A Hidden Markov Model for the TREC Novelty Task

John M. Conroy*

October 27, 2004

1 Introduction

The algorithms for choosing relevant sentences were tuned versions of those presented in the past DUC evaluations and TREC 2003 (see [4, 5, 10] [11] for more details). The enhancements to the previous system are detailed in Section 3.

Two methods were explored to find a subset of the relevant sentences that had good coverage but low redundancy. In the multi-document summarization system, the QR algorithm is used on term-sentence matrices. For this work, the method of maximum marginal relevance was also employed. The evaluation of these methods is discussed in Section 5.

2 Preprocessing

2.1 Tokenization

The tokenization was quite simple. First the text was converted to lower case. All contiguous strings of characters taken from the set {a,b,...,z} were terms except for those matched on a short list of stop words.

2.2 Parsing Files using DTDs

Using the SGML document type definition (DTD) for a document allowed us to determine the set of all possible SGML tags that exist in documents of that type. Using these tag sets, we distinguished which sentences 1) were candidates for relevant sentences, 2) were not candidates for relevant sentences but which contained key terms or phrases that would aid in identifying relevant sentences, and 3) contained no useful information for the task of extracting relevant sentences. We created a new attribute, *stype*, for the SGML tag denoting a sentence boundary, <s>, in order to denote each of these three types of sentences. The possible values for this new attribute are 1, 0, and -1, respectively. Table 1 presents the values of *stype* used for sentences embedded into the SGML tags encountered in the several types of documents used in the evaluation.

*IDA/Center for Computing Sciences, conroy@super.org

File	DTD	Filename	SGML Tag	*stype*
APW*	ACQUAINT	acquaint.dtd	<TEXT>	1
NYT*			<HEADLINE>	0
XIE*				0
FBIS*	FBIS	fbis.dtd	<TEXT>	1
			<TI>	0
			<H1>, ..., <H8>	0
FR*	Federal Register	fr.dtd	<TEXT>	1
			<SUMMARY>	1
			<SUPPLEM>	1
			<FOOTNOTE>	1
			<DOCTITLE>	0
FT*	Financial Times	ft.dtd	<TEXT>	1
			<HEADLINE>	0
LA*	LA Times	latimes.dtd	<TEXT>	1
			<HEADLINE>	0
			<SUBJECT>	0
			<GRAPHIC>	0

Table 1: Mapping SGML tags to *stype* values. All tags not shown but allowed by each DTD are assigned $stype = -1$.

Choosing to embed information into the document itself instead of creating a processing module in our algorithm allowed us flexibility in using the information throughout the various stages of our system. Furthermore, it will allow us to expand the types of sentence classification without changing the code.

3 Finding Relevant Sentences

An HMM, in contrast to a naive Bayesian approach ([1], [7]), has fewer assumptions of independence. In particular, it does not assume that the probability that sentence i is relevant is independent of whether sentence $i-1$ is relevant. In the HMM developed for this evaluation, we used a joint distribution for the features set which varied based upon the position in the document.

All of the features used by the HMM were based upon the terms (as defined in Section 2.1) found in a sentence. The features for the HMM were as follows:

- the entropy of the terms in the sentence–value is $o_0(i) = -\sum_j p_j log(p_j)$ where j ranges over all the terms in sentence i and p_j is the probability of term j for the given document.
- number of signature terms, n_{sig}, in a sentence—value is $o_1(i) = \log(n_{sig} + 1)$.
- number of tokens, n_{tok}, in a sentence—value is $o_2(i) = \log(n_{tok} + 1)$.

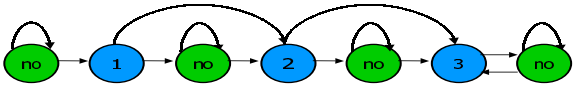

Figure 1: Markov Chain to Extract 2 Lead Sentences and Supporting Sentences

- position of the sentence in the document—built into the state-structure of the HMM.

The signature terms are the terms that are more likely to occur in the document (or document set) than in the corpus at large. To identify these terms, we used the log-likelihood statistic suggested by Dunning [6] and used first in summarization by Lin and Hovy [8]. The statistic is equivalent to a mutual information statistic and is based on a 2-by-2 contingency table of counts for each term.

The features were normalized component-wise to have mean zero and variance one. In addition, the features for sentences with *stype* 0 and -1 were coerced to be -1, which forced these sentences to have an extremely low probability of being selected as relevant sentences.

An HMM handles the positional dependence, dependence of features, and Markovity. (For more details about HMMs, see [2] and [9].) The model we proposed has $2s + 1$ states, with s relevance states and $s + 1$ non-relevance states. A picture of the Markov chain is given in Figure 1. Note that we allowed hesitation only in non-relevance states and skipping of states only from relevance states. This chain was designed to model the extraction of up to $s - 1$ lead relevant sentences and an arbitrary number of supporting relevant sentences. Using training data, we obtained a maximum-likelihood estimate for each transition probability and this formed an estimate, M, for the transition matrix for our Markov chain, where element (i, j) of M is the estimated probability of transitioning from state i to state j.

Associated with each state i is an output function, $b_i(O) = Pr(O|state\ i)$, where O is an observed vector of features. We made the simplifying assumption that the features were multivariate normal. The output function for each state was estimated by using the training data to compute the maximum-likelihood estimate of its mean and covariance matrix. We estimated $2s + 1$ means, but assumed that all of the output functions shared a common covariance matrix.

Training for the HMM was straightforward given marked data. Since the states of the HMM were known in the training data, creating the model simply amounted to computing the maximum likelihood statistics given the counts.

In particular, the training data helped determine the number of states for the HMM. The upshot was that a state space consisting of nine states (five relevance states and four non-relevance states) was optimal given TREC 2003 data. With this model we computed $\gamma_j(i)$, the probability that sentence j corresponded to state i. We computed the probability that a sentence was a relevant sentence by summing $\gamma_j(i)$ over all even values of i, values corresponding to relevance states. This

posterior probability, which we define as g_j, was used to select the most likely relevant sentences. We refer the reader to [3] for details.

This posterior probability was used to select which sentences were likely to be relevant. The selection algorithm attempted to choose the number of sentences so that the expected F_1 score was maximized. The approximate F_1 score was computed based on the expected precision, $E(P)$, and expected recall, $E(R)$, as follows:

$$\widehat{F_1} = \frac{2E(P)E(R)}{E(P) + E(R)}$$

where

$$E(P) = \frac{\sum_{t \epsilon S} g_t}{|S|}$$

where $|S|$ is the cardinality of the set S of sentences selected, and

$$E(R) = \frac{\sum_{t \epsilon S} g_t}{\sum_t g_t}\,.$$

The set S was chosen by selectively choosing the sentences in decreasing order of their probability of being a relevant sentence. The score $\widehat{F_1}$ was then computed and the set S increased as long as $\widehat{F_1}$ increased.

4 Finding New Sentences

To choose a subset of the candidate relevant sentences to produce new sentences we experimented with two algorithms: a pivoted QR decomposition and MMR. These methods work on the term–sentence matrix, A, where A_{ij} is 1 if term i occurs in relevant sentence j. Before applying the sentence selection algorithms, the columns of A were normalized; the Euclidean length of a column was set equal to the probability that the corresponding sentence was indeed relevant. For Tasks 2 and 4 these probabilities were 1 since the relevant sentences were given, while for Task 3, the probability was equal to the score produced by the HMM for that sentence, g_j.

A QR decomposition with partial pivoting, can be applied to the weighted term-sentence matrix $A_w = A$. The QR decomposition was used to determine whether a sentence should be considered new or redundant. In the QR factorization a sentence was considered redundant if the vector corresponding to it had a low value for $\widehat{F_1}$. This is in contrast to the method we used in 2003 where a small weight, say less than τ, a predefined threshold. [I believe this change in part accounted for the lower performance on Task 2.] Likewise this method was used by the MMR to determine when a redundant sentence was found.

The standard implementation of the pivoted QR decomposition is a "Gram-Schmidt" process and was used to select new sentences as follows.

Algorithm 4.1 (Pivoted QR Decomposition) *Suppose A_w has m rows and n columns: i.e., the document has m unique terms and n sentences. The following iteration constructs a matrix Q with columns q_i, a matrix R with nonzero elements r_{ji}, and an ordering for the columns in an array Index.*

For $i = 1, 2, \ldots, \min(m, n)$,

Among the remaining columns of A_w, choose the column with maximal norm. Denote this column by a_ℓ, where ℓ is its index in the original matrix.

Set $Index_i = \ell$.

Set $q_i = a_\ell / ||a_\ell||$.

Update the other columns of A_w to make them orthogonal to the chosen column: for each unchosen column a_j, set $r_{ji} = a_j^T q_i$ and set $a_j = a_j - r_{ji} q_i$.

The set of "new" sentences of size k contains sentences $Index_1, \ldots, Index_k$.

5 Results

For Task 1 the HMM used for TREC was trained using the marked relevant and new sentences in the Novelty data from TREC 2003. Specifically, for Task 1 two models were built. The first model used only one output for the HMM, (o_1, the number log of the number of signature terms+1). These runs have the prefix *ccs1f*. A second model used all three features and these entries have the prefix *ccs3f*.

Note that for Task 1 the suffixes "mmrt1" and "qrt1" denote the results using an MMR and those using just a pivoted QR, respectively. The suffix *0t1* simply passes all tagged relevant sentences as novel.

The median $F1$ scores are given in the tables for the entries and as well as the median rank of the entry based on the other Novelty submission, which there were 60 for Task 1. In addition, we computed the median $F1$ score for the document sets that contained only relevant documents, i.e., those with exactly 25 documents (SZ=25) versus those with some non-relevant documents in the set (SZ>25). Finally, we see the $F1$ scores for document sets which were Event topics versus Opinion topics.

Run	Median Precision	Median Recall	Median F1	Median Rank	SZ=25	SZ>25	Event F1	Opinion F1
ccs1f0t1	20	82	32	41	48	30	44	24
ccs1ftop0t1	20	75	31	48	48	29	42	24
ccs3fmmrt1	22	86	34	36	49	28	43	25
ccs3fqrt1	21	94	34	34	49	31	47	25
ccs3ftop0t1	22	86	34	36	49	28	43	25

Table 2: Performance of CCSUM on Task 1: Relevant Sentences; 60 Total Entries

In Task 2 we were given the relevant sentences and had to determine the new sentences. We submitted 5 approaches: a pivoted QR (*ccsumqrt2*), and four variations of MMR entries, which varied the weighting parameter between the HMM score and the redundancy penalty. For the task of selecting the new sentences given a list of putative relevant sentences (Task 2), our entries did much poorer this year than last year, where the pivoted QR did comparably with the best entries submitted. (We are still investigating; however, as mentioned above we believe this may have been due to the new stopping selection scheme.)

Run	Median Precision	Median Recall	Median F1	Median Rank	SZ=25	SZ>25	Event F1	Opinion F1
ccs1f0t1	7	82	13	44	11	14	18	11
ccs1ftop01t	7	74	13	42	11	13	18	10
ccs3fmmrt1	7	84	13	40	13	13	18	11
ccs3fqrt1	8	88	14	39	13	14	20	11
ccs3ftop0t1	7	86	13	41	12	13	18	11

Table 3: Performance of CCSUM on Task 1: New Sentences; 60 Total Entries

Run	Median Precision	Median Recall	Median F1	Median Rank	SZ=25	SZ>25	Event F1	Opinion F1
ccsmmr2t2	48	55	48	50	44	49	47	51
ccsmmr3t2	46	84	60	33	56	62	58	62
ccsmmr4t2	44	96	60	24	58	64	59	64
ccsmmr5t2	42	100	59	33	55	63	56	62
ccsqrt2	42	100	60	28	57	64	58	64

Table 4: Performance of CCSUM on Task 2: New Sentences; 55 Total Entries

In Task 3 we were given the relevant and new sentences for the first 5 documents of each of the document sets. We realized after submitting our results two of our entries (not shown in the table here) were bogus. They were processed as if they were Task 2 entries and are invalid entries. As such these entries scored 0 for relevant sentences and the best for the novel sentences!

Run	Median Precision	Median Recall	Median F1	Median Rank	SZ=25	SZ>25	Event F1	Opinion F1
ccs3fmmr95t3	20	72	32	32	44	29	39	25
ccs3fmmrt3	20	72	32	32	44	29	39	25

Table 5: Performance of CCSUM on Task 3:Relevant Sentences; 38 Total Entries

References

[1] C. Aone, M. Okurowski, J. Gorlinsky, and B. Larsen. "A Scalable Summarization System Using Robust NLP". In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 66–73, 1997.

[2] L. Baum, T. Petrie, G. Soules, and N. Weiss. "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains". *Ann. Math. Stat.*, 41:164–171, 1970.

[3] J. Conroy and D. O'Leary. "Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition". Technical report, University of Maryland, College Park, Maryland, March, 2001.

Run	Median Precision	Median Recall	Median F1	Median Rank	SZ=25	SZ>25	Event F1	Opinion F1
ccs3fmmr95t3	8	64	15	26	12	15	20	11
ccs3fmmrt3.new	9	60	15	24	12	16	21	12

Table 6: Performance of CCSUM on Task 3:New Sentences; 38 Total Entries

[4] J. Conroy, J. Schlesinger, D. O'Leary, and M. Okurowski. "Using HMM and Logistic Regression to Generate Extract Summaries for DUC". In *DUC 01 Conference Proceedings*, 2001.

[5] D. Dunlavy, J. Conroy, J. Schlesinger, S. Goodman, M. Okurowski, D. O'Leary, and H. van Halteren. "Performance of a Three-Stage System for Multi-Document Summarization". In *DUC 03 Conference Proceedings*, 2003.

[6] T. Dunning. "Accurate Methods for Statistics of Surprise and Coincidence". *Computational Linguistics*, 19:61–74, 1993.

[7] J. Kupiec, J. Pedersen, and F. Chen. "A Trainable Document Summarizer". In *Proceedings of the 18th Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 68–73, 1995.

[8] C. Lin and E. Hovy. "The Automatic Acquisition of Topic Signatures for Text Summarization". In *Proceedings of the 18th International Conference on Computational Liguistics (COLLING 2000)*, 2000.

[9] L. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications". *Proceedings of the IEEE*, 77:257–285, 1989.

[10] J. Schlesinger, M. Okurowski, J. Conroy, D. O'Leary, A.Taylor, J. Hobbs, and H. Wilson. "Understanding Machine Performance in the Context of Human Performance for Multi-document Summarization". In *DUC 02 Conference Proceedings*, 2002.

[11] I. Soboro and D. Harman. "Overview of the TREC 2003 Novelty Track". In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, to appear.