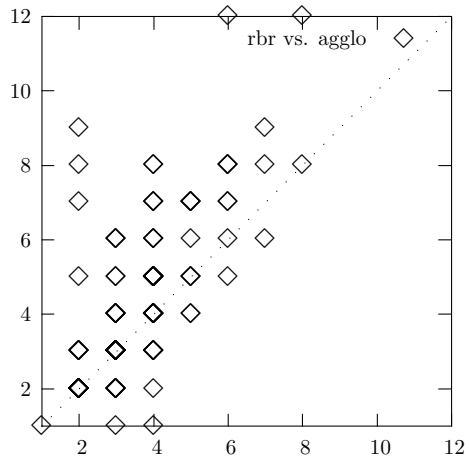

Figure 20: Comparison between Repeated Bisection and Agglomerative Clustering, (#clusters predicted by PK2), (Web)

5.5 Distinct vs. Subtle

We realize that the degree of similarity or dissimilarity between real contexts can be anywhere between fine-grained and coarse-grained. To simulate such properties of the real data as closely as possible we create datasets by conflating either related or disparate names or email-groups (described in Section 4.1).

Figure 27 shows a scatter plot comparing the performance of datasets from the nameconflate-distinct category with their respective baselines. The baseline, as described in Section 3.5, for each dataset is the majority sense present in the dataset. No cluster stopping measure was used in these experiments. That is, the appropriate number of clusters was specified. Although we see a few points below the diagonal which correspond to results that were below the baseline, we can clearly see that majority of the points are above the diagonal, and each of these points represents a result that outperformed the baseline. It is also heartening to see a few experiments achieve F-measure score of 1, that is, 100% precision and 100% recall. Most of the datasets under this category are balanced and thus high majority sense or baseline is a indicator of small number of senses. Given this, we can see another pattern emerging from this plot - datasets with small number of underlying entities

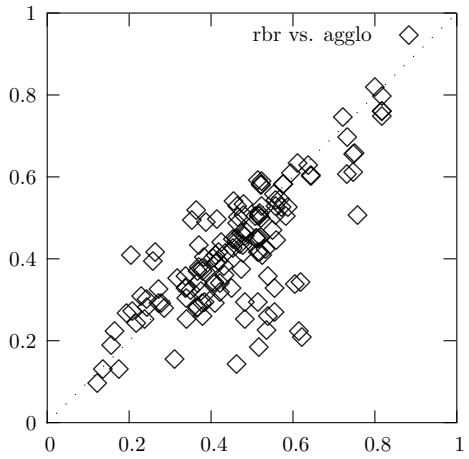

Figure 21: Comparison between Repeated Bisection and Agglomerative Clustering (F-measure), (PK2), (Nameconflate-Subtle)

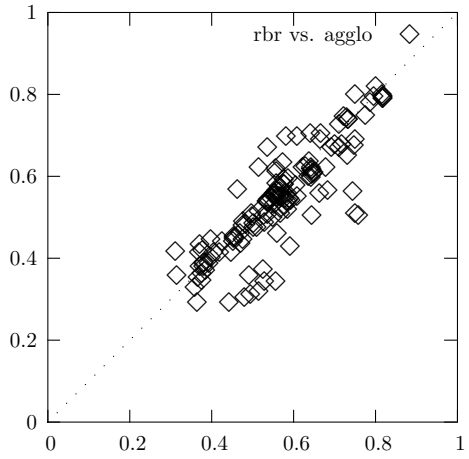

Figure 22: Comparison between Repeated Bisection and Agglomerative Clustering (purity), (PK2), (Nameconflate-Subtle)

(names/groups) perform better than the datasets with more number of entities - shown by the upper two towers.

Figure 28 compares the performances of datasets from the subtle category of the nameconflate data with their baselines. A majority of the points are again above the diagonal and thus above the baseline. However the highest F-measure value obtained is not much above 0.85 and most of the points exhibit F-measure scores of less than or equal to 0.6

Figure 29 and 30 compare the distinct and subtle datasets from the email category. Here again we see that although our methods perform better than the baseline in both the distinct and subtle cases, discriminating subtle senses is a non-trivial task.

5.6 PK2 vs. PK3 vs. Adapted Gap Statistic

In this Section we specifically look at the performance of each of our cluster stopping measures. Note that we do not include the PK1 cluster stopping measure in this comparison. Although it is a valid cluster stopping measure, we find its dependency on the user for the threshold value to be

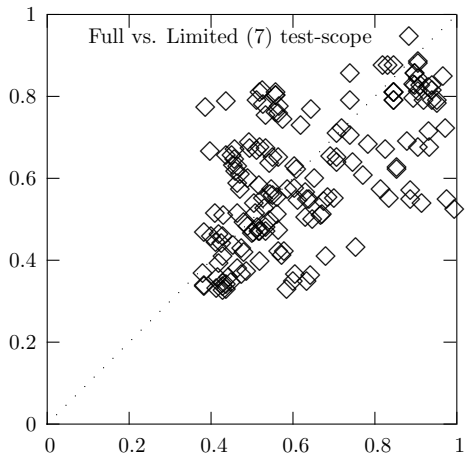

Figure 23: Comparison between Full and Limited (7) test-scope, (F-measure), (Manual), (Web)

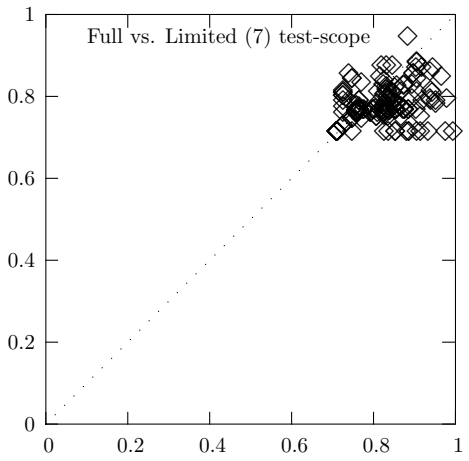

Figure 24: Comparison between Full and Limited (7) test-scope, (purity), (Manual), (Web)

against our primary goal of making the decision completely user-independent.

Here we perform an indirect as well as direct evaluation of the predictions made by each of the measures. For indirect evaluation we compare the F-measures obtained for an experimental setup once when x cluster stopping measure was used to predict the number of clusters and once when y cluster stopping measure was used to predict the number of clusters. The underlying assumption in this indirect evaluation is that if the prediction about the number of clusters by a measure is accurate then the the resulting clustering and thus the F-measure is better than that given by a poor prediction.

We perform this indirect comparison for each pairwise combination of the three cluster stopping measures - $PK2$, $PK3$ and $Adapted\ Gap\ Statistic$:

1. $PK2$ vs. Gap
2. $PK3$ vs. Gap
3. $PK2$ vs. $PK3$

In the direct evaluation we compare each prediction by a measure with the *given* number of the

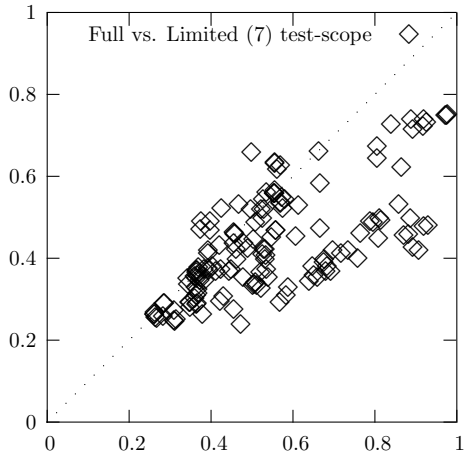

Figure 25: Comparison between Full and Limited (7) test-scope, (F-measure), (Manual), (Nameconflate-Distinct)

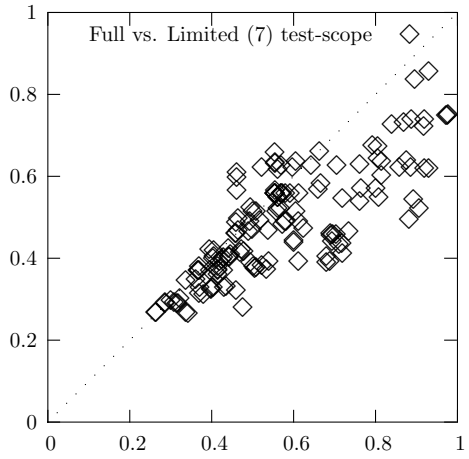

Figure 26: Comparison between Full and Limited (7) test-scope, (purity), (Manual), (Nameconflate-Distinct)

clusters for the dataset. To do this we represent the prediction results in the form of a confusion matrix. For example, the Table 14 shows one such confusion matrix where number of clusters predicted by a measure are given along the rows and the *given* number of clusters are along the column. The cell-values give the count of number of times the measure and the *given* number of clusters agreed. The cell-values in the cells along the slightly shifted diagonal give the count of exact match.

Please note that although the agreement between the prediction by a measure and the *given* value is important, a disagreement does not indicate a complete failure of the cluster stopping measure. This is because the task of grouping entities can be a highly subjective. The basic and the granularity of the distinction can largely differ from person to person or system to system. For example, although we say that there should be only two groups for the *Bill Gates* and *Jason Kidd* dataset, the system might uncover fine-grained features and distinctions corresponding to lets say, *Bill Gates - Microsoft Corporation*, Bill Gates - Gates foundation and *Jason Kidd* and might predict that the appropriate number of clusters is three for this dataset.

We evaluate each cluster stopping measure separately for each genre and sub-genre of datasets.

Table 14: Confusion matrix for the Name-Distinct datasets

	Given														
	PK2					**PK3**					**Gap**				
Predicted	**2**	**3**	**4**	**5**	**6**	**2**	**3**	**4**	**5**	**6**	**2**	**3**	**4**	**5**	**6**
1	5	3	4	2	1	13	4	8	3	4	48	38	19	19	23
2	**13**	1	1	2	5	**40**	30	15	19	23	**17**	12	9	9	8
3	23	**30**	8	10	8	20	**17**	11	7	5	8	**9**	4	2	2
4	23	18	**12**	11	13	4	14	**7**	8	4	1	5	**1**	4	-
5	7	10	4	**8**	5	5	8	-	**5**	5	1	3	1	**1**	3
6	12	7	7	3	**2**	4	3	4	2	**1**	2	2	1	1	**1**
7	4	3	4	1	1	3	5	1	1	1	-	4	1	5	2
8	4	2	2	-	2	1	3	-	1	2	2	-	3	1	2
9	1	2	2	3	2	-	1	1	-	1	-	3	2	-	2
10	-	1	1	-	3	1	-	1	1	1	1	1	-	-	1
11	3	2	-	-	-	-	2	-	1	-	3	-	-	2	-
12	-	3	2	3	-	1	-	-	1	1	-	2	1	1	-

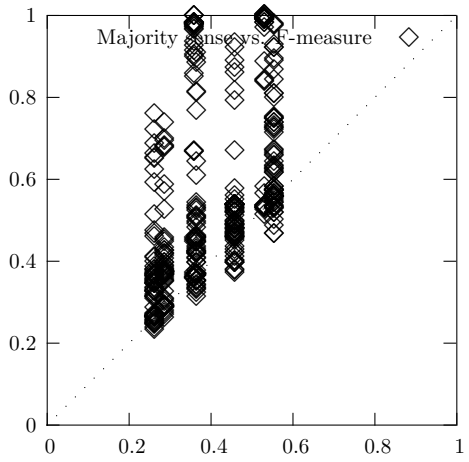

Figure 27: Baseline vs. F-measure, (Manual), (Nameconflate-Distinct)

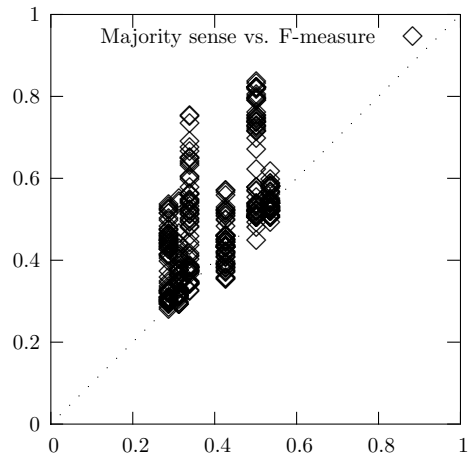

Figure 28: Baseline vs. F-measure, (Manual), (Nameconflate-Subtle)

Figures 31, 32 and 33 compare the three cluster stopping measures for the nameconflate-distinct datasets.

Both PK2 vs. Gap, Figure 31, and PK3 vs. Gap, Figure 32, exhibit more or less similar patterns - many comparable performances, shown across the diagonal but also more points in the bottom-right portion of the plot than in the top-left, indicating that the PK measures are out-performing Gap in many cases.

The plot comparing PK2 and PK3, Figure 33, tells a different story - the performance obtained for the nameconflate-distinct datasets when using PK2 or PK3 is comparable. Although, not all the points are aligned right on the diagonal, the spread of the points around the diagonal is limited and is almost balanced on both sides of the diagonal.

The confusion matrix comparing the prediction accuracy of each measure across the nameconflate-distinct datasets is shown in the Table 14. The reason for the poor performance in terms of F-measure when using Gap's predictions becomes evident in this matrix - in majority of the cases Gap predicts one as the appropriate number of the clusters for the datasets. This indicates that in many cases Gap is not able to see enough pattern to refute the null hypothesis of single cluster.

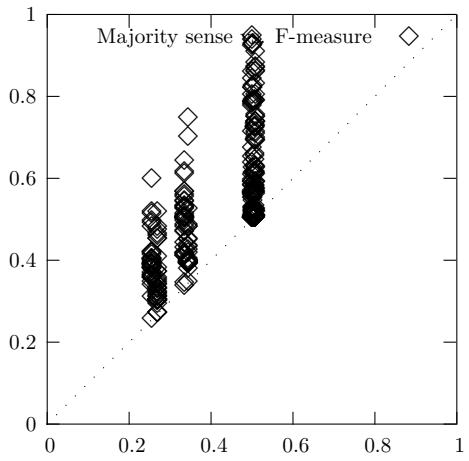

Figure 29: Baseline vs. F-measure, (Manual), (Email-Distinct)

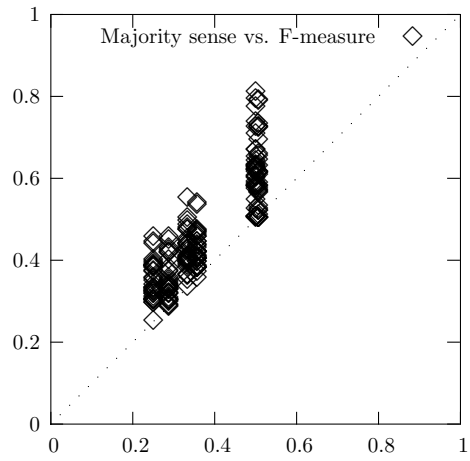

Figure 30: Baseline vs. F-measure, (Manual), (Email-Subtle)

If we sum together the bold-faced numbers along the diagonal for each cluster stopping measure then we see that the PK3 measure agrees with the *given* number of clusters the highest number of times (70) while PK2 follows with 62 agreements and Gap struggles with only 29 agreements.

For the subtle category of the nameconflate datasets, the comparison between PK measures and Gap exhibits similar results as the distinct category (Figures 34 and 35) - PK measures give better performance than Gap. However the results of the comparison between PK2 and PK3 (Figure 36) are different from those of the distinct category - predictions by PK3 measure seem to be giving better performance than that given by the predictions of PK2 measure. The sum of the diagonal values in the Table 15 makes the case clear - PK3 shows agreement in 84 cases while the PK2 exhibits only 48 agreements.

The Figures 37, 38 and 39 for email-distinct datasets, Figures 41, 42 and 40 for the email-subtle category, Figures 43, 44 and 45 for the web dataset and finally the Figures 47, 48 and 46 for the WSD data all more or less support the observations drawn from the name-subtle results:

1. Predictions made by PK measures lead to better overall performance than predictions made by Adapted Gap Statistic.

Table 15: Confusion matrix for the Name-Subtle datasets

	Given								
	PK2			**PK3**			**Gap**		
Predicted	**2**	**3**	**4**	**2**	**3**	**4**	**2**	**3**	**4**
1	7	4	3	8	5	7	52	46	51
2	**19**	6	5	**59**	50	38	**21**	18	15
3	28	**15**	17	22	**18**	24	14	**4**	8
4	14	33	**14**	7	16	**7**	2	5	-
5	11	18	25	6	7	8	2	5	3
6	9	13	9	4	5	5	1	7	1
7	3	6	8	2	1	2	5	1	4
8	3	4	4	2	3	-	2	3	3
9	3	1	1	3	1	2	2	2	1
10	3	-	1	-	1	2	3	1	-
11	3	-	4	-	-	-	-	4	-
12	2	-	1	-	1	1	1	-	3

Table 16: Confusion matrix for the Email-Distinct datasets

	Given								
	PK2			**PK3**			**Gap**		
Predicted	**2**	**3**	**4**	**2**	**3**	**4**	**2**	**3**	**4**
1	2	-	-	2	1	1	47	22	27
2	**8**	6	17	**43**	19	23	**10**	5	9
3	27	**14**	10	22	**11**	10	12	**5**	4
4	11	7	**8**	13	9	**11**	1	3	**3**
5	10	6	1	1	1	2	5	3	-
6	10	5	7	4	2	2	2	2	2
7	5	1	6	1	1	3	1	-	4
8	2	-	3	1	-	3	1	-	2
9	2	2	1	-	-	-	1	-	2
10	-	1	-	1	-	-	-	-	3
11	-	1	1	1	-	1	-	-	-
12	-	-	1	1	1	-	1	-	-

Table 17: Confusion matrix for the Email-Subtle datasets

	Given								
	PK2			**PK3**			**Gap**		
Predicted	**2**	**3**	**4**	**2**	**3**	**4**	**2**	**3**	**4**
1	-	-	-	-	1	-	32	32	24
2	**8**	4	9	**30**	30	26	**5**	5	5
3	14	**21**	11	14	**11**	17	5	**7**	8
4	6	4	**4**	4	5	**2**	1	2	-
5	8	5	3	1	2	3	2	-	-
6	6	7	6	3	4	-	1	2	-
7	2	3	1	3	-	3	1	1	6
8	-	3	5	-	2	2	1	3	1
9	2	4	1	-	1	-	2	-	1
10	2	-	2	2	3	-	-	-	1
11	1	2	1	1	2	-	-	1	2
12	-	1	-	-	-	1	-	1	-

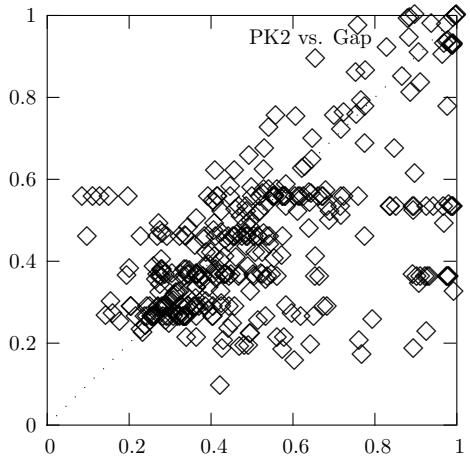

Figure 31: PK2 vs. Gap, (F-measure) (Name-Distinct)

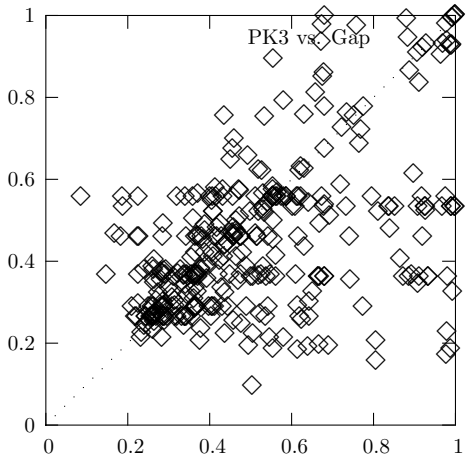

Figure 32: PK3 vs. Gap, (F-measure) (Name-Distinct)

2. Amongst PK measures, performance obtained using PK3 prediction outperforms the performance obtained using PK2 predictions.

Although the second observation conflicts with the observation drawn from the nameconflate-distinct datasets, we think that in general the observations drawn from the subtle category, Web and WSD datasets are more reliable than the distinct category because these genres are more faithful representative of the real data and also are the tougher cases of the problem. At the same time, it is important to notice that if the data is relatively clean and the senses are distinct then the performance given by PK2 prediction should be at par with performance given by PK3 prediction.

5.7 Cluster Labels

Table 20 shows a set of cluster labels generated by our approach for the dataset created by conflating the following names: Serena Williams, Richard Boucher, Michael D. Eisner.

We see that for each of the clusters the set of assigned discriminating labels (described in Section 3.6) completely overlapped with the set of assigned descriptive labels. This is a good sign because it indicates that the features that were most common (descriptive labels) were also the most discrim-

Table 18: Confusion matrix for the Web datasets

	Given								
	PK2			**PK3**			**Gap**		
Predicted	**2**	**3**	**4**	**2**	**3**	**4**	**2**	**3**	**4**
1	3	-	2	5	1	3	71	38	78
2	**41**	6	13	**92**	29	57	**20**	7	15
3	56	**13**	33	26	**12**	24	17	**4**	11
4	12	20	**29**	2	9	**18**	7	3	**3**
5	7	8	20	1	3	9	4	-	2
6	5	4	9	2	4	6	-	1	1
7	2	4	8	2	2	2	1	-	4
8	1	4	5	2	-	2	-	2	-
9	1	-	2	-	-	1	-	-	-
10	-	-	-	-	1	4	2	1	1
11	-	-	-	-	2	1	-	-	-
12	1	1	1	-	-	-	1	1	-

Table 19: Confusion matrix for the WSD datasets

	Given								
	PK2			**PK3**			**Gap**		
Predicted	**3**	**4**	**6**	**3**	**4**	**6**	**3**	**4**	**6**
1	1	2	7	4	2	15	21	21	38
2	18	4	44	26	17	60	11	5	37
3	**15**	19	21	**11**	16	10	**4**	2	7
4	10	**8**	12	4	**5**	10	1	-	4
5	3	5	8	1	2	4	-	2	-
6	1	4	**2**	-	1	**1**	3	3	**4**
7	-	1	3	-	1	2	1	3	3
8	-	1	-	1	1	-	1	1	3
9	-	-	1	1	-	-	-	1	1
10	-	-	1	-	1	-	-	2	-
11	-	-	-	-	-	-	-	2	-

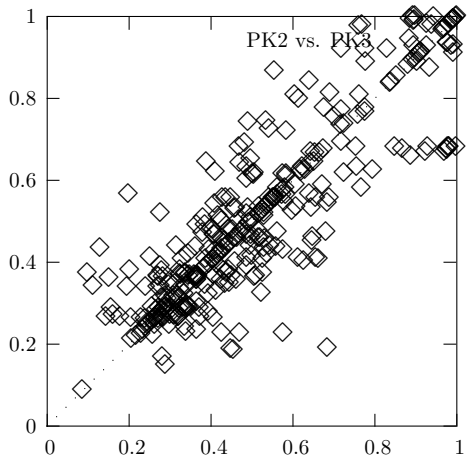

Figure 33: PK2 vs. PK3, (F-measure)
(Name-Distinct)

inating features. In other words, this indicates that the our method was able to separate the contexts into different clusters with high accuracy.

Table 20: Cluster labels for the dataset created by conflating the names: Serena Williams, Richard Boucher, Michael D. Eisner

Cluster	**Assigned Cluster Labels**
Michael D. Eisner	Walt Disney, Walt Co, Los Angeles, Disney Co, chief executive
Richard Boucher	State spokesman, United States, Department spokesman, State Department
Serena Williams	U S, Jennifer Capriati, Henin Hardenne, Grand Slam, New York

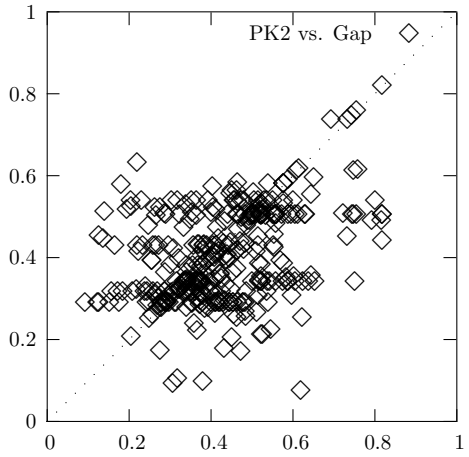

Figure 34: PK2 vs. Gap, (F-measure) (Name-Subtle)

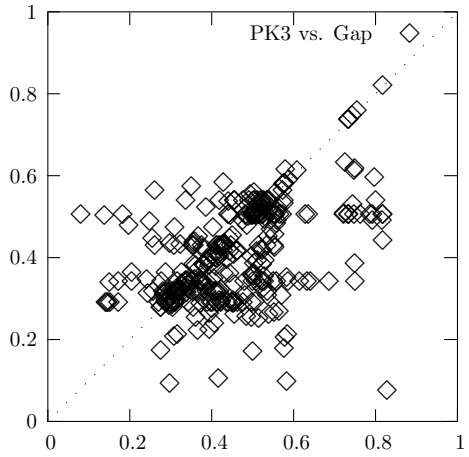

Figure 35: PK3 vs. Gap, (F-measure) (Name-Subtle)

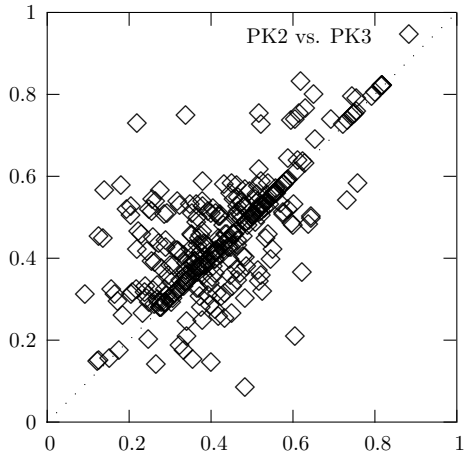

Figure 36: PK2 vs. PK3, (F-measure) (Name-Subtle)

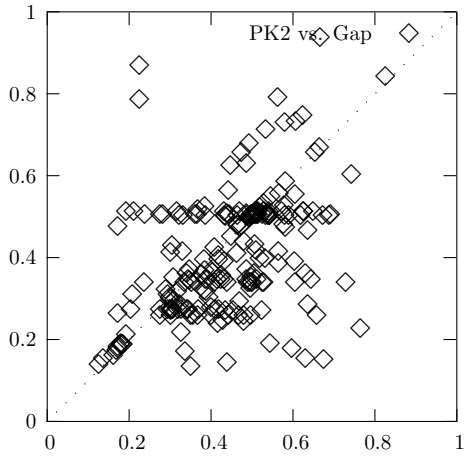

Figure 37: PK2 vs. Gap, (F-measure) (Email-Distinct)

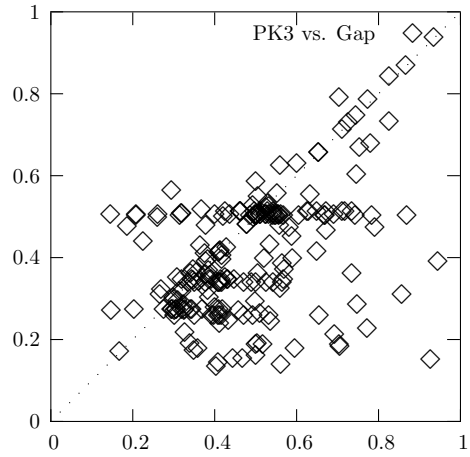

Figure 38: PK3 vs. Gap, (F-measure) (Email-Distinct)

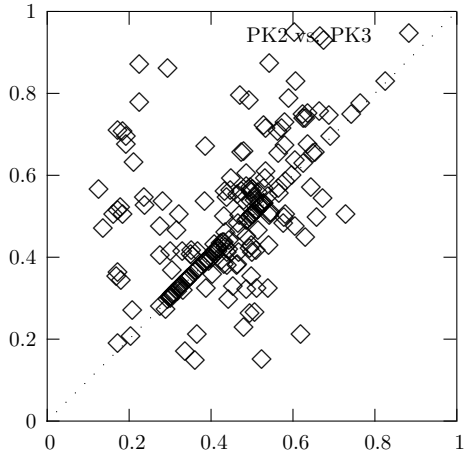

Figure 39: PK2 vs. PK3, (F-measure) (Email-Distinct)

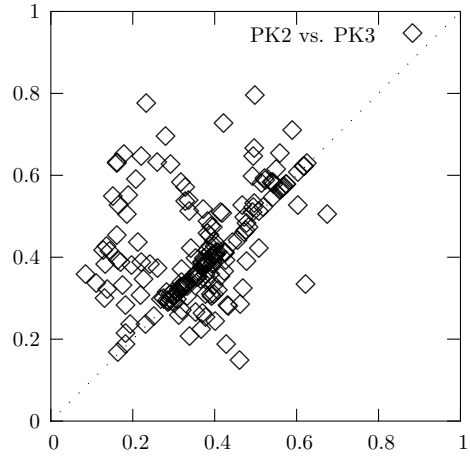

Figure 40: PK2 vs. PK3, (F-measure) (Email-Subtle)

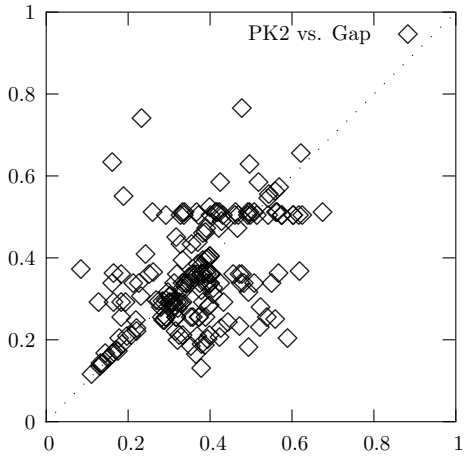

Figure 41: PK2 vs. Gap, (F-measure) (Email-Subtle)

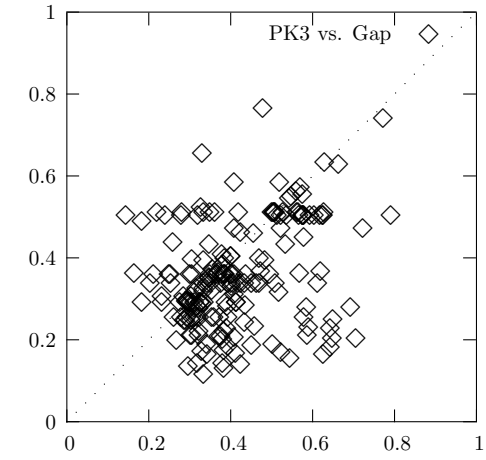

Figure 42: PK3 vs. Gap, (F-measure) (Email-Subtle)

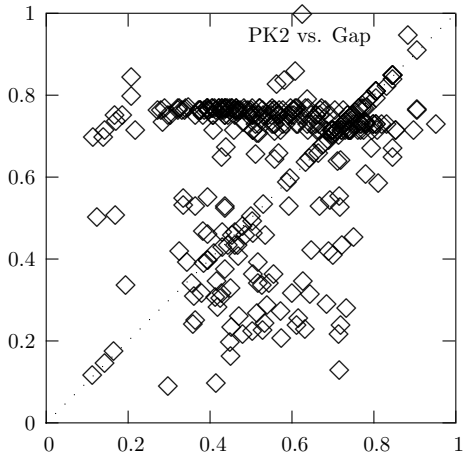

Figure 43: PK2 vs. Gap, (F-measure) (Web)

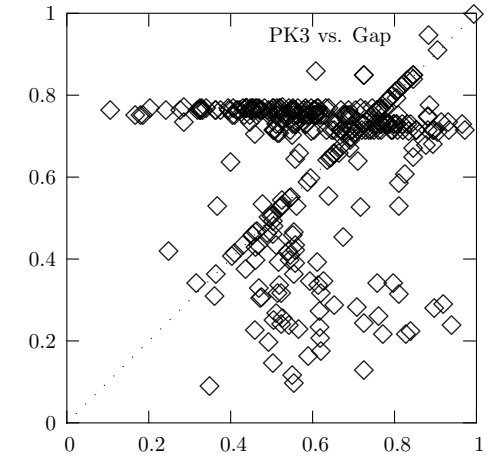

Figure 44: PK3 vs. Gap, (F-measure) (Web)

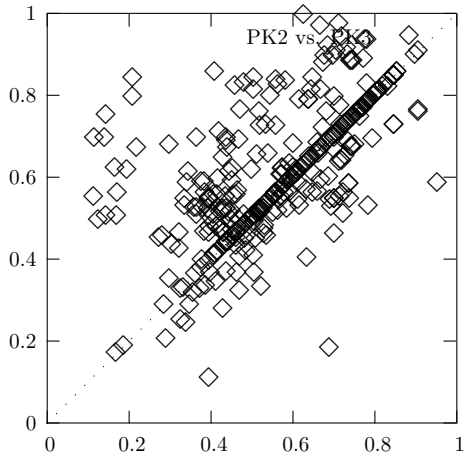

Figure 45: PK2 vs. PK3, (F-measure) (Web)

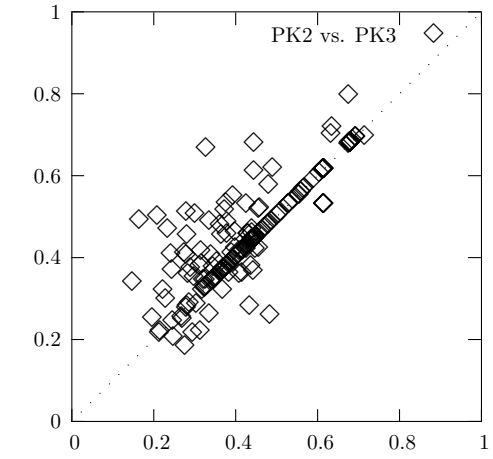

Figure 46: PK2 vs. PK3, (F-measure) (WSD)

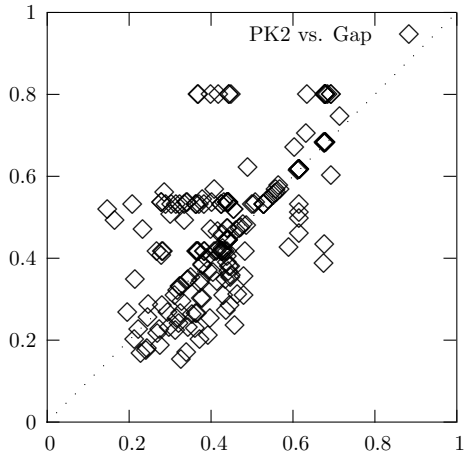

Figure 47: PK2 vs. Gap, (F-measure) (WSD)

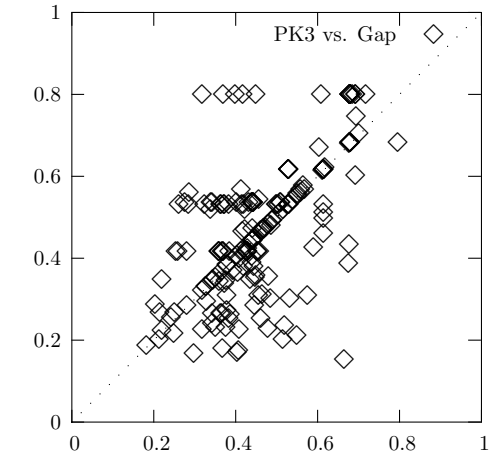

Figure 48: PK3 vs. Gap, (F-measure) (WSD)

6 Related Work

Following is a brief discussion about few different approaches that have been proposed by other people for tasks similar or related to that we investigate in this thesis.

6.1 Name Discrimination

Bagga and Baldwin [1998] have proposed a method using the Vector Space Model to disambiguate references to a person, place or event across documents. The proposed approach uses their previously developed system CAMP (from the University of Pennsylvania) to find *within document* coreference. For example, it might determine that *he* and *the President* refers to *Bill Clinton*. CAMP creates co-reference chains for each entity in a single document, which are then extracted and represented in the vector space model. This model is used to find the similarity among referents, and thereby identify the same referent that occurs in multiple documents.

Gooi and Allan [2004] present a comparison of Bagga and Baldwin's approach to two variations of their own. They used the John Smith Corpus, and created their own corpus which is called the Person-X corpus. Since it is rather difficult to obtain large samples of data where the actual identity of a truly ambiguous name is known, the Person-X corpus consists of pseudo-names that are ambiguous. These are created by disguising known names as Person–X and thereby introduce ambiguities. There are 34,404 mentions of Person–X, which refer to 14,767 distinct underlying entitles. Gooi and Allan re-implement Bagga and Baldwin's context-vector approach, and compare it to another context-vector approach that groups vectors together using agglomerative clustering. They also group instances together based on the Kullback-Liebler Divergence. Their conclusion is that the agglomerative clustering technique works particularly well.

Mann and Yarowsky [2003] take an approach to name discrimination that incorporates information from the World Wide Web. They propose to use various contextual characteristics that are typically found near and within an ambiguous proper-noun for the purpose of disambiguation. They utilize categorical features (e.g., age, date of birth), familial relationships (e.g., wife, son, daughter) and associations that the entity frequently shows (e.g. country, company, organization). Such biographical information about the entities to be disambiguated is mined from the Web using a bootstrapping

method. The Web pages containing the ambiguous name are assigned a vector depending upon the extracted features and then these vectors are grouped using agglomerative clustering.

6.2 Cluster Stopping

The *L method* proposed by Salvador and Chan [2004] estimates the optimal cluster number by looking at an evaluation graph - which is a two dimensional graph where the x-axis is number of clusters and the y-axis represents the clustering error. They note that the leftmost region of this graph (with the sharp slope) and the almost flat region on the right are generally linear. They try to fit lines through these two linear regions and have found that the x-axis value at which the best-fit lines intersect is the optimal number of clusters for that dataset. This approach is however restricted to hierarchical type of clustering algorithms and cannot be used for datasets where the expected number of clusters is one or two.

Hamerly and Elkan [2003] propose an algorithm called *G-means* for finding the appropriate number of clusters that a dataset should be divided into. In this algorithm they start with small number of k-means centers or clusters and iteratively increase the number of clusters as long as each of the current centers/clusters do not appear to belong to a single unimodal distribution such as Gaussian. They use Anderson-Darling statistic to test the clusters for Gaussian fit. However, the largest dimension of the data that they have experimented with is relatively very small (50 dimensions) for our domain.

6.3 Cluster Labeling

Pantel and Ravichandran [2004] have proposed an algorithm for labeling semantic classes, which can be viewed as a form of cluster. For example, a semantic class may be formed by the words: *grapes, mango, pineapple, orange* and *peach*. Ideally one would like this cluster to be labeled as the semantic class of *fruit*. Each word of the semantic class is represented by a feature vector. Each feature consists of syntactic patterns (like verb-object) in which the word occurs. The similarity between a few features from each cluster is found using point-wise mutual information (PMI) and their average is used to group and rank the clusters to form a grammatical template or signature for the class. Then syntactic relationships such as *Noun like Noun* or *Noun such as Noun* are searched

for in the templates to give the cluster an appropriate name label. The output is in the form of a ranked list of concept names for each semantic class.

6.4 Email Clustering

There has been some research on automatically organizing email based on topic or category. However, many of these techniques use supervised learning, which requires an existing pool of labeled examples to serve as training data, and the learned model is limited to assigning incoming email to an existing category.

For example, Bekkerman et al. [2004] propose a supervised approach for categorizing emails into predefined folders. They apply Maximum Entropy, naive Bayes, Support Vector Machine (SVM) and Wide-Margin Winnow classifiers to the Enron and SRI[16] email corpora. They have extended the conventional Winnow classifier for multi-class problems. A Winnow classifier is similar to a simple perceptron which learns weights to be applied to the data instances to determine their output class. The learning involves updating the weights using training data instances. Bekkerman et al. propose using a wider margin or separation between target classes by adjusting the weights not only when a training instance is classified incorrectly but also when it is classified correctly with very small margin over other classes. They use the traditional bag-of-words approach to represent features of the emails. They also introduce an evaluation methodology which they refer to as *step-incremental time-based split* that provides better evaluation of the proposed techniques for the given task of email foldering. There reason behind not using the traditional random test/train split is to account for the time-dependent nature of the email categorization task. Therefore they sort the emails based on their time-stamps and then train on the first N emails and test on the next N emails, subsequently train on the first 2N emails and test on the next N emails and so on. In their results, although the SVMs achieved the best accuracy most of the times, their Wide-Margin Winnow classifier compared fairly well given its simplicity and speed.

Kushmerick and Lau [2005] automate email management based on the structured activities that occur via email, e.g., ordering a book from Amazon.com will lead to a thread of emails regarding - order confirmation, order status, order delivery. The authors use finite-state automata to formalize

[16] http://www.ai.sri.com/project/CALO

this problem where the states of the automata are the status of the process (e.g., order conformation) and the transitions are the email messages. They divide the problem into four tasks of Activity Identification, Transition Identification, Automaton Induction and Message Classification. The first task handles the identification of various emails that are related to an activity. Next task identifies all the emails that cause transition from one state to another. In the third task the process model is generated using the data identified in the previous two tasks. Finally the last task takes care of assigning the appropriate transition to a new message entering the system. Kushmerick and Lau report the results in terms of the accuracy (86% to 97%) with which the methods were able to predict - the next state, the end of activity and the overlap between the predicted and correct transition message.

The 20 Newsgroup email corpora that we experiment with was developed by Ken Lang. This data was then used for the NewsWeeder system (Lang [1995]) which learns user preferences while reading the NetNews. NewsWeeder prompts the user to rate the document that he reads over the range of rating 1 to 5 and uses these ratings to build the user specific learning model. It uses the bag-of-words approach for feature representation and uses tf-idf (term frequency - inverse document frequency) or Minimum Description Length (MDL) to decide the predicted rating for any new document.

7 Conclusions

In this thesis we have extended and generalized the approach taken by Purandare and Pedersen [2004] for unsupervised word sense discrimination to a more widely applicable task of discriminating similar contexts. We have shown the validity of our approach by successfully applying it to various different types of applications and data.

We have shown that along with word sense discrimination our method can be easily used for discriminating contexts containing ambiguous proper names, in short for name discrimination. We demonstrate by performing email clustering that our methods can also be applied to tasks where instead of any specific word or name, the set of entire contexts is to be discriminated.

One of our major contributions in this thesis are the cluster stopping measures. Although clustering techniques and its different variations have been around for a long time now, there have not been many studies about generally applicable cluster stopping techniques. In this thesis we have proposed and implemented three cluster stopping measures (PK2, PK3, Adapted Gap Statistic) which can be used to predict the optimal number of clusters for any given dataset. Although we experiment only with the natural language data in this thesis, the cluster stopping measures can be used with datasets from other domains too. Further more, we think that these measures can be easily abstracted to be used as a framework where the criterion functions can be replaced with any other comparable metric.

We have experimented each of the cluster stopping measures with each of the different genres of data to test the generality of the cluster stopping measures. We find that they can be easily applied to the name discrimination, word sense discrimination and email clustering tasks. We also find that the agreement between the predicted number of clusters by a cluster stopping measure and the given value is maximum for the PK3 measure.

We also introduce a cluster-labeling technique which aims at assisting a user decipher the underlying entity for each generated cluster. Although in its preliminary stages, we show that this technique does a decent job in spite of its simplicity.

We find that if the test data is large enough and if the senses to be discriminated are relatively

distinct then the first order context representation is a better option than the second order context representation. However, for the opposite case of small data and subtle senses the second order gives a better performance. On the whole, the two context representations provide a complementary pair of options.

We conclude from our various experimental results that for our task and domain if one starts with a relatively clean and small data and applies good feature selection criterion like stoplisting, frequency cutoff and test of associations (if using bigrams or co-occurrences) then there is not much to be gained (at least while using the current context-by-feature representation) by applying SVD.

We have also shown that the hybrid clustering algorithm - Repeated Bisections is a good choice, especially for data with subtle senses, but otherwise the agglomerative clustering algorithm gives a comparable performance.

In this thesis we have shown that a broadly defined problem of context discrimination can be successfully approached using unsupervised clustering techniques. We have further made this approach robust and user-independent via our cluster stopping measures and cluster labeling.

8 Future Work

The following are among the most interesting issues that have come to light during our research, and which merit further investigation based on the findings of this thesis.

8.1 Statistical Evaluation

We plan to use Analysis of Variance (ANOVA) to perform statistical evaluation of our methods. The expectation is to use ANOVA to confirm the conclusions we have drawn based on visual inspection of the scatter plots. ANOVA will be helpful in deciding whether the differences in the performances shown by various configurations are statistically significant. If the differences are found to be significant then we plan to compare each pairwise combination of the various configurations using least significant difference (LSD) tests. Such pairwise comparisons try to find which configurations perform better than others. Please refer to Appendix A for an illustration.

8.2 Comparison with Latent Semantic Analysis

We have relied on representations of contexts that are based on finding word and feature co-occurrences that are, in the end, conveyed in a context by feature matrix. Our first order method represents each context with a vector that directly shows the features that occur in that context. Our second order representation also uses a context by feature representation, however, the vector that represents a context is based on the average of the features that co-occur with the features found in the contexts to be clustered.

However, Latent Semantic Analysis presents an alternative representation which is based on a feature by context representation. In other words, features are not represented based on the features that they co-occur with, rather they are represented with respect to the contexts that they co–occur with. We would like to compare the difference between the LSA representation and our representation, since it raises a fundamental question that we believe is very important. Specifically, is the meaning of a word or sentence best judged by determining which words co-occur with those words (which is the basis of our first and second order methods) or is it best represented by the contexts in which

the word or words occur?

In particular, we will compare LSA based (feature by context) context discrimination with first and second order methods (context by feature). We will also compare LSA based word clustering (feature by context) with second order word clustering (context by feature).

8.3 Similarity Values from Ontologies

In some applications we may have semantic information from a manually created ontology available (such as WordNet). We are interested in the effect of building our second order representation not from a word by word co–occurrence matrix, but rather from a concept by concept matrix derived from WordNet similarity scores. In order to represent contexts using "concept vectors", we may need to disambiguate the contexts semantically, but we have methods available (Patwardhan and Pedersen [2005]) that may be appropriate.

8.4 Creating Kernels for SVMs

Joshi [2006] shows that kernels derived from our word by word matrices used for creating second order representations can be incorporated into Support Vector Machines (SVMs) in order to improve the results of supervised Word Sense Disambiguation. How does altering the formulation of these kernels affect the performance of SVM disambiguation, and can we incorporate this into our SenseClusters system relatively seamlessly?

8.5 Improving the Quality of Automatically Generated Cluster Labels

At present our SenseClusters system generates labels for discovered clusters by finding significant bigrams in the contexts that make up a cluster. While this is a reasonable starting point for labels, the results are still somewhat coarse. We would like to improve the readability and quality of automatically generated labels for our clusters, since this will help a user to know the contents of a cluster without having to examine it in detail. Among the ideas we have is to use the context that represents the centroid of each cluster as a label, or to utilize automatic heading generation

techniques to create a summarizing label for a cluster.

8.6 Explore the Effect of Automatically Generated Stoplists

At present our stoplists are manually created. While this is a common technique, it does present an obstacle as we work with new languages and domains, and in general it should be possible to automatically generate stoplists by looking for words that occur in many contexts and are not likely to provide much discriminating power. We would like to explore the impact of using such stoplists, and determine how to best formulate them, since we have observed in our previous work Pedersen et al. [2006b] that stoplists can have a significant impact on overall discrimination accuracy.

8.7 Study Effect of Variations in Feature Selection Data

In recent work Pedersen et al. [2006a] we have observed that improved results can be obtained by using feature selection data that is from a mix of languages, when discriminating among contexts from a language with limited online resources. For example, we improved the results of discriminating on Romanian text by using features identified from large quantities of English corpora. This points out there there may be advantages to using a separate source of training data when dealing with relatively small amounts of data that are to be clustered. Perhaps the most obvious case is that of WWW results, where we may wish to cluster the top 10 or 20 snippets returned by a search engine. Would it be possible to obtain features from other types of corpora that could be used to cluster such results? In addition, we are curious as to the effect of differences in time. Can we cluster contexts created today based on features selected from data that was created last month or last year?

8.8 Develop Ensembles of Cluster Stopping Methods

It is clear that there are a number of cluster stopping methods that perform very effectively, and it is not at all certain that any single method will always perform best. As such we would like to develop ensemble methods that will take the results of multiple cluster stopping methods in order to arrive at a consensus as to the appropriate number of clusters for a given collection of contexts.

8.9 Evaluate Word Clustering Relative to An Application

We have developed methods that cluster words (not contexts) but have not had a clear application in which to use these, nor a clear method of evaluation (partially due to the lack of an application for these word clusters).

We would like to extend the methods of word clustering such that they can be used to identify classes of words (synonyms, for example) and then used to automatically construct ontologies or other knowledge sources that are currently constructed manually.

8.10 Incorporate Syntactic Features

We have preferred to use lexical features only, since we want our methods to be language independent. However, certainly part of speech features, parse structures, etc. could be quite useful for languages where we have tools available to find such features.

8.11 Email Experiments

Our current experiments with Email data have been based on very noisy data. We would like to explore the use of automatically generated stoplists in this domain, and also the effect of eliminating other forms of noise found in email data, such as headers.

A Appendix

A.1 Analysis of Variance

Analysis of Variance (ANOVA) is a statistical technique for comparing the performances given by different settings of an algorithm or by different algorithms all together. For example, given a set of results shown in Table 21, if we are interested in comparing the settings A, B, C, D & E, which are along the columns then we would perform ANOVA on these results. Each of the setting is tested with five different datasets shown along the first five rows. The F-measure score (in percentages) obtained for each of the settings when tested with each of the five datasets is recorded in the respective cells. We will refer to this example as Ex1 henceforth.

Table 21: Example: Ex1

	A	**B**	**C**	**D**	**E**
DS1	94.88	96.22	61.44	76.17	55.45
DS2	60.11	59.16	51.37	51.89	50.00
DS3	68.42	70.26	54.37	57.57	50.00
DS4	53.09	68.95	51.23	63.39	51.41
DS5	89.15	91.03	60.12	54.37	50.45
Total	365.65	385.62	278.53	303.39	257.31
Mean	73.13	77.12	55.71	60.68	51.46
Stdev	18.19	15.77	4.82	9.67	2.30

ANOVA uses hypothesis testing where the null hypothesis (H_0) is that the performance given by all the settings is comparable or in other words, none of the settings are better than others. The alternative hypothesis (H_1) is that the performances exhibited when using different settings are statistically different and thus the settings are not comparable. For example Ex1, the null hypothesis is that, all the five settings (A, B, C, D & E) are comparable and the alternative hypothesis is that, one or more of the settings are relatively better than the others.

Following is a brief description and an illustration based on EX1, about how ANOVA is performed.

To start with, the total, the mean and the standard deviation for each setting is computed. Next step is to compute the sum of squares between groups (SS_B) as follows:

$$SS_B = \frac{(\sum X_A)^2}{n_A} + \frac{(\sum X_B)^2}{n_B} + ... - \frac{(\sum X_T)^2}{n_T} \tag{14}$$

where the X_A refers to each of the five results obtained using the setting A, n_A is the number of results with setting A and thus $n_A = 5$ in this example, X_T refers to all the 25 results and n_T is the total number of results being compared which is 25 in this example.

and sum of squares within groups (SS_W) as:

$$SS_W = \left(\sum X_A^2 - \frac{(\sum X_A)^2}{n_A}\right) + \left(\sum X_B^2 - \frac{(\sum X_B)^2}{n_B}\right) + .. \left(\sum X_E^2 - \frac{(\sum X_E)^2}{n_E}\right) \tag{15}$$

with similar interpretation for symbols as in Formula 14.

Table 22: Analysis of Variance (ANOVA)

	Sum of Squares (SS)	**Degrees of Freedom (df)**	**Mean Square (MS)**	**F value**	**p value**
Between	2459.51	4	614.88	4.38	0.0105
Within	2805.90	20	140.30		
Total	5265.41	24			

The SS_B and SS_W values for the five settings from example Ex1 are shown in the first column of the Table 22.

The degrees of freedom between groups (df_B) is computed as:

$$df_B = \#settings - 1 \tag{16}$$

and the degrees of freedom within groups (df_W) is computed as:

$$df_W = n_T - \#settings \tag{17}$$

The $df_B = 5 - 1 = 4$ and $df_W = 25 - 5 = 20$ values for the example Ex1 are specified in the third column of the Table 22.

The Mean Square between groups is computed as:

$$MS_B = \frac{SS_B}{df_B} \tag{18}$$

and Mean Square within groups is computed as:

$$MS_W = \frac{SS_W}{df_W} \tag{19}$$

The Mean Square values for the example, Ex1, are shown in the fourth column of the Table 22.

$$F\ value = \frac{MS_B}{MS_W} \tag{20}$$

The last step to calculate the F value as shown in Formula 20 and to compare this value with the critical value of F at an appropriate alpha level for the given degrees of freedom. For example, for Ex1 we have looked up the critical value of F in a table of the F distribution at 0.05 alpha value, for degrees of freedom of 4 and 20 respectively and the critical value is 2.8661. As we can see our F value of 4.38 is greater than the critical value of 2.8661 and thus the null hypothesis can be rejected in favor of the alternative hypothesis. In fact, the exact p value (shown in the last column of Table 22) for the obtained F value of 4.38 is as low as 0.0105 which implies that the probability of erroneously rejecting the null hypothesis is as small as 0.0105. In short, we can conclude that for the example Ex1, the five settings being compared show significantly different performances.

A.2 Least Significant Difference (LSD)

Once we have confirmed using ANOVA that the settings being compared are not equal then we proceed to find the settings that are better than others. We use least significant differences (LSD) tests for this task where every pairwise combination of the settings is compared. The LSD test for comparing two settings A & B is formulated as:

$$F_{LSD} = \frac{\left(\overline{X_A} - \overline{X_B}\right)^2}{MS_W * \left(\frac{1}{n_A} + \frac{1}{n_B}\right)} \tag{21}$$

Table 23: Least significant Difference (LSD) Tests

Pair of Settings being compared	F_{LSD}	p value
A & B	0.284259	0.600
A & E	8.366333	0.009
B & E	11.734876	0.003
A & C	5.409950	0.031
B & D	4.819679	0.040

Once the F value (F_{LSD} is computed for a pair of settings, then the next step is to compare the F value to the critical F value at an appropriate alpha level and degrees of freedom. The degrees of freedom for LSD is 1 and $n_T - \#settings$, which is, $25 - 5 = 20$, for Ex1. Few pairwise comparisons for Ex1 along with their F_{LSD} values and the corresponding p values are shown in the Table 23. We can see from the p values that all the pairs except A & B exhibit significantly different performances.

References

A. Bagga and B. Baldwin. Entity–based cross–document co–referencing using the vector space model. In *Proceedings of the 17th international conference on Computational linguistics*, pages 79–85. Association for Computational Linguistics, 1998.

R. Bekkerman, A. McCallum, and G. Huang. Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora. In *Center for Intelligent Information Retrieval, Technical Report IR-418*, pages –, Amherst, MA, 2004.

R. Bruce and J. Wiebe. Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 139–146, 1994.

Ted Dunning. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74, 1994.

C. H. Gooi and J. Allan. Cross-document coreference on a large scale corpus. In S. Dumais, D. Marcu, and S. Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 9–16, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

G. Hamerly and C. Elkan. Learning the k in k-means. In *Advances in Neural Information Processing Systems*, pages 281–288, 2003.

J. Hartigan. *Clustering Algorithms*. Wiley, New York, 1975.

A. Jain, M. Murthy, and P. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3): 264–323, 1999.

M. Joshi. Kernel Methods for Word Sense Disambiguation and Abbreviation Expansion in the Medical Domain. Master's thesis, University of Minnesota, July 2006.

A. Kulkarni. Unsupervised discrimination and labeling of ambiguous names. In *Proceedings of the ACL Student Research Workshop*, pages 145–150, Ann Arbor, Michigan, 2005.

A. Kulkarni and T. Pedersen. Name discrimination and email clustering using unsupervised clustering and labeling of similar contexts. In *Proceedings of the Second Indian International Conference on Artificial Intelligence*, pages 703–722, Pune, India, 2005a.

A. Kulkarni and T. Pedersen. SenseClusters: Unsupervised clustering and labeling of similar contexts. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 105–108, Ann Arbor, Michigan, 2005b.

A. Kulkarni and T. Pedersen. How many different "John Smiths", and who are they? In *Proceedings of the Student Abstract and Poster Session of the Twenty-First National Conference on Artificial Intelligence*, pages –, Boston, Massachusetts, 2006.

N. Kushmerick and T. Lau. Automated Email Activity Management: An Unsupervised Learning Approach. In *International Conference on Intelligent User Interfaces*, pages 67–74, San Diego, CA, 2005.

K. Lang. Newsweeder: Learning to filter netnews. In *The Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, San Francisco, CA, 1995.

C. Leacock, M. Chodorow, and G. Miller. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24(1):147–165, March 1998.

C. Leacock, G. Towell, and E. Voorhees. Corpus-based statistical sense resolution. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 260–265, March 1993.

G. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In W. Daelemans and M. Osborne, editors, *Proceedings of CoNLL-2003*, pages 33–40. Edmonton, Canada, 2003.

G.A. Miller and W.G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991.

R. Mojena. Hierarchical grouping methods and stopping rules: An evaluation. *The Computer Journal*, 20(4):359–363, 1977.

J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5:32–38, 1957.

P. Pantel and D. Ravichandran. Automatically labeling semantic classes. In S. Dumais, D. Marcu, and S. Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 321–328, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

S. Patwardhan, S. Banerjee and T. Pedersen. SenseRelate::TargetWord - A Generalized Framework for Word Sense Disambiguation. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, Ann Arbor, Michigan, 2005.

T. Pedersen and R. Bruce. Distinguishing word senses in untagged text. In *Proceedings of the 2nd Conference on EMNLP*, pages 197–207, Providence, RI, August 1997.

T. Pedersen and A. Kulkarni. Automatic Cluster Stopping with Criterion Functions and the Gap Statistics. In *Proceedings of the Demonstration Session of the Human Language Technology Conference and the Sixth Annual Meeting of the North American Chapter of the Association for Computational Linguistic*, pages –, New York City, NY, 2006a.

T. Pedersen and A. Kulkarni. Selecting the "Right" Number of Senses Based on Clustering Criterion Functions. In *Proceedings of the Posters and Demo Program of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics*, pages –, Trento, Italy, 2006b.

T. Pedersen, A. Kulkarni, R. Angheluta, Z. Kozareva, and T. Solorio. Improving name discrimination : A language salad approach. In *Proceedings of the EACL 2006 Workshop on Cross-Language Knowledge Induction*, pages –, Trento, Italy, April 2006a.

T. Pedersen, A. Kulkarni, R. Angheluta, Z. Kozareva, and T. Solorio. An unsupervised language independent method of name discrimination using second order co-occurrence features. In *Proceedings of the Seventh International Conference on Intelligent Text Processing and Computational Linguistics*, pages 208–222, Mexico City, February 2006b.

T. Pedersen, A. Purandare, and A. Kulkarni. Name discrimination by clustering similar contexts. In *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 220–231, Mexico City, February 2005.

A. Purandare. Unsupervised word sense discrimination by clustering similar contexts. Master's thesis, University of Minnesota, August 2004.

A. Purandare and T. Pedersen. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the CoNLL*, pages 41–48, Boston, MA, 2004.

S. Salvador and P. Chan. Determining the Number of Clusters/Segments in Hierarchical Clustering/Segmentation Algorithms. In *The Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, pages 576–584, Boca Raton, FL, 2004.

H. Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the Gap statistic. *Journal of the Royal Statistics Society (Series B)*, 63:411–423, 2001.

P. Wiemer-Hastings. How latent is latent semantic analysis? In *International Joint Conference on Artificial Intelligence*, pages 932–941, 1999.

Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the 11th Conference of Information and Knowledge Management (CIKM)*, pages 515–524, 2002.