Lexical Semantic Ambiguity Resolution with Bigram-Based Decision Trees

Ted Pedersen

University of Minnesota Duluth, Duluth, MN 55812 USA
tpederse@d.umn.edu,
WWW home page: http://www.d.umn.edu/~tpederse

Abstract. This paper presents a corpus-based approach to word sense disambiguation where a decision tree assigns a sense to an ambiguous word based on the bigrams that occur nearby. This approach is evaluated using the sense-tagged corpora from the 1998 SENSEVAL word sense disambiguation exercise. It is more accurate than the average results reported for 30 of 36 words, and is more accurate than the best results for 19 of 36 words.

1 Introduction

Word sense disambiguation is the process of selecting the most appropriate meaning for a word, based on the context in which it occurs. For our purposes it is assumed that the set of possible meanings, i.e., the sense inventory, has already been determined. For example, suppose *bill* has the following set of possible meanings: a piece of currency, pending legislation, or a bird jaw. When used in the context of *The Senate bill is under consideration*, a human reader immediately understands that *bill* is being used in the legislative sense. However, a computer program attempting to perform the same task faces a difficult problem since it does not have the benefit of innate common–sense or linguistic knowledge.

In the last decade, natural language processing has turned to *corpus–based* methods. These approaches use techniques from statistics and machine learning to induce models of language usage from large samples of text. These models are trained to perform particular tasks, usually via supervised learning. This paper describes an approach where a *decision tree* is learned from some number of sentences where each instance of an ambiguous word has been manually annotated with a sense–tag that denotes the most appropriate sense for that context.

Prior to learning, the sense–tagged corpus must be converted into a more regular form suitable for automatic processing. Each sense–tagged occurrence of an ambiguous word is converted into a feature vector, where each feature represents some property of the surrounding text that is considered to be relevant to the disambiguation process. Given the flexibility and complexity of human language, there is potentially an infinite set of features that could be utilized. However, in corpus–based approaches features usually consist of information that can be

extracted directly from the text, without relying on extensive external knowledge sources. These typically include the part–of–speech of surrounding words, the presence of certain key words within some window of context, and various syntactic properties of the sentence and the ambiguous word. The approach in this paper relies upon a feature set made up of *bigrams*, two word sequences that appear in a text. The context in which an ambiguous word occurs is represented by some number of binary features that indicate whether or not a particular bigram has occurred in the sentence containing the ambiguous word, or in its immediate predecessor.

This paper continues with a discussion of our methods for identifying the bigrams that should be included in the feature set for learning. Then the decision tree learning algorithm is described, as are some benchmark learning algorithms that are included for purposes of comparison. The experimental data is discussed, and then the empirical results are presented. We close with an analysis of our findings and a discussion of related work.

2 Building a Feature Set of Bigrams

We define bigrams simply as two word sequences that occur consecutively in text. Given the sparse and skewed distributions of bigram data, it is important to choose a statistical test or measure appropriate for this kind of data. We explore two alternatives, the power divergence family of goodness of fit statistics and the Dice Coefficient, an information theoretic measure related to Mutual Information.

Figure 1 shows an example of a 2×2 contingency table used for storing bigram counts. We use this representation and notation in the following discussion. The value of n_{11} shows how many times the bigram *big cat* occurs in the corpus. The value of n_{12} shows how often bigrams occur where *big* is the first word and *cat* is not the second. The counts in n_{+1} and n_{1+} indicate how often words *big* and *cat* occur as the first and second words of any bigram in the corpus. The total number of bigrams in the corpus is represented by n_{++}.

	cat	¬cat	totals
big	n_{11}= 10	n_{12}= 20	n_{1+}= 30
¬big	n_{21}= 40	n_{22}= 930	n_{2+}= 970
totals	n_{+1}=50	n_{+2}=950	n_{++}=1000

Fig. 1. Representation of Bigram Counts

2.1 The Power Divergence Family

[3] introduce the power divergence family of goodness of fit statistics. A number of well known statistics belong to this family, including the likelihood ratio statistic G^2 and Pearson's X^2 statistic.

These measure the divergence of the observed (n_{ij}) and expected (m_{ij}) sample counts, where m_{ij} is calculated assuming that the the words in the bigram have no relationship or association to one another.

$$m_{ij} = \frac{n_{i+}\ n_{+j}}{n_{++}}$$

Given this value, G^2 and X^2 are calculated as:

$$G^2 = 2 \sum_{i,j} n_{ij} \log \frac{n_{ij}}{m_{ij}}$$

$$X^2 = \sum_{i,j} \frac{(n_{ij} - m_{ij})^2}{m_{ij}}$$

[5] argues in favor of G^2 over X^2, especially when dealing with very sparse and skewed data distributions. However, [3] show that there are cases where Pearson's statistic is more reliable than the likelihood ratio and that there is no reason to always prefer one over the other. In light of this, [10] presented Fisher's exact test as an alternative.

We have developed an approach to deciding which of these tests to use based on the observation that they should all produce the same result when the sample counts stored in the contingency table are not violating any of the distributional assumptions that underly the goodness of fit statistics. We compute values for X^2, G^2, and Fisher's exact test for each bigram. If they differ, then this is a case where the distribution of the bigram counts is causing at least one of the tests to become unreliable. When this occurs we rely upon the value from Fisher's exact test since it does not depend upon assumptions about the underlying distribution of data. Since Fisher's exact test can be computationally complex, a practical shortcut is to run both X^2 and G^2 and see if they differ. If they produce comparable results then they are likely reliable and Fisher's test can be omitted.

For the experiments in this paper, we identified the top 100 ranked bigrams that occur more than 5 times in the training corpus associated with a word. Given that low frequency bigrams are excluded, there are no cases where G^2, X^2, and Fisher's exact test disagreed. All of these statistics produced the same rankings, so hereafter we make no distinction among them and simply refer to them generically as the power divergence statistic.

2.2 Dice Coefficient

The Dice Coefficient is a descriptive statistic that provides a measure of association among two words in a corpus. It is similar to pointwise Mutual Information,

a widely used measure that was first introduced for identifying lexical relationships in [2]. Since Mutual Information is so well–known, we describe it first so as to make the relationship between it and the Dice Coefficient clear. Pointwise Mutual Information can be defined as follows:

$$MI(w_1, w_2) = log_2 \frac{n_{11} * n_{++}}{n_{+1} * n_{1+}}$$

where w_1 and w_2 represent the two words that make up the bigram, n_{11} represents the number of times the two words occur together as a bigram, n_{+1} and n_{1+} are the number of times the words occur as the first and second words of a bigram, and n_{++} represents the total number of bigrams in the corpus.

Mutual Information quantifies how often a word occurs in a bigram (the numerator) relative to how often it occurs overall in the corpus both in and out of the bigram (the denominator). However, there is a curious limitation to pointwise Mutual Information. A bigram $w_1 w_2$ that occurs n_{11} times in the corpus, and whose component words w_1 and w_2 also occur n_{11} times (i.e., the only time the component words occur is together in the bigram), will result in increasingly strong measures of association as the value of n_{11} decreases. Thus, the greatest possible pointwise Mutual Information value is attained when the frequencies of the bigram and its component words are all 1. This causes rankings to be dominated by very low frequency bigrams that may not be especially useful for the disambiguation process.

The Dice Coefficient overcomes this limitation, and can be defined as follows:

$$Dice(w_1, w_2) = \frac{2 * n_{11}}{n_{+1} + n_{1+}}$$

When $n_{11} = n_{1+} = n_{+1}$ the value $DC(w_1, w_2)$ will be 1 for any value of n_{11}. When the values of n_{11} is less than either of the marginal totals (the more typical case) the rankings produced by the Dice Coefficient are similar to those of Mutual Information. The relationship between Mutual Information and the Dice Coefficient is also discussed in [12].

3 Learning Decision Trees

Decision trees are among the most widely used machine learning algorithms. They perform a general to specific search of a feature space, adding the most informative features to a tree structure as the search proceeds. The objective is to select a minimal set of features that efficiently partitions the feature space into classes of observations and assemble them into a tree. In our case, the observations are manually sense–tagged examples of an ambiguous word in context and the partitions correspond to the different possible senses.

Each feature selected during the search process is represented by a node in the learned decision tree. Each node represents a choice point between a number of different possible values for a feature. Learning continues until all the training

examples are accounted for by the decision tree. In general, such a tree will be overly specific to the training data and not generalize well to new examples. Therefore learning is followed by a pruning step where some nodes are eliminated or reorganized to produce a tree that can generalize to new circumstances.

Test instances are disambiguated by finding a path through the learned decision tree from the root to a leaf node that corresponds with the observed features. In effect an instance of an ambiguous word is disambiguated by passing it through a series of tests, where each test asks if a particular bigram occurs nearby.

We also include three benchmarks in this study: the majority classifier, decision stumps, and the Naive Bayesian classifier.

The *majority classifier* assigns the most common sense in the training data to every instance in the test data. A *decision stump* is a one node decision tree[6] that is created by stopping the decision tree learner after the single most informative feature is added to the tree.

The *Naive Bayesian classifier* [4] is based on certain blanket assumptions about the interactions among features in a corpus. There is no search of the feature space performed to build a representative model as is the case with decision trees. Instead, all features are assumed to be relevant to the task at hand and are assigned weights based on their frequency of occurrence in the training data. It is most often used with a *bag of words* feature set, where every word in the training examples is represented with a binary feature that indicates whether or not it occurs in some proximity to the ambiguous word.

We have developed Perl software to identify bigrams and convert the text into feature vectors for input to a learning algorithm, and have made this freely available at our WWW site. We use the Weka [14] implementations of the C4.5 decision tree learner (known as J48), the decision stump, and the Naive Bayesian classifier.

4 Experimental Data

Our empirical study utilizes the training and test data from the 1998 SENSEVAL evaluation of word sense disambiguation systems. Ten teams participated in the supervised learning portion of this event. Additional details about the exercise, including the data and results referred to in this paper, can be found at the SENSEVAL web site and in [7].

We included all 36 tasks from SENSEVAL for which training and test data were provided. Each task requires that the occurrences of a particular word in the test data be disambiguated based on a model learned from the sense–tagged instances in the training data. Some words were used in multiple tasks as different parts of speech. For example, there were two tasks associated with *bet*, one for its use as a noun and the other as a verb. Thus, there are 36 tasks involving the disambiguation of 29 different words.

The words and part of speech associated with each task are shown in Table 1 in columns 1 and 2. Note that the parts of speech are encoded as *n* for noun,

a for adjective, v for verb, and p for words where the part of speech was not provided. The number of test and training instances for each task are shown in columns 3 and 5. Each instance consists of the sentence in which the ambiguous word occurs as well as either the preceding or succeeding sentence. In general the total context available for each ambiguous word is less than 100 surrounding words. The number of senses that exist in the test data for each task is shown in column 4.

5 Experimental Method

The following process is repeated for each task. Capitalization and punctuation are removed from the training and test data. Two feature sets are selected from the training data based on the top 100 ranked bigrams according to the power divergence statistic and the Dice Coefficient. The bigram must have occurred 5 or more times to be included as a feature. The training and test data are converted to feature vectors where each feature represents the occurrence of one of the bigrams that appears in the feature set. This representation of the training data is the actual input to the learning algorithms. There are two different decision tree and decision stump learning processes, one based on the feature set determined by the power divergence statistic and another from the feature set identified by the Dice Coefficient. The majority classifier does not use a feature set, but rather simply determines the most frequent sense in the training data and assigns that to all instances in the test data. The Naive Bayesian classifier is based on a feature set where every word that occurs 5 or more times in the training data is included as a feature.

The learned decision tree is then used to disambiguate the test data. The test data has been kept completely out of the process until now. It is not used to select bigram features nor is it used in any phase of decision tree learning. We employ a fine grained scoring method, where a word is counted as correctly disambiguated only when the assigned sense tag exactly matches the true sense tag. No partial credit is assigned for near misses.

6 Experimental Results

The accuracy attained by each of the learning algorithms is shown in Table 1. Column 6 reports the accuracy of the majority classifier, columns 7 and 8 show the best and average accuracy reported by the 10 participating SENSEVAL teams. The evaluation at SENSEVAL was based on precision and recall, so we converted those scores to accuracy by taking their product. However, the best precision and recall may have come from different teams, so the best accuracy shown in column 7 may actually be higher than that of any single participating SENSEVAL system. The average accuracy in column 8 is the product of the average precision and recall reported for the participating SENSEVAL teams. Column 9 shows the accuracy of the decision tree using the J48 learning algorithm and the features identified by power divergence statistic. Column 11

shows the accuracy of the decision tree when the Dice Coefficient selects the features. Columns 10 and 12 show the accuracy of the Decision Stump based on the power divergence statistic and the Dice Coefficient respectively. Finally, Column 14 shows the accuracy of the Naive Bayesian classifier based on a bag of words feature set.

The most accurate method is the decision tree based on a feature set determined by the power divergence statistic. The last line of Table 1 shows the win-tie-loss score for the decision tree/power divergence method. A win means it was more accurate than the method in the column, a loss means it was less accurate, and a tie means it was equally accurate. This approach was more accurate than the best reported SENSEVAL results for 19 of the 36 tasks, and more accurate for 30 of the 36 tasks when compared to the average reported accuracy. The decision stumps also fared well, proving to be more accurate than the best SENSEVAL results for 14 of the 36 tasks.

There are 6 tasks where the decision tree / power divergence approach is less accurate than the SENSEVAL average; promise-n, scrap-n, shirt-n, amaze-v, bitter-p, and sanction-p. The most dramatic difference occurred with amaze-v, where the SENSEVAL average was 92.4% and the decision tree accuracy was 58.6%. However, this was an unusual task where every instance in the test data belonged to a single sense that was not even the majority sense in the training data.

7 Analysis of Experimental Results

The characteristics of the decision trees and decision stumps learned for each word are shown in Table 2. Columns 1 and 2 show the word and part of speech. Columns 3, 4, and 5 are based on the feature set selected by the power divergence statistic while columns 5, 6, and 7 are based on the Dice Coefficient. Columns 3 and 6 show the node selected to serve as the decision stump. Columns 4 and 7 show the number of leaf nodes in the learned decision tree relative to the number of total nodes. Columns 5 and 8 show the number of bigram features selected to represent the training data.

This table shows that there is little difference in the decision stump nodes selected from feature sets determined by the power divergence statistics versus the Dice Coefficient. However this is to be expected since the top ranked bigrams for each measure are consistent, and the decision stump node is generally chosen from among those.

However, the power divergence statistic and Dice Coefficient do result in different sets of features overall, and this is reflected in the different sized trees that are learned from these feature sets. The number of leaf nodes and the total number of nodes for each learned tree is shown in columns 4 and 7. The number of leaf nodes shows how many unique paths from the root of the tree to a sense distinction/leaf node exist. The number of total nodes is the sum of the leaf nodes and the internal nodes. Since a bigram feature can only appear once in the decision tree, the number of internal nodes represents the number

Table 1. Experimental Results

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)
								j48	stump	j48	stump	naive
word	pos	test	senses	train	maj	best	avg	pow	pow	dice	dice	bayes
accident	n	267	8	227	75.3	87.1	79.6	85.0	77.2	83.9	77.2	83.1
behaviour	n	279	3	994	94.3	92.9	90.2	95.7	95.7	95.7	95.7	93.2
bet	n	274	15	106	18.2	50.7	39.6	41.8	34.5	41.8	34.5	39.3
excess	n	186	8	251	1.1	75.9	63.7	65.1	38.7	60.8	38.7	64.5
float	n	75	12	61	45.3	66.1	45.0	52.0	50.7	52.0	50.7	56.0
giant	n	118	7	355	49.2	67.6	56.6	68.6	59.3	66.1	59.3	70.3
knee	n	251	22	435	48.2	67.4	56.0	71.3	60.2	70.5	60.2	64.1
onion	n	214	4	26	82.7	84.8	75.7	82.7	82.7	82.7	82.7	82.2
promise	n	113	8	845	62.8	75.2	56.9	48.7	63.7	55.8	62.8	78.0
sack	n	82	7	97	50.0	77.1	59.3	80.5	58.5	80.5	58.5	74.4
scrap	n	156	14	27	41.7	51.6	35.1	26.3	16.7	26.3	16.7	26.7
shirt	n	184	8	533	43.5	77.4	59.8	46.7	43.5	51.1	43.5	60.9
amaze	v	70	1	316	0.0	100.0	92.4	58.6	12.9	60.0	12.9	71.4
bet	v	117	9	60	43.2	60.5	44.0	50.8	58.5	52.5	50.8	58.5
bother	v	209	8	294	75.0	59.2	50.7	69.9	55.0	64.6	55.0	62.2
bury	v	201	14	272	38.3	32.7	22.9	48.8	38.3	44.8	38.3	42.3
calculate	v	218	5	249	83.9	85.0	75.5	90.8	88.5	89.9	88.5	80.7
consume	v	186	6	67	39.8	25.2	20.2	36.0	34.9	39.8	34.9	31.7
derive	v	217	6	259	47.9	44.1	36.0	82.5	52.1	82.5	52.1	72.4
float	v	229	16	183	33.2	30.8	22.5	30.1	22.7	30.1	22.7	56.3
invade	v	207	6	64	40.1	30.9	25.5	28.0	40.1	28.0	40.1	31.0
promise	v	224	6	1160	85.7	82.1	74.6	85.7	84.4	81.7	81.3	85.3
sack	v	178	3	185	97.8	95.6	95.6	97.8	97.8	97.8	97.8	97.2
scrap	v	186	3	30	85.5	80.6	68.6	85.5	85.5	85.5	85.5	82.3
seize	v	259	11	291	21.2	51.0	42.1	52.9	25.1	49.4	25.1	51.7
brilliant	a	229	10	442	45.9	31.7	26.5	55.9	45.9	51.1	45.9	58.1
floating	a	47	5	41	57.4	49.3	27.4	57.4	57.4	57.4	57.4	55.3
generous	a	227	6	307	28.2	37.5	30.9	44.9	32.6	46.3	32.6	48.9
giant	a	97	5	302	94.8	98.0	93.5	95.9	95.9	94.8	94.8	94.8
modest	a	270	9	374	61.5	49.6	44.9	72.2	64.4	73.0	64.4	68.1
slight	a	218	6	385	91.3	92.7	81.4	91.3	91.3	91.3	91.3	91.3
wooden	a	196	4	362	93.9	81.7	71.3	96.9	96.9	96.9	96.9	93.9
band	p	302	29	1326	77.2	81.7	75.9	86.1	84.4	79.8	77.2	83.1
bitter	p	373	14	144	27.0	44.6	39.8	36.4	31.3	36.4	31.3	32.6
sanction	p	431	7	96	57.5	74.8	62.4	57.5	57.5	57.1	57.5	56.8
shake	p	356	36	963	23.6	56.7	47.1	52.2	23.6	50.0	23.6	46.6
win-tie-loss					23-7-6	19-0-17	30-0-6		28-9-3	14-15-7	28-9-3	24-1-11

of bigram features selected by the decision tree learner. This acts as a second level of feature selection, further reducing the original feature set selected by the power divergence statistic or the Dice Coefficient. In general the number of features included in the decision tree is quite a bit less than the original number of features. Note that the smallest decision trees are functionally equivalent to other classifiers. A decision tree with 1 leaf node and no internal nodes (1/1) acts as a majority classifier. A decision tree with 2 leaf nodes and 1 internal node (2/3) has the structure of a decision stump.

For most words the top ranked 100 bigrams constitute the feature set that is used to represent the training data. If there were ties in the top 100 then there may be more than 100 features, and if the there were fewer than 100 bigrams that occurred more than 5 times then feature selection reduces to choosing all of those bigrams.

8 Discussion

One of our long-term objectives is to identify a simple set of features that will be useful for disambiguating a wide class of words using both supervised and unsupervised methodologies. The results of this paper suggest that bigrams may be an appropriate starting point. Our interest in extending these methods to unsupervised approaches motivated the decision to include bigrams that did not include the ambiguous word as one of the components. Thus, many of the bigrams that were included in the decision trees are bigrams that can be selected without the aid of sense–tagged text.

We hypothesize that accurate decision trees of bigrams will generally include a relatively small number of bigram features. The decision stump results tend to support this view, showing that high accuracy is attainable with just a single bigram feature. Thus, we set the initial criteria for identifying bigram features to include at most the top 100 ranked bigrams and implement an aggressive pruning strategy during the decision tree learning stage. As a result, there were no decision trees that used all of the bigram features, and most of them discarded a considerable number of features. The number of features included in each decision tree can can be seen in Table 2 by taking the difference between the node and leaf counts in columns 4 or 7 and comparing that to the number of features shown in columns 5 or 8.

Decision trees have the considerable advantage that intuitive and understandable rules for disambiguation can be easily extracted from the tree structure. Each path from the root to a leaf node represents a series of binary choices based on whether or not a particular bigram occurs in the text being disambiguated. These rules can be used to discover more general principles of disambiguation and potentially identify features that are useful for a broad class of words.

We found that the feature sets selected by the power divergence statistic tended to result in more accurate decision trees than those selected by the Dice Coefficient. We hypothesize that this is due to the fact that the *gain ratio* used by the decision tree learner J48 to select nodes is based on Mutual Information

Table 2. Decision Tree and Stump Characteristics

		power divergence			dice coefficient		
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
word	pos	stump	node	leaf/node features	stump	node	leaf/node features
accident	n	by accident	8/15	101	by accident	12/23	112
behaviour	n	best behaviour	2/3	100	best behaviour	2/3	104
bet	n	betting shop	20/39	50	betting shop	20/39	50
excess	n	in excess	13/25	104	in excess	11/21	102
float	n	the float	7/13	13	the float	7/13	13
giant	n	the giants	16/31	103	the giants	14/27	78
knee	n	knee injury	23/45	102	knee injury	20/39	104
onion	n	in the	1/1	7	in the	1/1	7
promise	n	promise of	95/189	100	a promising	49/97	107
sack	n	the sack	5/9	31	the sack	5/9	31
scrap	n	scrap of	7/13	8	scrap of	7/13	8
shirt	n	shirt and	38/75	101	shirt and	55/109	101
amaze	v	amazed at	11/21	102	amazed at	11/21	102
bet	v	i bet	4/7	10	i bet	4/7	10
bother	v	be bothered	19/37	101	be bothered	20/39	106
bury	v	buried in	28/55	103	buried in	32/63	103
calculate	v	calculated to	5/9	103	calculated to	5/9	103
consume	v	on the	4/7	20	on the	4/7	20
derive	v	derived from	10/19	104	derived from	10/19	104
float	v	floated on	24/47	80	floated on	24/47	80
invade	v	to invade	55/109	107	to invade	66/127	108
promise	v	promise to	3/5	100	promise you	5/9	106
sack	v	return to	1/1	91	return to	1/1	91
scrap	v	of the	1/1	7	of the	1/1	7
seize	v	to seize	26/51	104	to seize	57/113	104
brilliant	a	a brilliant	26/51	101	a brilliant	42/83	103
floating	a	in the	7/13	10	in the	7/13	10
generous	a	a generous	57/113	103	a generous	56/111	102
giant	a	the giant	2/3	102	a giant	1/1	101
modest	a	a modest	14/27	101	a modest	10/19	105
slight	a	the slightest	2/3	105	the slightest	2/3	105
wooden	a	wooden spoon	2/3	104	wooden spoon	2/3	101
band	p	band of	14/27	100	the band	21/41	117
bitter	p	a bitter	22/43	54	a bitter	22/43	54
sanction	p	south africa	12/23	52	south africa	12/23	52
shake	p	his head	90/179	100	his head	81/161	105

and as such is closely related to the Dice Coefficient. We believe that this overly biases the feature selection processes towards Mutual Information and results in a feature set that is skewed towards that measure and not optimal for classification.

9 Related Work

Bigrams have been used as features for word sense disambiguation, particularly in the form of collocations where the ambiguous word is one component of the bigram (e.g., [1], [9], [16]). While some of the bigrams we identify are collocations that include the word being disambiguated, there is no requirement that this be the case. This makes our approach less dependent on sense–tagged text and suggests that it may extend more easily to environments where the amount of sense–tagged text is smaller or does not exist.

Decision trees have been used in supervised learning approaches to word sense disambiguation, and have fared well in a number of comparative studies (e.g., [8], [11]). In the former they were used with the bag of word feature sets and in the latter they were used with a mixed feature set that included part-of-speech, morphological, and collocation features. The approach in this paper is the first time that decision trees based strictly on bigram features have been employed.

The decision list is a closely related approach that has also been applied to word sense disambiguation (e.g., [15], [13], [17]). Rather than building and traversing a tree to perform disambiguation, a list is employed. In the general case a decision list may suffer from less fragmentation during learning than decision trees. However, we believe that fragmentation also reflects upon the feature set used to represent the training data. Our feature set is based on 100 binary features. This is a relatively small feature space and not as likely to suffer from fragmentation as a larger space.

10 Conclusions

This paper shows that bigrams are powerful features for performing word sense disambiguation. Our findings show that a simple decision tree where each node tests whether or not a particular bigram occurs near the ambiguous word results in accuracy comparable with state–of–the–art methods. This is demonstrated via an empirical comparison using data from the 1998 SENSEVAL word sense disambiguation exercise that shows the decision tree approach is more accurate than the best SENSEVAL results for 19 of 36 words.

11 Acknowledgments

This work has been supported by a Grant–in–Aid of Research, Artistry and Scholarship from the Office of the Vice President for Research and the Dean of the Graduate School of the University of Minnesota. I would like to thank the SENSEVAL organizers and participants for putting the data and results of the 1998 event in the public domain.

References

1. 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.
2. K. Church and P. Hanks. Word association norms, mutual information and lexicography. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 76–83, 1990.
3. N. Cressie and T. Read. Multinomial goodness of fit tests. *Journal of the Royal Statistics Society Series B*, 46:440–464, 1984.
4. R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, NY, 1973.
5. T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.
6. R. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11:63–91, 1993.
7. A. Kilgarriff and M. Palmer. Special issue on SENSEVAL: Evaluating word sense disambiguation programs. *Computers and the Humanities*, 34(1–2), 2000.
8. R. Mooney. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 82–91, May 1996.
9. H.T. Ng and H.B. Lee. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *Proceedings of the 34th Annual Meeting of the Society for Computational Linguistics*, pages 40–47, 1996.
10. T. Pedersen. Fishing for exactness. In *Proceedings of the South Central SAS User's Group (SCSUG-96) Conference*, pages 188–200, Austin, TX, October 1996.
11. T. Pedersen and R. Bruce. A new supervised learning algorithm for word sense disambiguation. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 604–609, Providence, RI, July 1997.
12. F. Smadja, K. McKeown, and V. Hatzivassiloglou. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1–38, 1996.
13. Y. Wilks and M. Stevenson. Word sense disambiguation using optimised combinations of knowledge sources. In *Proceedings of COLING/ACL-98*, 1998.
14. I. Witten and E. Frank. *Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan–Kaufmann, San Francisco, CA, 2000.
15. D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, 1994.
16. D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, MA, 1995.
17. D. Yarowsky. Hierarchical decision lists for word sense disambiguation. *Computers and the Humanities*, 34(1–2), 2000.