Homograph Disambiguation in Text-to-speech Synthesis

David Yarowsky

ABSTRACT This chapter presents a statistical decision procedure for lexical ambiguity resolution in text-to-speech synthesis. Based on decision lists, the algorithm incorporates both local syntactic patterns and more distant collocational evidence, combining the strengths of decision trees, N-gram taggers and Bayesian classifiers. The algorithm is applied to 7 major types of ambiguity where context can be used to choose a word's pronunciation.

1 Problem Description

In speech synthesis, one frequently encounters words and numbers whose pronunciation cannot be determined without context. Seven major types of these homographs will be addressed here:

1. *Different Part of Speech*: The largest class of pronunciation ambiguity consists of homographs with different parts of speech: *Three <u>lives</u> were lost* vs. *One <u>lives</u> to eat.* These cases can typically be resolved through local syntactic patterns.

2. *Same Part of Speech*: A word such as *bass* or *bow* exhibits different pronunciations with the same part of speech, and thus requires additional "semantic" evidence for disambiguation.

3. *Proper Names* such as *Nice* and *Begin* are ambiguous in capitalized contexts, including sentence initial position, titles and single-case text.

4. *Roman Numerals* are pronounced differently in contexts such as *Chapter III* and *Henry III.*

5. *Fractions/Dates* such as *5/16* may be pronounced as *five-sixteenths* or *May 16th.*

6. *Years/Quantifiers*: Numbers such as *1750* tend to be pronounced as *seventeen-fifty* when used as dates, and *one-thousand seven-hundred*

and fifty when used before measure words, as in *1750 miles*. Related cases include the distinction between *the 727 pilot* and *727 people*.

7. *Abbreviations* may exhibit multiple pronunciations, such as *St.* (Saint or Street) and *Dr.* (Doctor or Drive).

Some homographs exhibit multiple categories of ambiguity. For example, *lead* has different pronunciations when used as a verb and noun, and between two noun senses (*He followed her lead* verses *He covered the hull with lead*). In general, we will resolve the part-of-speech ambiguity first, and then resolve the additional semantic ambiguity if present.

2 Previous Approaches

N-gram taggers[Jel85][Chu88][Mer90] may be used to tag each word in a sentence with its part of speech, thereby resolving those pronunciation ambiguities that correlate with part-of-speech ambiguities. The AT&T TTS synthesizer [SHY92] uses Church's PARTS tagger for this purpose. A weakness of these taggers is that they are typically not sensitive to specific word associations. The standard algorithm relies on models of part-of-speech sequence, captured by probabilities of part-of-speech bigrams or trigrams, to the exclusion of lexical collocations. This causes difficulty with cases such as *a ribbon wound around the pole* and *a bullet wound in the leg*, which have identical surrounding part-of-speech sequences and require lexical information for resolution. A more fundamental limitation, however, is the inherent myopia of these models. They cannot generally capture longer distance word associations, such as between *wound* and *hospital*, and hence are not appropriate for resolving many semantic ambiguities.

Bayesian classifiers[MW64] have been used for a number of sense disambiguation tasks[GCY94], typically involving semantic ambiguity. In an effort to generalize from longer distance word associations regardless of position, an implementation proposed in [GCY92] characterizes each token of a homograph by the 100 words nearest to it, treated as an unordered bag.[1] Although such models can successfully capture topic-level differences, they lose the ability to make distinctions based on local sequence or sentence structure. In addition, these models have been greatly simplified by assuming that occurrence probabilities of content words are independent of each other, a false and potentially problematic assumption that tends to yield inflated probability values. One can attempt to model these dependencies (as in [BW94]), but data sparsity problems and computational constraints can make this difficult and costly to do.

[1] Leacock et al. have pursued a similar bag-of-words strategy, using an IR-style vector space model and neural network [LTV93].

Decision Trees [BFOS84][BDDM91] can be effective at handling complex conditional dependencies and non-independence, but often encounter severe difficulties with very large parameter spaces, such as the highly lexicalized feature sets frequently used in homograph resolution.

Current Algorithm: The algorithm described in this paper is a hybrid approach, combining the strengths of each of these 3 general paradigms. It was proposed in [SHY92] and refined in [Yar94]. It is based on the formal model of **decision lists** in [Riv87], although feature conjuncts have been restricted to a much narrower complexity, namely word and class trigrams. The algorithm models both local sequence and wide context well, and successfully exploits even highly non-independent feature sets.

3 Algorithm

Homograph disambiguation is ultimately a classification problem, where the output is a pronunciation label for an ambiguous target word and the feature space consists of the other words in the target word's vicinity. The goal of the algorithm is to identify patterns in this feature space that can be used to correctly classify instances of the ambiguous word in new texts.

For example, given instances of the homograph *lead* below, the algorithm should assign the appropriate label /lid/ or /lɛd/.

Pronunciation	Context		
(1) lɛd	.. it monitors the	*lead*	levels in drinking
(1) lɛd	... median blood	*lead*	concentration was
(1) lɛd	.. found layers of	*lead*	telluride inside ..
(1) lɛd	... conference on	*lead*	poisoning in ...
(1) lɛd	.. strontium and	*lead*	isotope zonation ..
(2) lid	maintained their	*lead*	Thursday over ...
(2) lid	.. to Boston and	*lead*	singer for Purple
(2) lid	Bush a 17-point	*lead*	in Texas , only 3
(2) lid	his double-digit	*lead*	nationwide . The
(2) lid	the fairly short	*lead*	time allowed on ..

The following sections will outline the steps in this process, using the individual homographs *lead* (lid/lɛd) and *bass* (beɪs/bæs) as examples. The application of this algorithm to large classes of homographs such as fractions vs. dates is described in Section 4.

Step 1: Collect and Label Training Contexts

For each homograph, begin by collecting all instances observed in a large text corpus. Then label each example of the target homograph with its correct pronunciation in that context. Here this process was partially automated, using tools that included a sense disambiguation system based on semantic classes [Yar92].

For this study, the training and test data were extracted from a 400-million-word corpus collection containing news articles (AP newswire and Wall Street Journal), scientific abstracts (NSF and DOE), 85 novels, two encyclopedias and 3 years of Canadian parliamentary debates, augmented with e-mail correspondence and miscellaneous Internet dialogues.

Step 2: Measure Collocational Distributions

The driving force behind this disambiguation algorithm is the uneven distribution of *collocations* (word associations) with respect to the ambiguous token being classified. For example, the following table indicates that certain word associations in various positions relative to the ambiguous token *bass* (including co-occurrence within a $\pm k$ word window[2]) exhibit considerable discriminating power[3].

Position	Collocation	beɪs	bæs
Word to the right (+1 w)	bass *player*	105	0
	bass *fishing*	0	94
	bass *are*	0	15
Word to the left (-1 w)	*striped* bass	0	193
	on bass	53	0
	sea bass	0	47
	white bass	0	26
	plays bass	16	0
Within ±20 words ($\pm k$ w)	*fish* (in ±20 words)	0	142
	guitar (in ±20 words)	136	0
	violin (in ±20 words)	49	0
	river (in ±20 words)	0	48
	percussion (in ±20 words)	41	0
	salmon (in ±20 words)	0	38

The goal of the initial stage of the algorithm is to measure a large and varied set of collocational distributions and select those which are most useful in identifying the pronunciation of the ambiguous word.

In addition to raw word associations, the present study also collected collocations of lemmas (morphological roots), which usually provide more

[2]Several different context widths are used. The ±20 words employed here is a practical width for many applications. The issues involved in choosing appropriate context widths are discussed in [GCY92].

[3]Such skewed distributions are in fact quite typical. A study in [Yar93] showed that $P(pronunciation|collocation)$ is a very low entropy distribution. Certain types of content-word collocations seen only *once* in training data predicted the correct pronunciation in held-out test data with 92% accuracy.

succinct and generalizable evidence than their inflected forms, and part-of-speech sequences, which capture syntactic rather than semantic distinctions in usage.[4] A richer set of positional relationships beyond adjacency and co-occurrence in a window is also considered, including trigrams and (optionally) verb-object pairs. The following table indicates the pronunciation distributions observed for the noun *lead* for these various types of evidence:

Position[5]	Collocation	lɛd	lid
+1 L	lead *level/N*	219	0
-1 W	*narrow* lead	0	70
+1 W	lead *in*	207	898
-1W,+1W	*of* lead *in*	162	0
-1W,+1W	*the* lead *in*	0	301
+1P,+2P	lead *,* <*NOUN*>	234	7
±k W	*zinc* (in $\pm k$ words)	235	0
±k W	*copper* (in $\pm k$ words)	130	0
-V L	*follow/V* + lead	0	527
-V L	*take/V* + lead	1	665

Step 3: Compute Likelihood Ratios

The discriminating strength of each piece of evidence is measured by magnitude of the the log-likelihood ratio:

$$Abs(Log(\frac{P(Pronunciation_1|Collocation_i)}{P(Pronunciation_2|Collocation_i)}))$$

The collocation patterns most strongly indicative of a particular pronunciation will have the most extreme log-likelihood. Sorting evidence by this value will list the strongest and most reliable evidence first.

Note that the estimation of $P(Pronunciation_j|Collocation_i)$ merits considerable care. Problems arise when an observed count in the collocation distribution is 0, a common occurrence. Clearly the probability of seeing *zinc* in the context of the /lid/ pronunciation of *lead* is not 0, even though no such collocation was observed in the training data. Finding a more accurate probability estimate depends on several factors, including the size

---

[4] The richness of this feature set is one of the key reasons for the success of this algorithm. Others who have very productively exploited a diverse feature set include [Hea91], [Bri93] and [DI94].

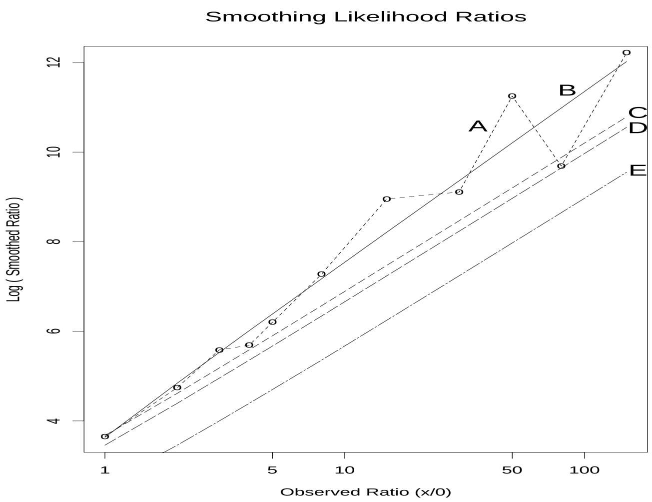
[5] Position markers include +1 (token to the right), −1 (token to the left), $\pm k$ (co-occurrence in $\pm k$-token window) and -V (head verb). Possible types of objects at these positions include W (raw words), P (parts of speech) and L (lemmas – a class of words consisting of different inflections of the same root, such as *take/V= takes, took, taken, take, taking*).

of the training sample, nature of the collocation (adjacent bigrams, verb-object pairs or wider context), our prior expectation about the similarity of contexts, and the amount of noise in the training data.

Several smoothing methods have been explored in this work, including those discussed in [GCY92]. The preferred technique is to take all instances of the same raw frequency distribution (such as 2/0 or 10/1), and collectively compute a smoothed ratio that better reflects the true probability distribution. This is done by holding out a portion of the training data and computing the mean observed distribution there (e.g. 1.8/0.2) for all of the collocations that have the same raw frequency distribution in the first portion (e.g. 2/0). This mean distribution in the held-out data is a more realistic estimate of the distribution expected in independent test data, and hence gives better predictive power and better probability estimates than using the unsmoothed values.

Smoothed ratios are very sensitive to type of collocation being observed. A 1/0 observed ratio for adjacent content words has a smoothed ratio of .92/.08, while a 1/0 observed ratio for function-word collocations 5 to 50 words away has a smoothed ratio close to .5/.5, indicating that a 1/0 training distribution is essentially noise here, with little predictive value.

The process of computing distributions of the form $x/0$ (and also $x/1$, etc.) for all values of x can be simplified by the observation that the mapping from observed ratios to smoothed ratios tends to exhibit a log-linear relationship when other factors such as distance are held constant. This is shown in the figure below for observed $x/0$ distributions for adjacent content word collocations. The points and jagged line (A) are the empirically observed values in held-out data. (B) constitutes the least-squares fit of this data, which is a reasonable fit, especially given that the empirical values are poor estimates for large x due to limited sample points in this range.



Satisfactory results may be obtained, however, by a much simpler smoothing procedure. Adding a small constant α to the numerator and denominator ($x/z \rightarrow (x+\alpha)/(z+\alpha)$) roughly captures the desired smoothing behavior, as shown in lines C ($\alpha = .085$), D ($\alpha = .1$), and E ($\alpha = .2$) above. The constant α is determined empirically for the different types of collocation and distance from the target word. However, the value does not vary greatly for different homographs, so adequate performance can be achieved by reusing previous values rather than estimating them afresh from held-out training data.

Step 4: Sort by Likelihood Ratio into Decision Lists

Preliminary decision lists are created by sorting all collocation patterns by the absolute value of the smoothed log likelihood ratio, computed as described above. The following are highly abbreviated examples:

Decision List for *lead (noun)* *(highly abbreviated)*		
LogL	Evidence	Pron.
11.40	*follow/V* + lead	⇒ lid
11.20	*zinc* in $\pm k$ wrds	⇒ lɛd
11.10	lead *level/N*	⇒ lɛd
10.66	*of* lead *in*	⇒ lɛd
10.59	*the* lead *in*	⇒ lid
10.51	lead *role*	⇒ lid
10.35	*copper* in $\pm k$	⇒ lɛd
10.16	lead *poisoning*	⇒ lɛd
8.55	*big* lead	⇒ lid
8.49	*narrow* lead	⇒ lid
7.76	*take/V* + lead	⇒ lid
5.99	lead , *NOUN*	⇒ lɛd
1.15	lead *in*	⇒ lid

Decision List for *bass* *(highly abbreviated)*		
LogL	Evidence	Pron.
10.98	*fish* in $\pm k$ wrds	⇒ bæs
10.92	*striped* bass	⇒ bæs
9.70	*guitar* in $\pm k$	⇒ beɪs
9.20	bass *player*	⇒ beɪs
9.10	*piano* in $\pm k$	⇒ beɪs
9.01	*tenor* in $\pm k$	⇒ beɪs
8.87	*sea* bass	⇒ bæs
8.49	*play/V* + bass	⇒ beɪs
8.31	*river* in $\pm k$	⇒ bæs
8.28	*violin* in $\pm k$	⇒ beɪs
8.21	*salmon* in $\pm k$	⇒ bæs
7.71	*on* bass	⇒ beɪs
5.32	bass *are*	⇒ bæs

The resulting decision lists are used to classify new examples by identifying the highest line in the list that matches the given context and returning the indicated classification. This process is described in Step 6.

Step 5: Optional Pruning and Interpolation

The decision lists created above may be used *as is* if we assume that the likelihood ratio for the jth entry in the list is roughly the same when computed on the entire training set and when computed on the residual portion of the training set where the first $j-1$ entries have failed to match. In other words, does the probability that *piano* indicates the /beɪs/ pronunciation of *bass* change significantly conditional on not having seen *fish*, *striped*, *guitar*, and *player* in the target context?

In most cases the *global* probabilities (computed from the full training set) are acceptable approximations of these *residual* probabilities. However, in many cases we can achieve improved results by interpolating between the two values. The residual probabilities are more relevant, but since the size of the residual training data shrinks at each level in the list, they are often much more poorly estimated (and in many cases there may be no relevant data left in the residual on which to compute the distribution of pronunciations for a given collocation). In contrast, the global probabilities are better estimated but less relevant. A reasonable compromise is to interpolate between the two where the interpolated estimate is $\beta_i \times global + (1 - \beta_i) \times residual$. When the residual probabilities are based on a large training set and are well estimated, β_i is small (the residual will dominate). In cases where the relevant residual is small or non-existent, β_i will be large and the smoothed probabilities will rely primarily on the better estimated global values. If all $\beta_i = 0$ (exclusive use of the residual), the result is a degenerate (strictly right-branching) decision tree with severe sparse data problems. Alternately, if one assumes that likelihood ratios for a given collocation are functionally equivalent at each line of a decision list, then one could exclusively use the global (all $\beta_i = 1$). This is clearly the easiest and fastest approach, as probability distributions do not need to be recomputed as the list is constructed.

Which approach is best? Using only the global proabilities does surprisingly well, and the results cited here are based on this readily replicable procedure. The reason is grounded in the strong tendency of a word to exhibit only one sense or pronunciation per collocation (discussed in Step 3 and [Yar93]). Most classifications are based on an x vs. 0 distribution, and while the magnitude of the log-likelihood ratios may decrease in the residual, they rarely change sign. There are cases where this does happen and it appears that some interpolation helps, but for *this* problem the relatively small difference in performance does not necessarily justify the greatly increased computational cost.

Two kinds of optional pruning can increase the efficiency of the decision lists. The first handles the problem of "redundancy by subsumption," which occurs when more general patterns higher in the list subsume more specific patterns lower down. The more specific patterns will never be used in Step 6 and may be omitted. Examples of this include lemmas (e.g. *follow/V*) subsuming inflected forms (*follow*, *followed*, *follows*, etc.), and bigrams subsuming trigrams. If a bigram unambiguously signals the pronunciation, probability distributions for dependent trigrams need not even be generated, since they will provide no additional useful information.

The second, pruning in a cross-validation phase, compensates for overmodeling of the training data (which appears to be minimal). Once a decision list is built it is applied to its own training set plus some held-out cross-validation data (*not* the test data). Lines in the list which contribute to more incorrect classifications than correct ones are removed. This also indirectly handles problems that may result from the omission of the interpolation step. If space is at a premium, lines which are never used in the cross-validation step may also be pruned. However, useful information is lost here, particularly for a small cross-validation corpus; these lines may have proved useful during later classification of the test data. Overall, a 3% drop in performance is observed, but an over 90% reduction in space is realized. The optimum pruning strategy is subject to cost-benefit analysis. In the results reported below, all pruning except this final space-saving step was utilized.

Step 6: Using the Decision Lists

Once the decision lists have been created, they may be used in real time to determine the pronunciations of ambiguous words in new contexts.

From a statistical perspective, the evidence at the top of this list will most reliably disambiguate the target word. Given a word in a new context to be assigned a pronunciation, if we may only base the classification on a single line in the decision list, it should be the highest ranking pattern that is present in the target context. This is uncontroversial, and is solidly based in Bayesian decision theory.

The question, however, is what to do with the less-reliable evidence that may also be present in the target context. The common tradition is to combine the available evidence in a weighted sum or product. This is done by Bayesian classifiers, neural nets, IR-based classifiers and N-gram part-of-speech taggers. The system reported here is unusual in that it does no such combination. *Only* the single most reliable piece of evidence matched in the target context is used.

There are several motivations for this approach. The first is that combining all available evidence rarely produces a different classification than just using the single most reliable piece of evidence, and when these differ it is as likely to hurt as to help. A study in [Yar94] based on 20 homographs showed that the two methods agreed in 98% of the test cases. Indeed, in the 2% cases of disagreement, using only the single best piece of evidence worked slightly *better* than combining evidence. Of course this behavior does not hold for all classification tasks, but *does* seem to be characteristic of lexically-based semantic classifications. This may be explained by the previously noted observation that in most cases, and with high probability, words exhibit only one sense in a given collocation[Yar93].

Thus for this type of ambiguity resolution, there is no apparent detriment, and some apparent performance gain, from using only the single most reliable evidence in a classification. There are other advantages as well, including run-time efficiency and ease of parallelization. However, the greatest gain comes from the ability to incorporate non-independent information types in the decision procedure. A given word in context may match several times in the decision list, once each for its part of speech, lemma, inflected form, bigram, trigram, and possible word-classes as well. By only using one of these matches, the gross exaggeration of probability from combining all of these non-independent log-likelihoods is avoided. While these dependencies may be modeled and corrected for in Bayesian formalisms, it is difficult and costly to do so. Using only one log-likelihood ratio without combination frees the algorithm to include a wide spectrum of highly non-independent information without additional algorithmic complexity or performance loss.

4 Decision Lists for Ambiguity Classes

This algorithm may also be directly applied to large classes of ambiguity, such as distinguishing between fractions and dates. Rather than train individual pronunciation discriminators for *5/16* and *5/17*, etc., training contexts are pooled for all individual instances of the class. Since the disambiguating characteristics are quite similar for each class member, enhanced performance due to larger training sets tends to compensate for the loss of specialization.

4.1 Class Models – Creation

Decision lists for ambiguity classes may be created by replacing all members of the class found in the training data (e.g. *5/16* and *5/17*) with a common class label (e.g. *X/Y*). The algorithm described in Section 3 may then be applied to this data.[6]

An abbreviated decision list for the fraction/date class is shown below:

Decision List for Fraction/Date Class		
LogL	Evidence	Pronunciation
8.84	*<NUMBER>* (X/Y)	⇒ FRACTION
7.58	(X/Y) *of*	⇒ FRACTION
6.79	*Monday* in $\pm k$ words	⇒ DATE
6.05	*Mon* in $\pm k$ words	⇒ DATE
5.96	(X/Y) *mile*	⇒ FRACTION
5.68	(X/Y) *inch*	⇒ FRACTION
4.22	*on* (X/Y)	⇒ DATE
3.96	*from* (X/Y) *to*	⇒ DATE

[6] There are advantages to filtering or weighting the training data such that each member of the class has roughly balanced representation. This causes the trained decision list to model the dominant common features of the class, rather than the idiosyncrasies of its most frequent members.

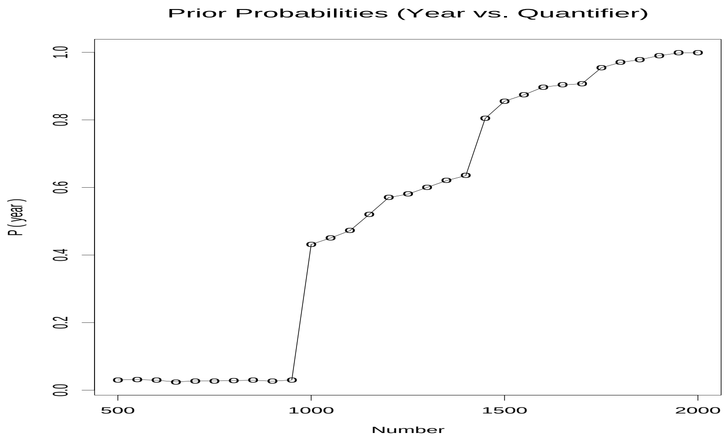
4.2 Class Models – Use

The use of these class decision lists requires one additional step: translating from the raw text (e.g. *5/16*) to its full pronunciation using the paradigm (e.g. FRACTION) selected by the decision list. In conjunction with the AT&T TTS speech synthesizer, the decision lists specify the chosen paradigm by using escape sequences surrounding the ambiguous form as output from the list (e.g. "`\!nfr 5/16 \!nfc`" for FRACTION).

Although the rules for this translation are typically straightforward and standard, a complication arises in the case of dates. American and British conventions differ regarding the order of day and month, pronouncing *3/7* in "*Monday, 3/7 at 5 PM*" as *March 7th* and *July 3rd*, respectively. It would seem reasonable to make this choice conditional on a global BRITISH or AMERICAN parameter, set for the region of use. However, even if one decided to treat ambiguous dates conservatively (e.g. *three slash seven*), there is still considerable merit in pronouncing known fractions properly (e.g. *"3/7 of the"* as *three sevenths* rather than *three slash seven*).

4.3 Class Models – Incorporating Prior Probabilities

Clearly not every member of the class has the same inherent probability, independent of context. We can gain leverage by modeling these differences in prior probability. For example, the class ambiguity YEAR/QUANTIFIER exhibits the following distribution of the prior probability of being a year, for numbers from 500 to 2000. [7]



[7] The spurt at 1000 is due to the possibility of numbers greater than 1000 being written with a comma. This tendency is greatest for literary and news text, inconsistent in informal correspondence, and relatively rare in scientific text. The second spurt at roughly 1492 is due to a strong American bias in the training data's historical references.

These prior probabilities may be used dynamically as follows. Train a decision list for the class assuming an uninformative prior. When applying the list, if the highest matching pattern based on context indicates the same pronunciation as the majority pronunciation based on the appropriate prior, return this result. If it indicates the minority pronunciation, find the highest matching pattern that indicates the majority pronunciation. If the difference in log likelihoods exceeds the log of the prior ratio, use the minority pronunciation.

4.4 Roman Numerals

Roman numerals are an example of where two-tiered class models may be productively used. The majority of Roman numerals (including II, III, VI, VII, VIII, IX, XII, XIII, ...) exhibit the basic distinction between the uses *Chapter VII* and *Henry VII*. These are modeled in the abbreviated decision list below.

Decision List for Roman Numerals (e.g. *VII*)		
LogL	Evidence	Pronunciation
9.63	<NEW-SENT> VII	⇒ SEVEN
9.59	*king* (within $\pm k$ words)	⇒ THE SEVENTH
9.35	*Chapter* VII	⇒ SEVEN
9.21	*Henry* VII	⇒ THE SEVENTH
9.16	*Edward* VII	⇒ THE SEVENTH
8.63	*Title* VII	⇒ SEVEN
7.82	*Volume* VII	⇒ SEVEN
7.65	*pope* (within $\pm k$ words)	⇒ THE SEVENTH
7.03	*Pius* VII	⇒ THE SEVENTH
6.57	*Mark* VII	⇒ SEVEN
6.04	*Gemini* VII	⇒ SEVEN
5.96	*Part* VII	⇒ SEVEN
	∘ ∘ ∘	
1.83	<PROP-NOUN> VII	⇒ THE SEVENTH

However, four Roman numerals exhibit an additional possible pronunciation. They include *IV* as /aɪ vi/ (for intravenous) and *I*, *V* and *X* (letters). For these cases, an initial decision list makes the primary distinction between these additional interpretations and the NUMERIC options, based on such collocations as *IV drug, fluid, dose, injection, oral* and *intramuscular*. If the NUMERIC option is identified, the general Roman numeral list is consulted to determine if the final pronunciation should be as in *Article IV* or *George IV*. This two-tiered list maximizes use of existing class models.

5 Evaluation

The following table provides a summary of the algorithm's performance on the classes of ambiguity studied.

System Performance		
Type of Ambiguity (Examp.)	Prior Prob.	% Correct
Diff. Part of Speech (lives)	62	98
Same Part of Speech (bass)	72	97
Proper Names (Nice, Begin)	63	97
Roman Numerals (III)	75	97
Fractions/Dates (5/16)	59	94
Years/Quantifiers (1750)	67	93
Abbreviations (St., Dr.)	87	98
AVERAGE	69	96

A breakdown of performance on a sample of individual homographs follows:

Word	Pron1	Pron2	Sample Size	Prior Prob.	% Correct
lives	laɪvz	lɪvz	33186	69	98 [8]
wound	waʊnd	wund	4483	55	98
lead (N)	lid	lɛd	12165	66	98
tear (N)	tɛə*	tɪə*	2271	88	97
axes (N)	ˈæksiz	ˈæksɪz	1344	72	96
Jan	dʒæn	jɑn	1327	90	98
routed	ɹutɪd	ɹaʊtɪd	589	60	94
bass	beɪs	bæs	1865	57	99
Nice	naɪs	nis	573	56	94
Begin	bɪˈgɪn	beɪgɪn	1143	75	97
Chi	tʃi	kaɪ	1288	53	98
Colon	koʊˈloʊn	ˈkoʊlən	1984	69	98
St.	seint	strit	624	74	99
in.	ɪntʃ	ɪntʃɪz	222	76	96
III	3	the 3rd	28146	70	98
IV	aɪ vi	NUMERIC	2090	83	99
IV (NUMERIC)	4	the 4th	1744	63	98
VII	7	the 7th	1514	76	98
AVERAGE			96558	69	**97**

[8] As a standard for comparison, the PARTS tagger achieves 88% and 82% accuracy on this test data for *lives* and *wound*, respectively. A primary reason for the difference in performance is the lexicalization issue discussed in Section 2.

Evaluation in each case is based on 5-fold cross-validation using held-out test data for a more accurate estimate of system performance. Unless otherwise specified in the text, these results are based on the simplest and most readily replicable options in the algorithm above, and are hence representative of the performance that can be expected from the most straightforward implementation. Using more sophisticated interpolation techniques yields performance above this baseline. The sources of the test (and training) data are described in Section 3, Step 1.

6 Discussion and Conclusions

The algorithm presented here has several advantages which make it suitable for general lexical disambiguation tasks that require attention to both semantic and syntactic context. The incorporation of word and optionally part-of-speech trigrams allows the modeling of many local syntactic and semantic constraints, while collocational evidence in a wider context allows for topic-based semantic distinctions. A key advantage of this approach is that it allows the use of multiple, highly non-independent evidence types (such as root form, inflected form, part of speech, thesaurus category or application-specific clusters) and does so in a way that avoids the complex modeling of statistical dependencies. This allows the decision lists to find the level of representation that best matches the observed probability distributions. It is a kitchen-sink approach of the best kind – throw in many types of potentially relevant features and watch what floats to the top. While there are certainly other ways to combine such evidence, this approach has many advantages. In particular, precision seems to be at least as good as that achieved with Bayesian methods applied to the same evidence. This is not surprising, given the observation in [LTV93] that widely divergent sense-disambiguation algorithms tend to perform roughly the same given the same evidence. The distinguishing criteria therefore become:

- How readily can new and multiple types of evidence be incorporated into the algorithm?
- Are probability estimates provided with a classification?
- How easy is it to understand the resulting decision procedure and the reasons for any given classification?
- Can the resulting decision procedure be easily edited by hand?
- Is the algorithm simple to implement, and can it be applied quickly to new domains?

The current algorithm rates very highly on all these standards of evaluation, especially relative to some of the impenetrable black boxes produced by many machine learning algorithms. Its output is highly perspicuous: the resulting decision list is organized like a recipe, with the most useful evidence first and in highly readable form. The generated decision procedure is also easy to augment by hand, changing or adding patterns to the list. The algorithm is also extremely flexible—it is quite straightforward to use any new feature for which a probability distribution can be calculated. This is a considerable strength relative to other algorithms which are more constrained in their ability to handle diverse types of evidence. In a comparative study [Yar94b], the decision list algorithm outperformed both an N-Gram tagger and Bayesian classifier primarily because it could effectively integrate a wider range of available evidence types.

Overall, the decision list algorithm demonstrates considerable hybrid vigor, combining the strengths of N-gram taggers, Bayesian classifiers and decision trees in a highly effective, general purpose decision procedure for lexical ambiguity resolution.

Acknowledgments: This research was conducted in affiliation with the Linguistics Research Department of AT&T Bell Laboratories. It was also supported by an NDSEG Graduate Fellowship, ARPA grant N00014-90-J-1863 and ARO grant DAAL 03-89-C0031 PRI. The author would like to thank Jason Eisner and Mitch Marcus for their very helpful comments.

7 References

[BFOS84] L. Brieman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth & Brooks, Monterrey CA, 1984.

[Bri93] E. Brill. A Corpus-Based Approach to Language Learning. Ph.D. Thesis, University of Pennsylvania, 1993.

[BDDM91] P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. Word sense disambiguation using statistical methods. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 264–270, Berkeley, 1991.

[BW94] R. Bruce and 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, Las Cruces, NM, 1994.

[Chu88] K. W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143, 1988.

[DI94] I. Dagan and A. Itai. Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics*, 20:563–596, 1994.

[GCY92] W. Gale, K. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439, 1992.

[GCY94] W. Gale, K. Church, and D. Yarowsky. Discrimination decisions for 100,000-dimensional spaces. In A. Zampoli, N. Calzolari and M. Palmer (eds.), *Current Issues in Computational Linguistics: In Honour of Don Walker*, Kluwer Academic Publishers, pages 429–450, 1994.

[Hea91] M. Hearst. Noun homograph disambiguation using local context in large text corpora. In *Using Corpora*, University of Waterloo, Waterloo, Ontario, 1991.

[Jel85] F. Jelinek. Markov source modeling of text generation. In *Impact of Processing Techniques on Communication*, J. Skwirzinski, ed., Dordrecht, 1985.

[LTV93] C. Leacock, G. Towell and E. Voorhees. Corpus-based statistical sense resolution. In *Proceedings, ARPA Human Language Technology Workshop*, pages 260-265, Princeton, 1993.

[Mer90] B. Merialdo. Tagging text with a probabilistic model In *Proceedings of the IBM Natural Language ITL*, Paris, France, pages 161–172, 1990.

[MW64] F. Mosteller and D. Wallace *Inference and Disputed Authorship: The Federalist* Addison-Wesley, Reading, Massachusetts, 1964.

[Riv87] R. L. Rivest. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

[SHY92] R. Sproat, J. Hirschberg and D. Yarowsky. A corpus-based synthesizer. In *Proceedings, International Conference on Spoken Language Processing*, Banff, 1992.

[Yar92] D. Yarowsky. Word-Sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings, COLING-92*, pages 454–460, Nantes, 1992.

[Yar93] D. Yarowsky. One sense per collocation. In *Proceedings, ARPA Human Language Technology Workshop*, pages 266–271, Princeton, NJ, 1993.

[Yar94] D. Yarowsky. Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Las Cruces, NM, 1994.

[Yar94b] D. Yarowsky. A comparison of corpus-based techniques for restoring accents in Spanish and French text. In *Proceedings, 2nd Annual Workshop on Very Large Corpora*, Kyoto, pages 19–32, 1994.