# Approaches for Automatically Tagging Affect

**Nathanael Chambers**
Institute for Human and Machine Cognition
40 S. Alcaniz Street
Pensacola, FL 32501
nchambers@ihmc.us

**Joel Tetreault** and **James Allen**
University of Rochester
Department of Computer Science
Rochester, NY, 14627
{tetreaul,allen}@cs.rochester.edu

**Abstract**

The tagging of discourse is important not only for natural language processing research, but for many applications in the social sciences as well. This paper describes an evaluation of a range of different tagging techniques to automatically determine the attitude of speakers in transcribed psychiatric dialogues. It presents results in a marriage counseling domain that classifies the attitude and emotional commitment of the participants to a particular topic of discussion. It also gives results from the Switchboard Corpus to facilitate comparison for future work. Finally, it describes a new Java tool that learns attitude classifications using our techniques and provides a flexible, easy to use platform for tagging of texts.

## Introduction

There are many applications that require an analysis of the actions performed in discourse. Most obvious would be work on dialogue systems, where identifying the correct act is crucial to producing an appropriate response. However, knowing the correct act can also aid in other parts of the understanding task. For example, behavioral scientists often need to analyze conversations in order to draw conclusions about the participants' state of mind. A psychiatrist may study the physician/patient relationship by coding transcripts with different styles of interaction and the attitudes each participant has toward each other. This tagging operation has been proven to be very useful in analyzing patients' interactions and providing assistance to them (Hodson, Shields, and Rousseau 2003). Tagging by hand can be extremely time consuming and tends to be unreliable, so automated tools that can quickly learn tagging schemes from small amounts of data would be very useful.

In this paper, we evaluated several different tagging approaches with respect to their tagging accuracy, ranging from simple n-gram approaches with statistical models that are easy to learn and apply, to more complex information retrieval techniques that require significant computation to produce the models. The goal was to identify the most promising techniques to use in an automatic tagging tool. This tool will then be used to develop tagging models for new domains and to use these models to automatically tag new data in the domains.



We evaluated our approach on two corpora. We used the Rochester Marriage Counseling Corpus (Shields 1997) because it provides a rich dialogue between married couples, providing informal utterances that are tagged for the attitude each spouse has toward a particular topic. The conversations are frank, sometimes intense dialogues between people who have known each other for years. It is a unique domain that provides excellent data for analyzing attitude and affect in dialogue. We also used the Switchboard Corpus (Godfrey, Holliman, & McDaniel 1992) for the two-fold purpose of comparing our results to previous work and future approaches in determining dialogue attitude. The two corpora are ideal because they are meticulously hand-tagged and are relatively informal dialogues in which the attitude and affect of the participants is not repressed by a simulated experiment.

## Background

Work in textual Affective Computing has spawned a myriad of approaches to identify the emotions expressed by a sentence or discourse. The earliest approaches involve searching a text for predetermined keywords and other cue phrases that convey strong emotion, and tagging the sentence depending on which keywords (if any) it contains. While this method has the advantage of speed, it fails in sentences with negation and is limited by the list of keywords.

A tagged dictionary can identify (Boucouvalas and Ze 2002) the basis of emotion in phrases and then use grammatical features to determine which of the words in the sentence carry some emotional weight. Its main advantage is that it has a scaled model of emotion, but has the drawback of having to create a tagged dictionary that encompasses the target domain.

Statistical methods have the advantage of being free from the pretagged list constraints but are dependent on having a large enough tagged corpus for training and, in some instances, do not fare well with tagging at the sentence level. Goertzel's Webmind project (Goertzel, Silverman, and Hartley 2000) uses latent semantic analysis for affect classification in its domain. Wu (Wu et al. 2002) uses transformation based learning for tagging affect in chat-conversation analysis. The method works by automatically generating rules for tagging, and then refining the rules by comparing them with a tagged corpus (ground truth) and iterating until no more

improvements in the rules can be made.

One successful technique is Liu (Liu, Lieberman, and Selker 2003), who created an affect tagger for email that uses a corpus of commen-sense rules. The premise is that to successfully identify affect, one must take into account everyday knowledge (which for them was manually encoded) that would not be captured in statistical approaches.

We differ from past approaches in two ways. First, we use different statistical methods based on computing n-grams, and tag sentences individually as opposed to tagging whole documents or paragraphs. Our approaches are based on techniques that have been successful in another domain, discourse act tagging. Second, our approaches are incorporated into a tagging tool which a user interacts with to better tag a transcribed session.

## Rochester Marriage Counseling Corpus

For our initial study we used a corpus of 45 annotated transcripts of conversations between married couples. These transcripts were provided by researchers from the Center for Future Health at the University of Rochester. At the start of each conversation, the moderator gave a husband and wife the task of discussing how they would cope if one of them developed Alzheimer's disease. Each transcript is broken into *thought units*, one or more sentences that represent how the speaker feels toward the topic. Each thought unit takes into account positive and negative words, comments on health, family, jobs, emotion, travel, sensitivity, detail, and many more. There are roughly two dozen tags, but we account for the five major ones: GEN, DTL, SAT, TNG, and ACK.

- **GEN** verbal content towards illness is vague or generic. Discussion tends to be about outcomes. It can also can indicate that the speaker does not take ownership of emotions. (i.e. "It would be hard," "I think that it would be important.")
- **DTL** speaker's verbal content is distinct with regards to illness, emotions, dealing with death, etc. Speaker tends to describe the process rather than the outcome. (i.e. "It would be hard for me to see you so helpless," "I would take care of you.")
- **SAT** statements about the task; couple discusses what the task is and how to go about it. (i.e. "I thought I would be the caregiver.")
- **TNG** tangents; statements that are not related to the task. Thought unit contains no emotional content related to the central issues of the task. (i.e. "talking about a friend with a disease.")
- **ACK** acknowledgments of the other speaker's comments. (i.e. "yeah" and "right")

The corpus contains a total of 14,390 unigrams and bigrams, of which 9,450 occur only once. There are 4,040 total thought units. The distribution of tags is as follows: GEN: 1964 (41.51%), ACK: 1075 (22.72%), DTL: 529 (11.18%), SAT: 337 (7.12%), TNG: 135 (2.85%).

## Approaches to Tagging

We studied two classes of approaches to automatic tagging. The first is based solely on building n-gram statistical models from training data. The second is a vector-based approach that builds sentence vectors from the n-grams in the training data.

### N-gram Based Approaches

The n-gram approaches tag a thought unit based on previously seen n-grams in the training data. The approach is motivated by the assumption that there are key phrases in each thought unit that identify which tag (emotion, attitude, etc.) should be used. Depending on the size of the n-grams being collected, a significant amount of word ordering can also be captured. However, n-gram models often lose many long range dependencies that extend beyond the $n$ length of the n-gram.

The following approaches all use n-grams ranging from unigrams to 5-grams. Any n-gram that appears only once in the training corpus is considered sparse and ignored. In addition, all unigrams, bigrams, and trigrams that appear only twice are thrown out. Due to the greater information content of the high n-grams, those that appeared twice are deemed helpful and are not ignored. The start of the thought unit was also considered a word (i.e. *null well* is a bigram in a sentence that begins with the word *well*).

**Naive Approach** The Naive Approach to tagging the thought units is the most basic approach, it simply selects the *best* n-gram that appears.

$$P(tag_i|utt) = \max_{j,k} P(tag_i|ngram_{jk})$$

Where $tag_i$ ranges over the set of available tags (GEN, DTL, SAT, TNG, ACK) and $ngram_{jk}$ is the $jth$ n-gram of length $k$ ($k$-gram) in the current thought unit $utt$ of the test set. The example below illustrates how the naive approach would tag a sentence, showing the n-gram with the highest probability of each n-gram length.

*I don't want to be chained to a wall*

| N-gram Size (tag) | Top N-gram | Probability |
|---|---|---|
| 1: (GEN) | don't | 0.665 |
| 2: (GEN) | to a | 0.692 |
| 3: (GEN) | *null* I don't | 0.524 |
| 4: (DTL) | don't want to be | 0.833 |
| 5: (DTL) | I don't want to be | 1.00 |

The highest n-gram is *I don't want to be* with 1.00 probability, indicating that this phrase always appeared with the DTL tag in the training set. Therefore, the unit is tagged DTL.

**Weighted Approach** The Weighted Approach builds upon the naive by assuming that higher n-grams provide more reliable information and that the sum of all n-gram probabilities will give a broader estimation. Each n-gram that appears in the given thought unit is multiplied by a weight assigned to the length of the n-gram. In more detail, the probability of a thought unit being tagged is:

$$P(tag_i|utt) = \sum_{k=0}^{m}((max_j P(tag_i|ngram_{jk})) * weight_k)$$

Where $m$ is the length of the longest n-gram (m=5 in this paper) and again, $ngram_{jk}$ is the $jth$ n-gram of length $k$. $weight_k$ is the weight of an n-gram of length $k$. This paper will refer to the weights $0.4, 0.4, 0.5, 0.8, 0.8$ (i.e. unigrams and bigrams are weighted $0.4$, trigrams are $0.5$, etc.) whenever the Weighted Approach is used.

We tested many different weights, each between 0 and 1 to try and obtain better results. However, the improvement was both minimal and approximately the same no matter which weights we chose (as long as the longer n-grams are weighted more than unigrams and bigrams).

As before, we will use the same example sentence to illustrate this approach. The top GEN and DTL n-grams are shown.

*I don't want to be chained to a wall*

| N-gram Size (tag) | Top N-gram | Prob |
|---|---|---|
| 1: (GEN) | don't | 0.665 |
| 1: (DTL) | want | 0.452 |
| 2: (GEN) | to a | 0.692 |
| 2: (DTL) | want to | 0.443 |
| 3: (GEN) | *null* I don't | 0.592 |
| 3: (DTL) | I don't want | 0.524 |
| 4: (GEN) | I don't want to | 0.27 |
| 4: (DTL) | don't want to be | 0.833 |
| 5: (GEN) | *null* I don't want to | 0.25 |
| 5: (DTL) | I don't want to be | 1.00 |

GEN sum (w/weights) 1.255
DTL sum (w/weights) 2.086

Ignoring the other possible tags in this example, it is easy to see that DTL gains ground in the higher probabilities, $0.833$ for the 4-gram and $1.00$ for the 5-gram. DTL's final sum of the n-gram weighted probabilities is clearly higher and the sentence is correctly tagged DTL.

The added weights do not differ much from the Naive Approach in this example, but one of the many cases where it does differ can be seen here:

and really decide if there were places we wanted

The Naive Approach pulls out the unigram *really* as the highest single probability and tags it GEN. However, the Weighted Approach sums the max of each n-gram for the tags and DTL wins because *decide if there* has a higher probability than GEN's top trigram. Trigrams are weighted higher than the lower n-grams where GEN is considered more likely, so the Weighted Approach chooses DTL while the naive approach would incorrectly choose GEN.

**Lengths Approach** The Lengths Approach also builds upon the Naive Approach by adding the lengths of each thought unit as a type of weight (much like the Weighted Approach) to compute the maximum n-gram probabilities. During training with the training set, we count the number of words in each tag's thought units. By calculating the average utterance length for each tag and its corresponding standard deviation, we can obtain a length weight for each new thought unit in the test corpus.

$$lenWeight_{ts} = \frac{e^{(-(n_s - n_t)^2)/(2dev_t^2)}}{\sqrt{2\pi} * dev_t}$$

Where $n_s$ is the number of words in utterance $s$, $n_t$ is the average word length for the tag $t$, and $dev_t$ is the length standard deviation for tag $t$. The length weights from our example sentence of *nine* words are as follows:

| GEN | DTL | ACK | SAT | TNG |
|---|---|---|---|---|
| 0.0396 | 0.0228 | 0.0004 | 0.0613 | 0.0354 |

As you can see, the weights are relatively equal except for the ACK and SAT tags. Acknowledgment phrases tend to be very short and concise, one or two words long, so this low weight for a sentence of nine words is consistent with our method.

Once these tag weights are computed, we choose our tag by the following:

$$P(tag_i|utt) = (\max_{j,k} P(tag_i|ngram_{jk})) * lenWeight_{im}$$

Where $m$ is the word length of $utt$. Below shows how this method influences our example utterance. Again, only the top n-gram for each is shown.

*I don't want to be chained to a wall*

| | | | |
|---|---|---|---|
| 1: (GEN) | don't | .665 * .0396 = | .026 |
| 2: (GEN) | to a | .692 * .0396 = | .027 |
| 3: (GEN) | *null* I don't | .592 * .0396 = | .021 |
| 4: (DTL) | don't want to be | .833 * .0228 = | .019 |
| 5: (DTL) | I don't want to be | 1.00 * .0228 = | .023 |

The highest weighted n-gram is now the bigram, *to a*, with $0.027$ final probability. The sentence is tagged GEN. The length weight changed the Naive Approach's result to GEN. The majority of cases where the Lengths Approach differs the most can be found in ACK thought units. For example, the utterance, *I don't either* should be tagged ACK, but none of the three words are strong indicators of the ACK tag and the training set does not contain this trigram. However, the length weight of ACK is $0.174$ while the nearest tag, GEN, is $0.029$. The Naive Approach normally considers this sentence three times more likely to be GEN than ACK. However, the length weight takes the short length of this utterance and correctly weights the ACK probability higher than GEN.

**Weights with Lengths Approach** After finding only minor improvements over both the Weighted and Lengths Approaches, we combined the two together. We continue our example by recalling the results of the Weighted Approach followed by the addition of the lengths weight:

| | | |
|---|---|---|
| GEN sum (w/weights) | 1.255 | |
| DTL sum (w/weights) | 2.086 | |
| GEN weight/length | 1.255 | * 0.0396 = 0.0497 |
| DTL weight/length | 2.086 | * 0.0228 = 0.0476 |

Adding the length weight to this example reverses the Weighted Approach's DTL tag to a GEN tag. This occurs because DTL utterances typically are very long, while GEN utterances are very often under $10$ words long (as is our example).

**Analytical Approach** We found that many ACK utterances were being mis-tagged as GEN. Many of these were grounding utterances that repeated some of what was said in

the previous utterance. For example, the second utterance below is mis-tagged:

B - so then you check that your tire is not flat
A - check the tire

This is a typical example of grounding where speaker A repeats a portion of B's last utterance in order to indicate understanding. Instead of adding another weight to our already growing list on the Naive Approach, we created a model that would take repeated words and the length of the two utterances into account (the repeated phrase is usually shorter than the original).

$$P(w_1|T) * P(w_2|T) * ... * P(w_n|T) * P(R_{w_1}|O_{w_1}, L, L_p, T) * ... * P(R_{w_n}|O_{w_n}, L, L_p, T) * P(L|T) * P(T)$$

Where $w_i$ is a unigram in the utterance and $0 \leq i < n$ where $n$ is the length of the utterance, $O_{w_i}$ is a unigram occurring in the previous utterance, $R_{w_i}$ is the repeated unigram (i.e. the unigram appeared in this utterance as well, $L$ is the length of the current utterance, and $L_p$ is the length of the previous utterance. The third line of the equation is the length weight brought over from the Lengths Approach. Due to the obvious sparseness of the data for such an ambitious statistic, we put each unigram in one of four buckets according to its number of occurrences in the training data (5, 30, 100, infinity). The sentence lengths are thrown into only two buckets (2, 100000). Since most acknowledgements are two or less words in length, this statistic should help find the majority of them while the repeated unigrams will find the rest.

This method produced worse results than the Naive Approach. Other bucket sizes were experimented with, but none significantly altered the result. This may be a direct result from the frequency of tags GEN and ACK. These two tags make up 64% of the 5 tags that occur in the corpus and the probability $P(L|T)$ heavily favors one of the two tags (ACK dominates short sentences and GEN dominates the rest). Unfortunately it seems to pull down the overall results when it is a factor in any calculation.

## Information Retrieval-based Approaches

We looked at two vector-based methods for automatically tagging the utterances in the marriage corpus. The first method is based on the Chu-Carroll and Carpenter routing system (Chu-Carroll and Carpenter 1999). The second method dispenses with matrix construction of exemplar vectors and compares test sentences against a database of sentences.

**Chu-Carroll and Carpenter** Chu-Carroll and Carpenter's domain is a financial call center with 23 possible caller destinations. The algorithm to route callers to the appropriate destination is summarized here. The caller's voice request is sent to a parser and a routing module. If the module generates only one destination, then the call is routed to that department. If more than one destination is generated, the system tries to generate a disambiguation query. If such a query cannot be generated then the call defaults to a human operator (calls are also sent to human operators if the routing module does not generate a destination). When the clarifying query is generated, the algorithm is repeated with the caller's disambiguating response to the query.

The routing module is the most important part of the system. It consists of a database of a large collection of documents (previous calls) where each document is a vector in $n$-dimensional space. A query is a sentence transformed into a single vector and compared to each document (or destination) in the database. The document most similar to the query vector is the final destination.

Creation of the database of documents consists of a training process. Each word of the input is filtered morphologically, stop-words are removed, and all unigrams, bigrams and trigrams are extracted. The database or term-document matrix is a $M \times N$ matrix where $M$ is the number of salient terms and $N$ is the number of destinations. $A_{t,d}$ is the number of times term $t$ occurs in calls to destination $d$. This matrix is then normalized for n-grams:

$$B_{t,d} = \frac{A_{t,d}}{\sqrt{\sum_{1 \leq e \leq n} A_{t,e}^2}}$$

A second normalizing metric, inverse document frequency (IDF), is also employed to lower the weight of a term that occurs in many documents since it is not a good indicator of any destination:

$$IDF(t) = log_2 \frac{n}{d(t)}$$

where $n$ is the number of documents in the corpus and d(t) is the number of documents containing term $t$. Each entry in the matrix thus becomes:

$$C_{t,d} = IDF(t) \times B_{t,d}$$

Query matching consists of transforming the input call to a vector in the same manner as above. A cosine distance metric is then used to compare this vector against the $n$ destinations in the matrix.

**Method 1: Routing-based Method** Our method is a modified version of Chu-Carroll and Carpenter's algorithm. Stop-word filtering is not done, so some common stop words such as "hmmm" or "uh" are included in the list of n-grams. We use a 2-gram model, so the term extraction phase generates all unigrams and bigrams. The same weighting principles and IDF metrics are employed for the matrix construction phase.

One modification to the algorithm was the introduction of the entropy (amount of disorder) of each term. If a term is found in several documents then it exhibits low entropy, or a low amount of disorder. On the other hand, terms such as *yeah* or *right*, which appear in several tags, are bad exemplars and should be pruned from the database given their entropy. We use the following formula for determining the entropy of a term:

$$entropy(t) = -\sum_{1 \leq e \leq n} \frac{A_{t,e}}{\sum_{1 \leq f \leq n} A_{t,f}} \times \frac{logA_{t,e}}{\sum_{1 \leq f \leq n} A_{t,f}}$$

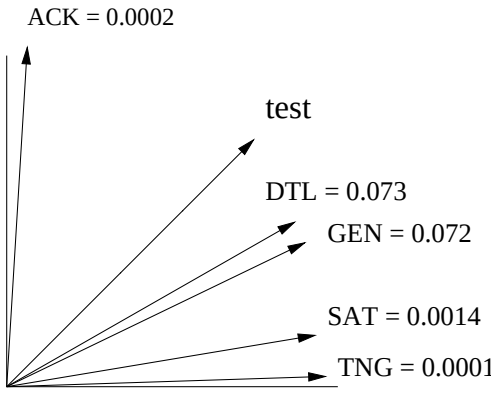


Figure 1: IR Method 1 Results of Test Sentence (not to scale)

| Length | GEN | ACK | TNG | DTL | SAT |
|---|---|---|---|---|---|
| 1-2 | 218 | 991 | 10 | 13 | 42 |
| 3-4 | 389 | 140 | 26 | 29 | 68 |
| 5-6 | 400 | 49 | 31 | 68 | 55 |
| 7-8 | 312 | 20 | 15 | 72 | 63 |
| over | 1107 | 19 | 54 | 426 | 126 |

Figure 2: Distribution of the lengths of thought units for each tag.

A pruning threshold must be determined in order to use entropy to eliminate unhelpful terms. In the marriage counseling corpus, the maximum entropy for any term was 2.9. We repeated the testing algorithm starting with this cutoff value and decrementing by 0.1 for each new test.

Figure 1 shows how this method would assign the DTL tag to the sentence, *I don't want to be chained to a wall*.

After the matrix of canonical vectors is created, the test sentence (in vector form) is compared to each of the vectors. The one that is closest to our example thought unit is DTL, if only by a small margin over GEN. Hence it is selected, correctly, as the tag.

It should be noted that the diagram is purely conceptual. In reality, the cosine test is done over an $n$-dimensional space (as opposed to 2) where $n$ in this test is 38,636.

In addition, we created two more vectors to raise the tagging accuracy: sentence length and repetition. The intuition behind the former is that tags tend to be correlated with sentences of a certain length. For example, ACK utterances are usually one or two words in length because most of them are phrases such as "yeah" or "yeah ok." On the other hand, tags that tend to communicate more, such as DTL, would have longer sentences. The distribution can be seen in figure 2.

Additional vectors are added to the matrix for each sentence length and the frequencies above are encoded in these vectors using the normalization schemes. During the test phase, the length of the test sentence is included in its vector representation.

The intuition behind the repetition metric is that ACK's

| Cosine Score | Tag | Sentence |
|---|---|---|
| 0.6396 | SAT | Are we supposed to get them? |
| 0.6030 | GEN | That sounds good) |
| 0.6030 | TNG | That's due to my throat |
| 0.5583 | DTL | But if I said to you I don't want to be Ruth and I don't want to get anywhere close to being Ruth, that I would expect that you would respect my wishes |
| 0.5449 | DTL | If it were me, I'd want to be a guinea pig to try things |

Figure 3: IR Method 2 Results of Test Sentence

can be better modelled by taking into account repeated words from the previous thought unit. If key words are repeated then it is likely that the second speaker used those words to signal a backchannel. Repetition, or percent overlap, is calculated by dividing the number of n-grams in the intersection of the current and previous utterance by the number of current utterance n-grams. We use a "bin" system as used in the sentence length metric: we make new tags of 0% repetition, 1- 25%, 26-50%, 51-75%, 76-100% and 100% overlap. All unigrams are thrown out under the assumption that they won't add any useful information since higher order n-grams are better predictors. Keeping them results in a 1% drop in performance.

**Method 2: Direct Comparison of Sentence Vectors** In this method we are directly comparing a test vector to a database. Unlike the Chu-Carroll and Carpenter algorithm, there is no construction of n-grams and normalizing of data.

Each thought unit is converted to a vector in which the index of the vector is a term. The cosine test is again used to determine similarity, but here it is used somewhat differently. We compare our test vector to every other vector in the database. The ten highest cosine scores (highest similarities) are kept and used to calculate the best tag. First, each of the ten thought units' tags are normalized to 100%, so if a thought unit had multiple tags and tag counts of ((GEN 2) (DTL 1)), the result would be (GEN 66.67%) and (DTL 33.33%). Next, these percents are normalized by multiplying by the respective cosine score and dividing by the sum of all ten cosine scores. This makes vectors that are very close to the test vector more significant (weighting). Finally, each tag is summed across the ten thought units and the one with the highest sum is returned as the most likely tag. Using the same test example, the top five sentences selected using this method are shown in figure 3. Weighting the tags appropriately, the scores are: DTL = 0.3741, SAT = 0.2169, GEN = 0.2045, TNG = 0.2044, ACK = 0. So DTL is correctly selected.

## Evaluations

The different methods discussed in this paper were evaluated based on the six-fold cross validation of their percent accuracy. In regards to the Marriage Counseling Corpus, the

| Naive | Weighted | Lengths | Weight w/Length | Analyt. |
|---|---|---|---|---|
| 66.80% | 67.43% | 64.35% | 66.02% | 66.60% |

Figure 4: Percent Correct comparison of the five n-gram approaches for the UR Marriage Corpus

| Prune-1 | Entropy | IDF | Accuracy |
|---|---|---|---|
| X | X | | 66.16% |
| | X | | 65.69% |
| | | | 65.17% |
| X | X | X | 61.56% |
| | X | X | 61.40% |
| X | | X | 61.37% |

Figure 5: IR Method 1 Results

corpus is divided into six even portions and cross validation was used (train on five, test on the remaining one) to extract the final tagging accuracy percentage for each approach. All six combinations of the validation were tested and the average of the six is used as the final percentage.

**N-Gram based Approaches:** The five n-gram approaches performed relatively the same on the marriage corpus. The results are given in figure 4. The Weighted Approach performed the best, scoring almost one percent above the Naive Approach and more so above the others. The Naive, which simply takes the n-gram with the highest probability, performed better than the other added metrics.

**IR Method 1** We evaluated several instantiations of the modified Chu-Carroll and Carpenter algorithm by toggling these pruning methods:

- **Prune-1**: terms that occur only once are pruned
- **Entropy**: the above entropy method is used
- **IDF**: terms are pruned using IDF

To determine which values were best, we tested all six combinations of our three pruning methods. The X's denote which pruning method was used.

These results show that only using the entropy model (with or without terms that occur only once) offers roughly a 4% advantage over the IDF method. The top two combinations of models (using entropy with/without Prune-1) were then run on the entire corpus with the model eliminating values that occur only once. This resulted in an average of 66.16% over the 6 cross-validation sets, while the model not using Prune-1 averaged 65.69%.

| Prune-1 | Entropy | IDF | Accuracy |
|---|---|---|---|
| | X | N/A | 63.10% |
| X | X | N/A | 65.25% |

Figure 6: IR Method 2 Results

| Metric | Accuracy |
|---|---|
| Sentence Length | 66.76 |
| Repetition: No Unigrams | 66.39 |
| Repetition: All N-grams | 65.59 |

Figure 7: IR Extension Metrics

Using the best instantiation (Prune-1 and Entropy) from the first six tests, we made new n-grams for sentence length: sentences less than 2, of length 3 or 4, of length 5 or 6, and so forth so the final new n-gram was sentences 10 words or longer. We found a marginal increase in performance at 66.76%.

**IR Method 2** The result (using cross-validation as in Method 1 and the best instantiation: Prune-1 and Entropy, but no IDF) is an overall average of 63.16%, slightly lower than Method 1.

## Discussion

Of the two vector approaches, the Routing-Based Method achieved the best result with almost a $1\%$ improvement over the Direct Method. Both approaches did best when Prune-1 and Entropy were factored in. Pruning terms that appeared only once helped in most cases (except for IDF, adding Prune-1 lowered the score); this intuitively makes sense. Words with only one occurrence do not convey accurate information and tend to bring the accuracy down. The entropy factor also improved performance, adding $1.5\%$ to IDF and almost $2\%$ to the combination of Prune-1 and IDF. When Prune-1 and entropy were combined, they resulted in the highest Routing-Based and Direct Method results. However, with a score of $66.16\%$, the Routing-Based Prune-1 and Entropy scored almost $1\%$ over the Direct's $65.25\%$.

The N-Gram Methods proved to be much more versatile than we initially thought. The Naive Approach does surprisingly well, scoring a half percentage higher than the best Routing-Based method, $66.80\%$. We attempted several additions to the Naive to improve performance, but nothing significantly improved our results. Giving the n-grams different weights produced the most accurate results with a 6-fold cross validation score of $67.43\%$. Putting the lengths of the thought units into consideration actually hurt the results in the marriage corpus. The Analytical Approach looks the most promising since it seems to state the length of utterances and repeated words more precisely than the Naive Approach; however, the Analytical performs relatively the same as the Naive, scoring $0.2\%$ lower.

The Weighted Approach performed the best on the marriage corpus out of all the methods discussed in this paper. This result goes against intuition and shows us that a simple n-gram comparison between the training set and the testing set performs as well and even better than the more complicated vector based approaches. Further, adding different weights to the Naive often hurts and very rarely improves the performance. Choosing the most indicative n-gram for the correct tag produces the most accurate results.

| Approach | % Correct |
|---|---|
| Base Model | 58.42% |
| Entropy & IDF | 57.26% |
| Repeated Words | 60.93% |
| Length | 60.92% |
| Repeated Words & Length | 60.93% |
| Direct Approach | 63.53% |

Figure 8: Comparison of Information Retrieval Approaches for Switchboard Corpus

### Switchboard Corpus Evaluation

**Switchboard Data Set** To further determine the effectiveness of a tagging system it is necessary to compare it to other methods on a common corpus. We selected the Stolcke et al. modified Switchboard Corpus of spontaneous human-to-human telephone speech (Godfrey, Holliman, and McDaniel 1992) from the Linguistic Data Consortium. Unlike the Marriage Counseling data set, the Switchboard set is composed of conversations of random topics as opposed to planned task-oriented ones. In addition, the data set has a much richer tagging set, being composed of 42 Dialog Acts (DA's) that are the Switchboard's version of thought unit tags. The five most prominent tags, which comprise 78% of the corpus, are (and their percentage of the corpus): Statement (36%), Backchannel/Acknowledgement (19%), Opinion (13%), Abandoned/Uninterruptable (6%), Agreement/Accept (5%). The tagged corpus consists of 205,000 utterances and 1.4 million words making it significantly larger than any other similarly tagged corpora.

**Evaluation Method and Results** We split the Switchboard Corpus into six subsets just as we did with the Marriage Corpus in order to perform the six fold cross-validation test. All the results are averages over the six combinations of training on five and testing on one subset.

Figure 8 shows the results of the vector-based approaches on the Switchboard Corpus. The base model uses the original Chu-Carroll and Carpenter formalism of pruning entries that occur once and using IDF. The second method adds the entropy metric. The following approaches incorporate sentence length and word repetition as n-grams in the vector. As in the Marriage Counseling corpus, these improve accuracy slightly. The Direct Method performs significantly better than it did in the other corpus, mostly due to the fact there is more data to compare with.

Figure 9 shows the results of the five n-gram approaches described in section . The performance is similar to that on the Marriage Corpus (figure 4), but the Analytical Approach experiences a drop of $7.8\%$. This can be attributed to an even denser clustering of tags in the Switchboard corpus, where the three most frequent tags appear $71\%$ of the time in the corpus. The length probability in the Analytical Approach favors them greatly (so it is often used instead of the correct tag). We do, however, see that the Weighted with Lengths Approach out-performs the other approaches. It correctly tags $1.6\%$ more than the closest approach, breaking $70\%$.

| Base Naive | Weighted | Lengths | Weight w/Length | Analyt. |
|---|---|---|---|---|
| 68.41% | 68.77% | 69.01% | 70.08% | 61.40% |

Figure 9: Percent Correct comparison of the five n-gram approaches for Switchboard Corpus

## CATS Tool

The culmination of this research has led to the development of an easy to use tool that is based off of the Naive Approach. Written in Java for multi-platform portability, this automated tagging system (CATS) employs the Lengths Approach, but uses only unigrams and bigrams. One of the main concerns during development of CATS was the hardware limitations that exist for a helpful tool. We found that the Naive Approach performs as well as the vector method approaches and requires less computing power. In addition, using just unigrams and bigrams perform almost as well. This is most likely the result of sparse data. As a result, CATS employs a unigram-bigram naive approach with length weights. Computing time is acceptable and the results are comparable.

The program's main goal is ease of use for the researcher. As can be seen in figure 10, the main tagging attributes (case #, speaker, tag, text) are contained in separate windows to keep a clean workspace. Features such as auto-completion, tag flexibility, model customization, and others make the program a solid text editor on top of an automated tagger. One can build a naive method model off of data by loading any text file with thought units and tags into the CATS program. A simple click of the button builds the model. To tag new text based off of this model, a new file of thought units (without tags obviously) can be loaded and automatically tagged with another click of the mouse.

CATS splits the new thought units into unigrams and bigrams and draws its tags using the naive length weighted approach discussed in this paper. A certainty percentage, the tagger's assessment of its tag choices, is given with each tag as well. The percent is based off of the second highest tag choice for the thought unit (high / (high $+$ 2nd high) $*$ 100). Low certainty measurements are colored red to bring the researcher's attention to a potentially incorrect tag.

Currently, there is no program that can train on data with a click of the mouse and instantly build a statistical model based on the given tags. Tagging new data is completed in seconds with CATS, saving the researcher weeks or even months of work. We have worked closely with the Center for Future Health and are seeing results of 80 and even 90% accuracy (based on smaller tag sets than the one discussed in this paper) with this tool.

The fields of anthropology, linguistics, and psychology frequently employ several undergraduates and graduates to tag speech dialogues for information content. The process is extremely tedious and can take months. However, CATS has the potential to cut the process down to a few seconds. Once an initial dataset of adequate size is tagged, CATS trains itself on the data and is able to accurately and quickly tag new data. We feel the tool will be greatly accepted in the research

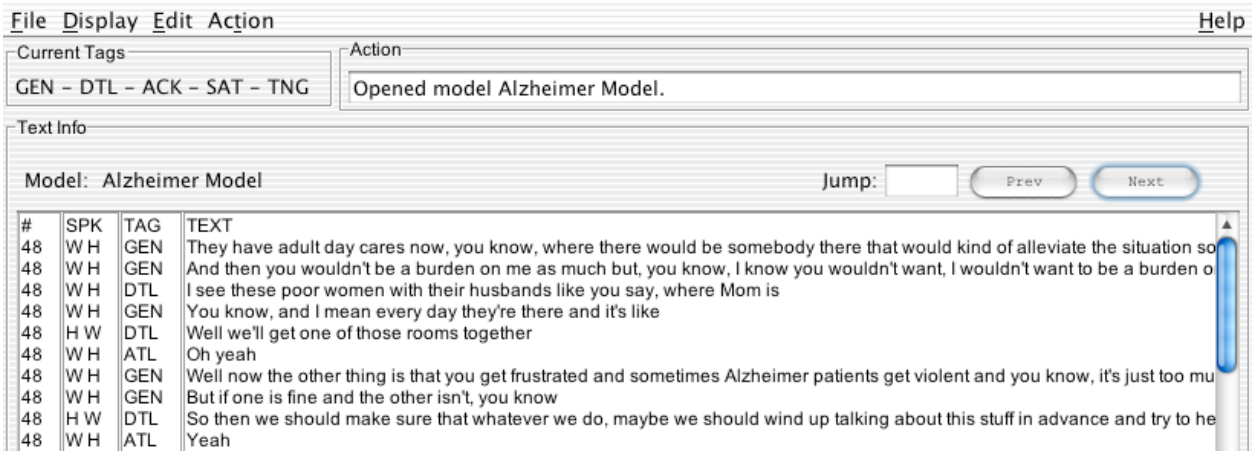


Figure 10: CATS tags data and gives a certainty measure to its left. Low certainty tags can be glanced over by the researcher to assure correctness.

community.

## Related Work

Stolcke (Stolcke et al. 2000) developed a dialogue model that is "based on treating the discourse structure of a conversation of a hidden Markov model and the individual dialogue acts are modeled via a dialogue act n-gram." Whereas word n-grams form the basis of analysis for both metrics discussed in this paper, word n-grams are part of a set of methods used to implement the statistical dialogue grammar. Decision trees and neural nets are also used. In addition, while our methods only look at written text, Stolcke et al. have models that integrate prosody and speech.

Their model fared about the same as the Lengths and Weights metric, tagging 71% correctly. In comparison, their study of human annotators had human annotation scoring 84% and a baseline of 35%.

## Conclusion

We have described several vector-based and n-gram approaches to automatically tagging text. We have described a new domain involving married couples in which our approaches perform reasonably well, but without much variation from each other. The computationally expensive vector based approaches for recognizing the emotional content of a speaker is outperformed by much more simple n-gram approaches. We have also described a platform independent tagging tool that can be employed by researchers to learn tagging patterns and automatically tag large corpora of dialogues or written text. Finally, we have shown that our approaches perform similarly on the Switchboard Corpus and have provided scores for others to compare future work on automatic tagging.

## Acknowledgments

We would like to thank Dr. Cleveland Shields for insight and the evaluation dialogues. This work was supported in part by ONR grant 5-23236 and the National Institute on Deafness and Other Communication Disorders grant T32 DC00035-10.

## References

Boucouvalas, A. C., and Zhe, X. 2002. Text-to-emotion engine for real time internet communication. In *International Symposium on CSNDSP 2002*.

Chu-Carroll, J., and Carpenter, B. 1999. Vector-based natural language call routing. *Computational Linguistics*.

Godfrey, J.; Holliman, E.; and McDaniel, J. 1992. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*.

Goertzel, B.; Silverman, K.; Hartley, C.; S. Bugaj, S.; and Ross, M. 2000. The baby webmind project. In *Proceedings of AISB 2000*.

Hodson, J. H.; Shields, C. G.; and Rousseau, S. L. 2003. Disengaging communication in later-life couples coping with brest cancer. *Family Systems and Health*.

Liu, H.; Lieberman, H.; and Selker, T. 2003. A model of textual affect sensing using real-world knowledge. In *Proceedings of the Seventh International Conference on Intelligent User Interfaces*.

Shields, C. 1997. Annotation scheme. for Center for Future Health.

Stolcke, A.; Ries, K.; Coccaro, N.; Shriberg, E.; Bates, R.; Jurafsky, D.; Taylor, P.; and Martin, R. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*.

Wu, T.; Khan, F. M.; Fisher, T. A.; Shuler, L. A.; and Pottenger, W. M. 2002. Posting act tagging using transformation-based learning. In *Proceedings of the Workshop on Foundations of Data Mining and Discovery, IEEE International Conference on Data Mining (ICDM'02)*.