Proceedings of the Workshop on Automatic Summarization (including DUC 2002),
Philadelphia, July 2002, pp. 1-8. Association for Computational Linguistics.

Using Maximum Entropy for Sentence Extraction

Miles Osborne
osborne@cogsci.ed.ac.uk
Division of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW
United Kingdom.

Abstract

A maximum entropy classifier can be used to extract sentences from documents. Experiments using technical documents show that such a classifier tends to treat features in a categorical manner. This results in performance that is worse than when extracting sentences using a naive Bayes classifier. Addition of an optimised prior to the maximum entropy classifier improves performance over and above that of naive Bayes (even when naive Bayes is also extended with a similar prior). Further experiments show that, should we have at our disposal extremely informative features, then maximum entropy is able to yield excellent results. Naive Bayes, in contrast, cannot exploit these features and so fundamentally limits sentence extraction performance.

1 Introduction

Sentence extraction —the recovery of a given set of sentences from some document— is useful for tasks such as document summarisation (where the extracted sentences can form the basis of a summary) or question-answering (where the extracted sentences can form the basis of an answer). In this paper, we concentrate upon extraction of sentences for inclusion into a summary. From a machine learning perspective, sentence extraction is interesting because typically, the number of sentences to be extracted is a very small fraction of the total number of sentences in the document. Furthermore, those clues which determine whether a sentence should be extracted or not tend to be either extremely specific, or very weak, and furthermore interact together in non-obvious ways. From a linguistic perspective, the task is challenging since success hinges upon the ability to integrate together diverse levels of linguistic description.

Frequently (see section 6 for examples), sentence extraction systems are based around simple algorithms which assume independence between those features used to encode the task. A consequence of this assumption is that such approaches are fundamentally unable to exploit dependencies which presumably exist in the features that would be present in an ideal sentence extraction system. This situation may be acceptable when the features used to model sentence extraction are simple. However, it will rapidly become unacceptable when more sophisticated heuristics, with complicated interactions, are brought to bear upon the problem. For example, Boguraev and Neff (2000a) argue that the quality of summarisation can be increased if lexical cohesion factors (rhetorical devices which help achieve cohesion between related document utterances) are modelled by a sentence extraction system. Clearly such devices (for example, lexical repetition, ellipsis, coreference and so on) all contribute towards the general discourse structure of some text and furthermore are related to each other in non-obvious ways.

Maximum entropy (log-linear) models, on the other hand, do not make unnecessary independence assumptions. Within the maximum entropy framework, we are able to optimally integrate together whatever sources of knowledge we believe potentially to be useful for the task. Should we use features that are beneficial, then the model will be able to exploit this fact. Should we use features that are irrelevant, then again, the model will be able to notice this, and effectively ignore them. Models based on maximum entropy are therefore well suited to the sentence extraction task, and furthermore, yield competitive results on a variety of language tasks (Ratnaparkhi, 1996; Berger et al., 1996; Charniak, 1999; Nigam et al., 1999).

In this paper, we outline a conditional maximum

entropy classification model for sentence extraction. Our model works incrementally, and does not always need to process the entire document before assigning classification.[1] It discriminates between those sentences which should and should not be extracted. This contrasts with ranking approaches which need to process the entire document before extracting sentences. Because we model whether a sentence should be extracted or not in terms of features that are extracted from the sentence (and its context in the document), we do not need to specify the size of the summary. Again, this contrasts with ranking approaches which need to specify a priori the summary size.

Our maximum entropy approach for sentence extraction does not come without problems. Using reasonably standard features, and when extracting sentences from technical papers, we find that precision levels are high, but recall is very low. This arises from the fact that those features which predict whether a sentence should be extracted tend to be very specific and occur infrequently. Features for sentences that should not be extracted tend to be much more abundant, and so more likely to be seen in the future. A simple prior probability is shown to help counter-act this tendency. Using our prior, we find that the maximum entropy approach is able to yield results that are better than a naive Bayes classifier.

Our final set of experiments looks more closely at the differences between maximum entropy and naive Bayes. We show that when we have access to an oracle that is able to tell us when to extract a sentence, then in the situation when that information is encoded in *dependent* features, maximum entropy easily outperforms naive Bayes. Furthermore, we also show that even when that information is encoded in terms of *independent features*, naive Bayes can be incapable of fully utilising this information, and so produces worse results than maximum entropy.[2]

[1] *Incremental* classification means that a document is processed from start-to-finish and decisions are made as soon as sentences are encountered. Some of our features (in particular, those which encode sentence position in a document) do require processing the entire document. Using such features prevents true incremental processing. However, it is trivial to remove such features and so ensure true incrementality.

[2] As a reviewer commented, under certain circumstances, naive Bayes can do well even when there are strong dependencies within features (Domingos and Pazzani, 1997). For example, when the sample size is small, naive Bayes can be competitive with more sophisticated approaches such as maximum entropy. Given this, a fuller comparison of naive Bayes and maximum entropy for sentence extraction requires considering sample size in addition to the choice of features.

The rest of this paper is as follows. Section 2 outlines the general framework for sentence extraction using maximum entropy modelling. Section 3 presents our naive Bayes classifier (which is used as a comparison with maximum entropy). We then show in section 4 how both our maximum entropy and naive Bayes classifiers can be extended with an (optimised) prior. The issue of summary size is touched upon in section 5. Section 6 discusses related work. We then present our main results (section 7). Finally, section 8 discusses our results and considers future work.

2 Maximum Entropy for Sentence Extraction

2.1 Conditional Maximum Entropy

The parametric form for a conditional maximum entropy model is as follows (Nigam et al., 1999):

$$P(c \mid s) = \frac{1}{Z(s)} \exp(\sum_i \lambda_i f_i(c, s)) \quad (1)$$

$$Z(s) = \sum_c \exp(\sum_i \lambda_i f_i(c, s)) \quad (2)$$

Here, c is a label (from the set of labels C) and s is the item we are interested in labelling (from the set of items S). In our domain, C simply consists of two labels: one indicating that a sentence should be in the summary ('keep'), and another label indicating that the sentence should not be in the summary ('reject'). S consists of a training set of sentences, linked to their originating documents. This means that we can recover the position of any given sentence in any given document.

Within maximum entropy models, the training set is viewed in terms of a set of *features*. Each feature expresses some characteristic of the domain. For example, a feature might capture the idea that abstract-worthy sentences contain the words *in this paper*. In equation 1, $f_i(c, s)$ is a feature. In this paper we restrict ourselves to integer-valued functions. An example feature might be as follows:

$$f_i(c, s) = \begin{cases} 1 & \text{if } s \text{ contains the phrase} \\ & \textit{in this paper} \\ & \text{and } c \text{ is the label } \textit{keep} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Features are related to each other through *weights* (as can be seen in equation 1, where some feature f_i has a weight λ_i). Weights are real-valued numbers. When a closed form solution cannot be found,

they are determined by numerical optimisation techniques. In this paper, we use conjugate gradient descent to find the optimal set of weights. Conjugate Gradient descent converges faster than Improved Iterative Scaling (Lafferty et al., 1997), and empirically we find that it is numerically more stable.

2.2 Maximum Entropy Classification

When classifying sentences with maximum entropy, we use the equation:

$$\text{label}(s) = \text{argmax}_{c \in C} P(c \mid s) \quad (4)$$

In practice, we are not interested in the probability of a label given a sentence. Instead we use the unnormalised score:

$$\text{label}(s) = \text{argmax}_{c \in C} \exp(\sum_i \lambda_i f_i(c, s)) \quad (5)$$

Note that this maximum entropy classifier assumes a uniform prior. Section 4 shows how a non-uniform prior is used in place of this uniform prior.

We now present our basic naive Bayes classifier. Afterwards, we extend this classifier with a non-uniform prior.

3 Naive Bayes Classification

As an alternative to maximum entropy, we also investigated a naive Bayes classifier. Unlike maximum entropy, naive Bayes assumes features are conditionally independent of each other. So, comparing the two together will give an indication of the level of statistical dependencies which exist between features in the sentence extraction domain. For our experiments, we used a variant of the multi-variate Bernoulli event model (McCallum and Nigam, 1998). In particular, we did not consider features that are absent in some example. This allows us to avoid summing over all features in the model for each example. Note that our maximum entropy model also did not consider absent features.

Within our naive Bayes approach, the probability of a label given the sentence is as follows:

$$P(c \mid s) = \frac{P(c) \prod_{i=1}^{n} P(g_i \mid c)}{P(s)} \quad (6)$$

As before, s is some sentence, c the label, and g_i is some active feature describing sentence s. Naive Bayes models can be estimated in a closed form by simple counting. For features which have zero counts, we use add-k smoothing (where k is a small number less than one).

Since the probability of the data ($P(s)$) is constant:

$$P(c \mid s) \propto P(c) \prod_{i=1}^{n} P(g_i \mid c) \quad (7)$$

If we assume a uniform prior (in which case $P(c)$ is a constant for all c), this can be further simplified to:

$$P(c \mid s) \propto \prod_{i=1}^{n} P(g_i \mid c) \quad (8)$$

Our basic naive Bayes classifier is as follows:

$$\text{label}(s) = \text{argmax}_{c \in C} \prod_{i=1}^{n} P(g_i \mid c) \quad (9)$$

As with the maximum entropy classifier, we later replace the uniform prior with a non-uniform prior.

4 Maximum a Posteriori Classification

In this section, we show how our classifiers can be extended with a non-uniform prior. We also describe how such a prior can be optimised.

4.1 Adding a non-uniform prior

Now, the two classifiers mentioned previously (equations 9 and 5) are both based on maximum likelihood estimation. However, as we describe later, for sentence extraction, the maximum entropy classifier tends to over-select labels. In particular, it tends to reject too many sentences for inclusion into the summary. So, it it useful to extend the two previous classifiers with a non-uniform prior. For the naive Bayes classifier, we have:

$$\text{label}(s) = \text{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(g_i \mid c) \quad (10)$$

Here, $P(c)$ is our prior. The probability of the data ($P(s)$) is constant and so can be dropped.

For the maximum entropy case, we are not interested in the actual probability:

$$\text{label}(s) = \text{argmax}_{c \in C} F(c) \exp(\sum_i \lambda_i f_i(c, s)) \quad (11)$$

$F(c)$ is a function equivalent to the prior when using the unnormalised classifier. When this prior distribution (or equivalent function) is uniform, classification is as before (namely as outlined in sections 2 and 3), and depends upon the maximum entropy or naive Bayes component. When the prior is non-uniform, the classifier behaviour will change. This prior therefore allows us to affect the performance of our system. In particular, we can change the precision-recall balance.

4.2 Optimising the prior

We treat the problem of selecting a prior as an optimisation task: select some $P(c)$ (or $F(c)$) such that performance, as measured by some objective function of the overall classifier, is maximised. Since the choice of objective function is up to us, we can easily optimise the classifier in any way we decide. For example, we could optimise for recall by using as our objective function an f-measure that weighted recall more highly than precision. In this paper, we optimise the prior using as an objective function the f2 score of the classifier (section 7 details this score). Our prior therefore does not reflect relative frequencies of labels (as found in some corpus).

We now need to optimise our prior. Brent's one dimensional function minimisation method is well suited to this task (Press et al., 1993), since for a random variable taking two values, the probability of one value can be defined in terms of the other value. Section 7 describes the held-out optimisation strategy used in our experiments.

Should we decide to use a more elaborate prior (for example, one which was also sensitive to properties of documents) then we would need to use a multi-dimensional function minimisation method.

Note that we have not simultaneously optimised the likelihood and prior probabilities. This means that we do not necessarily find the optimal maximum a posteriori (MAP) solution. It is possible to integrate into maximum entropy estimation (simple) conjugate priors that do allow MAP solutions to be found (Chen and Rosenfeld, 1999). Although it is an open question whether more complex priors can be directly integrated, future work ought to consider the efficacy of such approaches in the context of summarisation.

5 Summary size

Determining the size of the summary is an important consideration for summarisation. Frequently, this is carried out dynamically, and specified by the user. For example, when there is limited opportunity to display long summaries a user might want a terse summary. Alternatively, when recall is important, a user might prefer a longer summary. Usually, systems rank all sentences in terms of how abstract-worthy that are, and then take the top n most highly ranked sentences. This always requires the size of summary to be specified.

In our classification framework, sentences are processed (largely) independently of each other, and so there is no direct way of controlling the size of the summary. Altering the prior will indirectly influence the summary size. For more direct control over summary size, we can rank sentences using our classifiers (we not only label but can also assign label probabilities) and select the top n most highly ranked sentences.

Within our classification approach, the optimised prior plays a similar role to the user-defined number of sentences that a ranking approach might return.

Experiments (not reported here) showed that ranking sentences using our maximum entropy classifier, and then selecting the top n most highly ranked sentences produced slightly worse results than when selecting sentences in terms of classification.

6 Related Work

The summarisation literature is large. Here we consider only a representative sample.

Kupiec et al. (1995) used Naive Bayes for sentence extraction. They did not consider the role of the prior, nor did they use Naive Bayes for classification. Instead, they used it to rank sentences and selected the top n sentences. The *TEXTRACT* system included a sentence extraction component that is frequency-based (Boguraev and Neff, 2000b). Whilst the system uses a wide variety of linguistic cues when scoring sentences, it does not combine these scores in an optimal manner. Also, it does not consider interactions between the linguistic cues. Goldstein et al. (1999) used a centroid similarity measure to score sentences. They do not appear to have optimised their metric, nor do they deal with statistical dependencies between their features.

7 Experiments

Summarisation evaluation is a hard task, principally because the notion of an objective summary is ill-defined. That aside, in order to compare our various systems, we used an *intrinsic* evaluation approach. Our summaries were evaluated using the standard $f2$ score:

$$r = \frac{j}{m} \quad p = \frac{j}{k} \quad \text{f2} = \frac{2pr}{p+r}$$

where:

r	=	Recall
p	=	Precision
j	=	Number of correct sentences in summary
k	=	Number of sentences in summary
m	=	Number of correct sentences in the document

A sentence being 'correct' means that it was marked as being somehow important (abstract-worthy) by a human and labelled 'keep' by one of

our classifiers. Summaries produced by our systems will therefore attempt to mimic the process of selecting what it means for a sentence to be important in a document.

Naturally this premise —that an annotator can decide a priori whether a sentence is abstract-worthy or not— is open to question. That aside, in other sentence extraction scenarios, it may well be the case that sentences can be reliably annotated.

The $f2$ score treats recall and precision equally. This is a sensible metric to use as we have no a priori reason to believe in some other non-equal ratio of the two components.

Our evaluation results are based on the following approach:

1. Split the set of documents into two disjoint sets ($T1$ and $T2$), with 70 documents in $T1$ and 10 documents in $T2$.

2. Further split $T1$ into two disjoint sets $T3$ and $T4$. $T3$ is used to train a model, and $T4$ is a held-out set. The prior is estimated using Brent's line minimisation method, when training using $T3$ and evaluating on $T4$. $T3$ consisted of 60 documents and $T4$ consisted of 10 documents.

3. Results are then presented using a model trained on $T1$, with the prior just found, and evaluated using $T2$. $T1$ is therefore the training set and $T2$ is the testing set. Results are also presented using a flat prior.

4. The whole process is then repeated after randomising the documents. The final results are then averaged over these n runs. We set n to 40.

7.1 Document set

For data, we used the same documents that Teufel (2001) used in her experiments.[3] In brief, these were 80 conference papers, taken from the Comp-lang preprint archive, and semi-automatically converted from LaTeXto XML. The XML annotated documents were then additionally manually marked-up with tags indicating the status of various sentences. This document set is modest in size. On the other hand, the actual documents are longer than newswire messages typically used for summarisation tasks. Also, the documents show variation in style. For example, some documents are written by non-native speakers, some by students, some by multiple authors and so on. Summarisation is therefore hard.

[3] A superset of the documents is described in (Teufel and Moens, 1997).

Here are some properties of the documents. On average, each document contained 8 sentences that were marked as being abstract-worthy (standard deviation of 3.1). The documents on average each contained in total 174 sentences (standard deviation 50.7). Here, a 'sentence' is either any sequence of words that happened to be in a title, or else any sequence of words in the rest of the document. As can be seen, the summaries are not uniformly long. Also, the documents vary considerably in length. Summary size is therefore not constant.

7.2 Features

We used the following, fairly standard features when describing all sentences in the documents:

- Word pairs. Word pairs are consecutive words as found in a sentence. A word pair feature simply indicates whether a particular word pair is present. All words were reduced: truncated to be at most 10 characters long. Stemming (as for example carried out by the Porter stemmer) produced worse results. We extracted all word pairs found in all sentences, and for any given sentence, found the set of (reduced) word pairs.

- Sentence length. We encoded in three binary features whether a sentence was less than 6 words in length, whether it was greater than 20 words in length, or whether it was in between these two ranges. We also used a feature which encoded whether a *previous* sentence was less than 5 words or longer. This captured the idea that summary sentences tend to follow headings (which are short).

- Sentence position. Summary sentences tend to occur either at the start, or the end of a document. We used three features: whether a given sentence was within the first 8 paragraphs of a document, whether a sentence was in the last 3 paragraphs, or whether the sentence was in a paragraph between these two ranges to encode sentence position. Note that this feature requires the whole document to be processed before classification can take place.

- (Limited) discourse features. Our features described whether a sentence immediately followed typical headings such as *conclusion* or *introduction*, whether a sentence was at the start of a paragraph, or whether a sentence followed some generic heading.

Our features are not exhaustive, and are not designed to maximise performance. Instead, they are designed

to be typical of those found in sentence extraction systems. Note that some of our features exploit the fact that the documents are annotated with structural information (such as headers etc).

Experiments with removing stop words from documents resulted in decreased performance. We conjecture that this is because our word pairs are extremely crude syntax approximations. Removing stop words from sentences and then creating word pairs makes these pairs even worse syntax approximations. However, using stop words increased the number of features in our model, and so again reduced performance. We therefore compromised between these two positions, and mapped all stop words to the same symbol prior to creation of word pair features. We also found it useful to remove word pairs which consisted solely of stop words. Finally, for maximum entropy, we deleted any feature that occurred less than 4 times. Naive Bayes did not benefit from a frequency-based cutoff.

7.3 Classifier comparison

Here we report on our classifiers.

As a baseline model, we simply extracted the first n sentences from a given document. Figure 1 summarises our results as n varies. In this table, as in all subsequent tables, P and R are averaged precision and recall values, whilst $F2$ is the f2 score of these averaged values.

n	F2	P	R	n	F2	P	R
1	0	0	0	26	16	10	36
6	3	3	2	31	18	12	45
11	19	15	26	36	18	11	53
16	20	16	29	41	17	10	58
21	23	16	38	46	16	9	58

Figure 1: Results for the baseline model

Figure 2 shows our results for maximum entropy, both with and without the prior. Prior optimisation was with respect to the f2 score. As in subsequent tables, we show system performance when adding more and more features.

Performance without the prior is heavily skewed towards precision. This is because our features are largely acting categorically: the sheer presence of some feature is sufficient to influence labelling choice. Further evidence for this analysis is supported by inspecting one of the models produced when using the full set of all feature types. We see that of the 85883 feature instances in the model, the vast majority are deeded irrelevant by maximum entropy, and assigned a zero weight. Only 7086 features (roughly 10% in total) had non zero weights.

Features	Flat prior			Optimised prior		
	F2	P	R	F2	P	R
Word pairs	8	5	30	20	40	14
and sent length	25	63	16	36	36	36
and sent position	28	62	18	39	35	45
and discourse	35	63	24	42	43	41

Figure 2: Results for the maximum entropy model

Performance using the optimised prior shows more balanced results, with an increase in F2 score. Clearly optimising the prior has helped counter the categorical behaviour of features in our maximum entropy classifier.

Figure 3 shows the results we obtained when using a naive Bayes classifier. As before, the results show performance with and without the addition of the optimised prior. Naive Bayes outperforms maximum entropy when both classifiers do not use a prior. Performance with and without the prior however, is worse than the performance of our maximum entropy classifier with the prior. Evidently, even our relatively simple features interact with each other, and so approaches such as maximum entropy are required to fully exploit them.

Features	Flat prior			Optimised prior		
	F2	P	R	F2	P	R
Word pairs	26	29	23	29	26	32
and sent length	31	33	28	32	29	35
and sent position	33	34	33	36	31	43
and discourse	38	39	37	39	38	40

Figure 3: Results for the naive Bayes model

7.4 Using informative features

Our previous results showed that maximum entropy could outperform naive Bayes. However, the differences, though present, were not large. Clearly, our feature set was imperfect.[4] It is therefore instructive to see what happens if we had access to an oracle who always told us the true status of some unseen sentence. To make things more interesting, we

[4] Another possible reason for the closeness of the results is the small sample size. There may just not be enough evidence to reliably estimate dependencies within the data.

Features	Naive Bayes			Maxent		
	F2	P	R	F2	P	R
Word pairs	30	34	26	32	93	19
and sent length	35	38	32	99	100	99
and sent position	40	41	39	100	100	100
and discourse	43	44	41	99	100	97

Figure 4: Results for basic naive Bayes and maximum entropy models using dependent informative features

Features	Naive Bayes			Maxent		
	F2	P	R	F2	P	R
Word pairs	84	74	97	25	15	91
and sent length	85	75	97	100	100	100
and sent position	84	73	97	100	100	100
and discourse	84	74	97	100	100	100

Figure 5: Results for basic naive Bayes and maximum entropy models using independent informative features

encoded this information in terms of dependent features. We simulated this oracle by using two features which were active whenever a sentence should not be in the summary; for sentences that should be included in the summary, we let either one of those two features be active, but on a random basis. Our features therefore are only informative when the learner is capable of noting that there are dependencies. We then repeated our previous maximum entropy and naive Bayes experiments. Figure 4 summarise our results.

Unsurprisingly, we see that when features are highly dependent upon each other, maximum entropy easily outperforms naive Bayes.

Even when we have access to features that are independent of each other, naive Bayes can still do worse than maximum entropy. To demonstrate this, we used a feature that was active whenever a sentence should be in the summary. This feature was not active on sentences that should not be in the summary. Figure 5 summarises our results.

As can be seen (figure 5), even when naive Bayes has access to a perfectly reliable informative feature, the fact that the other features are not suitably discounted means that performance is worse than that of maximum entropy. Maximum entropy can discount the other features, and so can take advantage of reliable features.

8 Comments and Future Work

We showed how maximum entropy could be used for sentence extraction, and in particular, that adding a prior could deal with the categorical nature of the features. Maximum entropy, with an optimised prior, did yield marginally better results than naive Bayes (with and without a similarly optimised prior). However, the differences were not that great. Our further experiments with informative features showed that this lack of difference was probably due (at least in part) to the actual features used, and not due to the technique itself.

Our oracle results are an idealisation. A fuller comparison should use more sophisticated features, along with more data. As a result of this, we conjecture that should we use a much more sophisticated feature set, we would expect that the differences between maximum entropy and naive Bayes would become greater.

Our approach treated sentences largely independently of each other. However, abstract-worthy sentences tend to bunch together, particularly at the beginning and end of a document. We intend capturing this idea by making our approach sequence-based: future decisions should also be conditioned on previous choices.

A problem with supervised approaches (such as ours) is that we need annotated material (Marcu, 1999). This is costly to produce. Future work will consider *weakly* supervised approaches (for example *cotraining*) as a way of bootstrapping labelled material from unlabelled documents (Blum and Mitchell, 1998). Note that there is a close connection between multi-document summarisation (where many alternative documents all consider similar issues) and the concept of a *view* in cotraining. We expect that this redundancy could be exploited as a means of providing more annotated training material, and so yield better results.

In summary, maximum entropy can be beneficially used in sentence extraction. However, one needs to guard against categorial features. An optimised prior can provide such help.

Acknowledgement

We would like to thank Rob Malouf for supplying the excellent log-linear estimation code, Simone Teufel for providing the annotated data, Karen Spark Jones for a discussion about summarisation, Steve Clark for spotting textual bugs and the anonymous reviewers for useful comments.

References

Adam Berger, Stephen Della Pietra, and Vincent Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 21–22.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann Publishers.

Branimir K. Boguraev and Mary S. Neff. 2000a. The effects of analysing cohesion on document summarisation. In *Proceedings of the* 18^{th} *International Conference on Computational Linguistics*, volume 1, pages 76–82, Saarbrucken.

Branmir K. Boguraev and Mary S. Neff. 2000b. Discourse Segmentation in Aid of Document Summarization. In *Proceedings of the* 33^{rd} *Hawaii International Conference on Systems Science*.

Eugene Charniak. 1999. A maximum-entropy-inspired parser. Technical Report CS99-12, Department of Computer Science, Brown University.

Stanley F. Chen and Ronald Rosenfeld. 1999. A Gaussian prior for smoothing maximum entropy models. Technical Report CMU-CS-99-108, Carnegie Mellon University.

Pedro Domingos and Michael J. Pazzani. 1997. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130.

Jade Goldstein, Mark Kantrowitz, Vibhu O. Mittal, and Jaime G. Carbonell. 1999. Summarizing text documents: Sentence selection and evaluation metrics. In *Research and Development in Information Retrieval*, pages 121–128.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A Trainable Document Summarizer. In *Proceedings of the* 18^{th} *ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 68–73.

J. Lafferty, S. Della Pietra, and V. Della Pietra. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, April.

Daniel Marcu. 1999. The automatic construction of large-scale corpora for summarization research. In *Research and Development in Information Retrieval*, pages 137–144.

A. McCallum and K. Nigam. 1998. A comparison of event models for naive bayes text classificatio. In *AAAI-98 Workshop on Learning for Text Categorization.*

Kamal Nigam, John Lafferty, , and Andrew McCallum. 1999. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering,*.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1993. *Numerical Recipes in C: the Art of Scientific Computing*. Cambridge University Press, second edition.

Adwait Ratnaparkhi. 1996. A Maximum Entropy Part-Of-Speech Tagger. In *Proceedings of Empirical Methods in Natural Language*, University of Pennsylvania, May. Tagger: `ftp://ftp.cis.upenn.edu/pub/adwait/jmx`.

S. Teufel and M. Moens. 1997. Sentence extraction as a classification task. In *ACL/EACL-97 Workshop on Intelligent and Scalable Text Summarization*, Madrid, Spain.

Simone Teufel. 2001. Task-Based Evaluation of Summary Quality: Describing Relationships Between Scientific Papers. In *NAACL Workshop on Automatic Summarization*, Pittsburgh, Pennsylvania, USA, June. Carnegie Mellon University.