A Phrase-Based HMM Approach to Document/Abstract Alignment

Hal Daumé III and **Daniel Marcu**
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
{hdaume,marcu}@isi.edu

Abstract

We describe a model for creating word-to-word and phrase-to-phrase alignments between documents and their human written abstracts. Such alignments are critical for the development of statistical summarization systems that can be trained on large corpora of document/abstract pairs. Our model, which is based on a novel Phrase-Based HMM, outperforms both the Cut & Paste alignment model (Jing, 2002) and models developed in the context of machine translation (Brown et al., 1993).

1 Introduction

There are a wealth of document/abstract pairs that statistical summarization systems could leverage to learn how to create novel abstracts. Detailed studies of such pairs (Jing, 2002) show that human abstractors perform a range of very sophisticated operations when summarizing texts, which include reordering, fusion, and paraphrasing. Unfortunately, existing document/abstract alignment models are not powerful enough to capture these operations. To get around directly tackling this problem, researchers in text summarization have employed one of several techniques.

Some researchers (Banko et al., 2000) have developed simple statistical models for aligning documents and headlines. These models, which implement IBM Model 1 (Brown et al., 1993), treat documents and headlines as simple bags of words and learn probabilistic word-based mappings between the words in the documents and the words in the headlines. As our results show, these models are too weak for capturing the operations that are employed by humans in summarizing texts beyond the headline level.

Other researchers have developed models that make unreasonable assumptions about the data, which lead to the utilization of a very small percent of available data. For instance, the document and sentence compression models of Daumé III, Knight, and Marcu (Knight and Marcu, 2002; Daumé III and Marcu, 2002a) assume that sentences/documents can be summarized only through deletion of contiguous text segments. Knight and Marcu found that from a corpus of $39,060$ abstract sentences, only 1067 sentence extracts existed: a recall of only 2.7%.

An alternate techinque employed in a large variety of systems is to treat the summarization problem as a sentence extraction problem. Such systems can be trained either on human constructed extracts or extracts generated automatically from document/abstract pairs (see (Marcu, 1999; Jing and McKeown, 1999) for two such approaches).

None of these techniques is adequate. Even for a relatively simple sentence from an abstract, we can see that none of the assumptions listed above holds. In Figure 1, we observe several phenomena:

- Alignments can occur at the granularity of words and at the granularity of phrases.
- The ordering of phrases in an abstract can be different from the ordering in the document.
- Some abstract words do not have direct correspondents in the document, and some document words are never used.

It is thus desirable to be able to automatically construct alignments between documents and their abstracts, so that the correspondences between the pairs are obvious. One might be initially tempted to use readily-available machine translation systems like GIZA++ (Och and Ney, 2003) to perform such

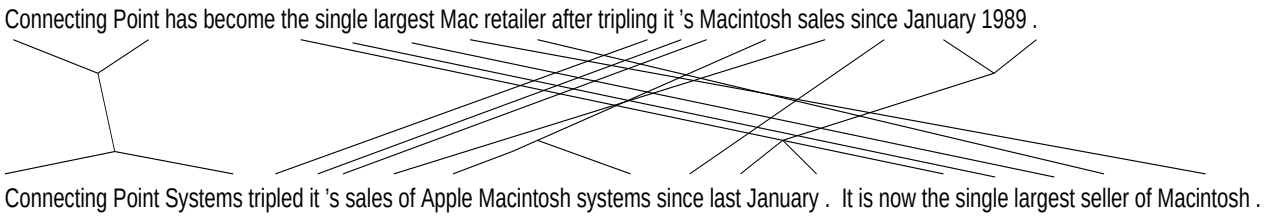

Figure 1: Example abstract/text alignment.

alignments. However, as we will show, the alignments produced by such a system are inadequate for this task.

The solution that we propose to this problem is an alignment model based on a novel mathematical structure we call the Phrase-Based HMM.

2 Designing a Model

As observed in Figure 1, our model needs to be able to account for phrase-to-phrase alignments. It also needs to be able to align abstract phrases with arbitrary parts of the document, and not require a monotonic, left-to-right alignment.[1]

2.1 The Generative Story

The model we propose calculates the probability of an alignment/abstract pair in a generative fashion, generating the summary $S = \langle s_1 \dots s_m \rangle$ from the document $D = \langle d_1 \dots d_n \rangle$.

In a document/abstract corpus that we have aligned by hand (see Section 3), we have observed that 16% of abstract words are left unaligned. Our model assumes that these "null-generated" words and phrases are produced by a unique document word $\varnothing$, called the "null word." The parameters of our model are stored in two tables: a *rewrite/paraphrase table* and a *jump table*. The rewrite table stores probabilities of producing summary words/phrases from document words/phrases and from the null word (namely, probabilities of the form *rewrite* $\left(\bar{s} \,\middle|\, \bar{d}\right)$ and *rewrite* $(\bar{s} \,|\, \varnothing)$); the jump table stores the probabilities of moving within a document from one position to another, and from and to $\varnothing$.

The generation of a summary from a document is assumed to proceed as follows:

1. Choose a starting index i and jump to position d_i in the document with probability *jump* (i). (If the first summary phrase is null-generated, jump to the null-word with probability *jump* $(\varnothing)$.)

2. Choose a document phrase of length $k \geq 0$ and a summary phrase of length $l \geq 1$. Generate summary words s_1^l from document words d_i^{i+k} with probability *rewrite* $\left(s_1^l \,\middle|\, d_i^{i+k}\right)$.[2]

3. Choose a new document index i' and jump to position $d_{i'}$ with probability *jump* $(i' - (i + k))$ (or, if the new document position is the empty state, then *jump* $(\varnothing)$).

4. Choose k' and l' as in step 2, and generate the summary words $s_{1+l}^{1+l+l'}$ from the document words $d_{i'}^{i'+k'}$ with probability *rewrite* $\left(s_{1+l}^{1+l+l'} \,\middle|\, d_{i'}^{i'+k'}\right)$.

5. Repeat from step 3 until the entire summary has been generated.

6. Jump to position d_{n+1} in the document with probability *jump* $(n + 1 - (i' + k'))$.

Note that such a formulation allows the same document word/phrase to generate many summary words: unlike machine translation, where such behavior is typically avoided, in summarization, we observe that such phenomena do occur. However, if one were to build a decoder based on this model, one would need to account for this issue to avoid degenerate summaries from being produced.

The formal mathematical model behind the alignments is as follows: An alignment $\aleph$ defines both a segmentation of the summary S and a mapping from the segments of S to the segments of the document D. We write s_i to refer to the ith segment of S, and M to refer to the total number of segments

[1] In the remainder of the paper, we will use the words "summary" and "abstract" interchangeably. This is because we wish to use the letter s to refer to summaries. We could use the letter a as an abbreviation for "abstract"; however, in the definition of the Phrase-Based HMM, we reuse common notation which ascribes a different interpretation to a.

[2] We write x_a^b for the subsequence $\langle x_a \dots x_b \rangle$.

in S. We write $d_{\aleph(i)}$ to refer to the words in the document which correspond to segment s_i. Then, the probability of a summary/alignment pair given a document ($\Pr(S, \aleph \,|\, D)$), becomes:

$$\prod_{i=1}^{M+1} \left(\textit{jump}\left(\aleph(i) \,|\, \aleph(i-1)\right) \textit{rewrite}\left(s_i \,|\, d_{\aleph(i)}\right)\right)$$

Here, we implicitly define s_{m+1} to be the end-of-document token $\langle\omega\rangle$ and $d_{\aleph(m+1)}$ to generate this with probability 1. We also define the initial position in the document, $\aleph(0)$ to be 0, and assume a uniform prior on segmentations.

2.2 The Mathematical Model

Having decided to use this model, we must now find a way to efficiently train it. The model is very much like a Hidden Markov Model in which the summary is the observed sequence. However, using a standard HMM would not allow us to account for phrases in the summary. We therefore extend a standard HMM to allow multiple observations to be emitted on one transition. We call this model a Phrase-Based HMM (PBHMM).

For this model, we have developed equivalents of the forward and backward algorithms, Viterbi search and forward-backward parameter re-estimation. Our notation is shown in Table 1.

Here, S is the state space, and the observation sequences come from the alphabet K. π_j is the probability of beginning in state j. The transition probability $a_{i,j}$ is the probability of transitioning from state i to state j. $b_{i,j,\bar{k}}$ is the probability of emitting (the non-empty) observation sequence $\bar{k}$ while transitioning from state i to state j. Finally, x_t denotes the state after emitting t symbols.

The full derivation of the model is too lengthy to include; the interested reader is directed to (Daumé III and Marcu, 2002b) for the derivations and proofs of the formulae. To assist the reader in understanding the mathematics, we follow the same notation as (Manning and Schutze, 2000). The formulae for the calculations are summarized in Table 2.

2.2.1 Forward algorithm

The forward algorithm calculates the probability of an observation sequence. We define $\alpha_j(t)$ as the probability of being in state j after emitting the first $t-1$ symbols (in whatever grouping we want).

2.2.2 Backward algorithm

Just as we can compute the probability of an observation sequence by moving forward, so can we calculate it by going backward. We define $\beta_i(t)$ as the probability of emitting the sequence o_t^T given that we are starting out in state i.

2.2.3 Best path

We define a path as a sequence $P = \langle p_1 \ldots p_L \rangle$ such that p_i is a tuple $\langle t, x \rangle$ where t corresponds to the last of the (possibly multiple) observations made, and x refers to the state we were coming from when we output this observation (phrase). Thus, we want to find:

$$\operatorname*{argmax}_P \Pr\left(P \,|\, o_1^T, \mu\right) = \operatorname*{argmax}_P \Pr\left(P, o_1^T \,|\, \mu\right)$$

To do this, as in a traditional HMM, we estimate the ζ table. When we calculate $\zeta_j(t)$, we essentially need to choose an appropriate i and t', which we store in another table, so we can calculate the actual path at the end.

2.2.4 Parameter re-estimation

We want to find the model μ which best explains observations. There is no known analytic solution for standard HMMs, so we are fairly safe in assuming that we will not find an analytic solution for this more complex problem. Thus, we also revert to an iterative hill-climbing solution analogous to Baum-Welch re-estimation (i.e., the Forward Backward algorithm). The equations for the re-estimated values $\hat{a}$ and $\hat{b}$ are shown in Table 2.

2.2.5 Dirichlet Priors

Using simple maximum likelihood estimation is inadequate for this model: the maximum likelihood solution is simply to make phrases as long as possible; unfortunately, doing so will first cut down on the number of probabilities that need to be multiplied and second make nearly all observed summary phrase/document phrase alignments unique, thus resulting in rewrite probabilities of 1 after normalization. In order to account for this, instead of finding the maximum likelihood solution, we instead seek the maximum a posteriori solution.

The distributions we deal with in HMMs, and, in particular, PBHMMs, are all multinomial. The Dirichlet distribution is in the conjugate family to the multinomial distribution[3]. This makes Dirichlet priors very appealing to work with, so long as

[3]This effectively means that the product of a Dirichlet and multinomial yields a multinomial.

S	set of states
K	output alphabet
$\Pi = \{\pi_j : j \in S\}$	initial state probabilities
$A = \{a_{i,j} : i, j \in S\}$	transition probabilities
$B = \{b_{i,j,\bar{k}} : i, j \in S, \bar{k} \in K^+\}$	emission probabilities

Table 1: Notation used for the PBHMM

$$
\begin{aligned}
\alpha_j(t) &= \Pr\left(o_1^{t-1}, x_{t-1} = j \,\middle|\, \mu\right) = \sum_{t'=0}^{t-1} \sum_{i \in S} \left(\alpha_i(t'+1) \cdot a_{i,j} \cdot b_{i,j,o_{t'+1}^t}\right) \\
\beta_i(t) &= \Pr\left(o_t^T \,\middle|\, \mu, x_{t-1} = i\right) = \sum_{t'=t}^{T} \sum_{j \in S} \left(a_{i,j} \cdot b_{i,j,o_t^{t'}} \cdot \beta_j(t'+1)\right) \\
\zeta_j(t) &= \max_{l, p_1^{l-1}} \Pr\left(p_1^{l-1}, o_1^{t-1}, p_l.t = t-1, p_l.x = j \,\middle|\, \mu\right) = \zeta_i(t') a_{i,j} b_{i,j,o_{t'}^{t-1}} \\
\tau_{i,j}(t',t) &= E\left[\#\text{ of transitions } i \rightsquigarrow j \text{ emitting } o_{t'}^t\right] = \frac{\alpha_i(t') a_{i,j} b_{i,j,o_{t'}^t} \beta_j(t+1)}{\Pr\left(o_1^T \,\middle|\, \mu\right)} \\
\hat{a}_{i,j} &= \frac{E\left[\#\text{ of transitions } i \rightsquigarrow j\right]}{E\left[\#\text{ of transitions } i \rightsquigarrow ?\right]} = \frac{\sum_{t'=1}^{T} \sum_{t=t'}^{T} \tau_{i,j}(t',t)}{\sum_{t'=1}^{T} \sum_{t=t'}^{T} \sum_{j' \in S} \tau_{i,j'}(t',t)} \\
\hat{b}_{i,j,\bar{k}} &= \frac{E\left[\#\text{ of transitions } i \rightsquigarrow j \text{ with } \bar{k} \text{ observed}\right]}{E\left[\#\text{ of transitions } i \rightsquigarrow j\right]} = \frac{\sum_{t=1}^{T+1-|\bar{k}|} \delta(\bar{k}, o_t^{t+|\bar{k}|-1}) \tau_{i,j}(t, t+|\bar{k}|-1)}{\sum_{t'=1}^{T} \sum_{t=t'}^{T} \tau_{i,j}(t',t)}
\end{aligned}
$$

Table 2: Summary of equations for a PBHMM

we can adequately express our prior beliefs in their form. (See (Gauvain and Lee, 1994) for the application to standard HMMs.)

Applying a Dirichlet prior effectively allows us to add "fake counts" during parameter re-estimation, according to the prior. The prior we choose has a form such that fake counts are added as follows: word-to-word rewrites get an additional count of 2; identity rewrites get an additional count of 4; stem-identity rewrites get an additional count of 3.

2.3 Constructing the PBHMM

Given our generative story, we construct a PBHMM to calculate these probabilities efficiently. The structure of the PBHMM for a given document is conceptually simple. We provide values for each of the following: the set of possible states S; the output alphabet K; the initial state probabilities Π; the transition probabilities A; and the emission probabilities B.

2.3.1 State Space

The state set is large, but structured. There is a unique initial state p, a unique final state q, and a state for each possible document phrase. That is, for all $1 \leq i \leq i' \leq n$, there is a state that corresponds to the document phrase beginning at position i and ending at position i', $d_i^{i'}$, which we will refer to as $r_{i,i'}$. There is also a null state for each document position $r_{\varnothing,i}$, so that when jumping *out* of a null state, we can remember what our previous position in the document was. Thus, $S = \{p, q\} \cup \{r_{i,i'} : 1 \leq i \leq i' \leq n\} \cup \{r_{\varnothing,i} : 1 \leq i \leq n\}$. Figure 2 shows the schematic drawing of the PBHMM constructed for the document "a b". K, the output alphabet, consists of each word found in S, plus the token ω.

2.3.2 Initial State Probabilities

For initial state probabilities: since p is our initial state, we say that $\pi_p = 1$ and that $\pi_r = 0$ for all $r \neq p$.

2.3.3 Transition Probabilities

The transition probabilities A are governed by the jump table. Each possible jump type and it's associated probability is shown in Table 3. By these calculations, regardless of document phrase lengths, transitioning forward between two consecutive segments will result in *jump* (1). When transitioning

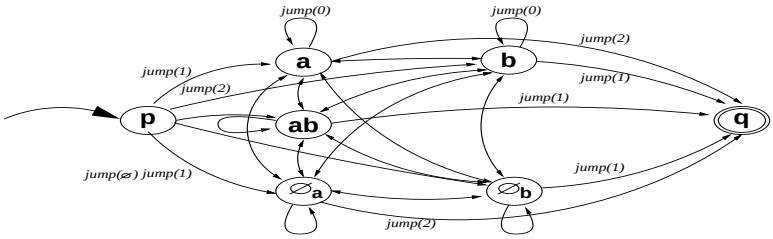

Figure 2: Schematic drawing of the PBHMM (with some transition probabilities) for the document "a b"

source	target	probability
p	$r_{i,i'}$	*jump* (i)
$r_{i,i'}$	$r_{j,j'}$	*jump* $(j - i')$
$r_{i,j'}$	q	*jump* $(m + 1 - i')$
p	$r_{\varnothing,i}$	*jump* $(\varnothing)$ *jump* (i)
$r_{\varnothing,i}$	$r_{j,j'}$	*jump* $(j - i)$
$r_{\varnothing,i}$	$r_{\varnothing,j}$	*jump* $(\varnothing)$ *jump* $(j - i)$
$r_{\varnothing,i}$	q	*jump* $(m + 1 - i)$
$r_{i,i'}$	$r_{\varnothing,j}$	*jump* $(\varnothing)$ *jump* $(j - i')$

Table 3: Jump probability decomposition

from p to $r_{i,i'}$, the value $a_{p,r_{i,i'}} =$ *jump* (i). Thus, if we begin at the first word in the document, we incur a transition probability of *jump* (1). There are no transitions into p.

2.3.4 Rewrite Probabilities

Just as the transition probabilities are governed by the jump table, the emission probabilities B are governed by the rewrite table. In general, we write $b_{x,y,\bar{k}}$ to mean the probability of generating $\bar{k}$ while transitioning from state x to state y. However, in our case we do not need the x parameter, so we will refer to these as $b_{j,\bar{k}}$, the probability of generating $\bar{k}$ when jumping into state j. When $j = r_{i,i'}$, this is *rewrite* $\left(\bar{k} \middle| d_i^{i'}\right)$. When $j = r_{\varnothing,i}$, this is *rewrite* $\left(\bar{k} \middle| \varnothing\right)$. Finally, any state transitioning into q generates the phrase $\langle\omega\rangle$ with probability 1 and any other phrase with probability 0.

Consider again the document "a b" (the PBHMM for which is shown in Figure 2) in the case when the corresponding summary is "c d". Suppose the correct alignment is that "c d" is aligned to "a" and "b" is left unaligned. Then, the path taken through the PBHMM is $p \to a \to q$. During the transition $p \to a$, "c d" is emitted. During the transition $a \to q$, ω is emitted. Thus, the probability for the alignment is: *jump* (1) *rewrite* ("cd" | "a") *jump* (2).

The rewrite probabilities themselves are governed by a mixture model with unknown mixing parameters. There are three mixture component, each of which is represented by a multinomial. The first is the standard word-for-word and phrase-for-phrase table seen commonly in machine translation, where *rewrite* $\left(\bar{s} \middle| \bar{d}\right)$ is simply a normalized count of how many times we have seen $\bar{s}$ aligned to $\bar{d}$. The second is a stem-based table, in which suffixes (using Porter's stemmer) of the words in $\bar{s}$ and $\bar{d}$ are thrown out before a comparison is made. The third is a simple identity function, which has a constant zero value when $\bar{s}$ and $\bar{d}$ are different (up to stem) and a constant non-zero value when they have the same stem. The mixing parameters are estimated simultaneously during EM.

2.3.5 Parameter Initialization

Instead of initializing the jump and rewrite tables randomly or uniformly, as it typically done with HMMs, we initialize the tables according to the distribution specified by the prior. This is not atypical practice in problems in which a MAP solution is sought.

3 Evaluation and Results

In this section, we describe an intrinsic evaluation of the PBHMM document/abstract alignment model. All experiments in this paper are done on the Ziff-Davis corpus (statistics are in Table 4). In order to judge the quality of the alignments produced by a system, we first need to create a set of "gold standard" alignments. Two human annotators manually constructed such alignments between documents and their abstracts. Software for assisting this process was developed and is made freely available. An annotation guide, which explains in detail the document/abstract alignment process was also prepared and is freely available.[4]

[4]Both the software and documentation are available on the first author's web page. The alignments are also available; contact the authors for a copy.

	Abstracts	Extracts
Documents	2033	
Sentences	$13k$	$41k$
Words	$261k$	$1m$
Types	$14k$	$26k$
	$29k$	
Sentences/Doc	6.28	21.51
Words/Doc	128.52	510.99
Words/Sent	20.47	23.77

Table 4: Ziff-Davis extract corpus statistics

3.1 Human Annotation

From the Ziff-Davis corpus, we randomly selected 45 document/abstract pairs and had both annotators align them. The first five were annotated separately and then discussed; the last 40 were done independently.

Annotators were asked to perform phrase-to-phrase alignments between abstracts and documents and to classify each alignment as either possible P or sure S, where $P \subseteq S$. In order to calculate scores for phrase alignments, we convert all phrase alignments to word alignments. That is, if we have an alignment between phrases A and B, then this induces word alignments between a and b for all words $a \in A$ and $b \in B$. Given an alignment A, we could calculate precision and recall as (see (Och and Ney, 2003)):

$$\text{Precision} = \frac{|A \cap P|}{|A|} \quad \text{Recall} = \frac{|A \cap S|}{|S|}$$

One problem with these definitions is that phrase-based models are fond of making phrases. That is, when given an abstract containing "the man" and a document also containing "the man," a human may prefer to align "the" to "the" and "man" to "man." However, a phrase-based model will almost always prefer to align the entire phrase "the man" to "the man." This is because it results in fewer probabilities being multiplied together.

To compensate for this, we define soft precision (SoftP in the tables) by counting alignments where "a b" is aligned to "a b" the same as ones in which "a" is aligned to "a" and "b" is aligned to "b." Note, however, that this is not the same as "a" aligned to "a b" and "b" aligned to "b". This latter alignment will, of course, incur a precision error. The soft precision metric induces a new, soft F-Score, labeled SoftF.

Often, even humans find it difficult to align function words and punctuation. A list of 58 function words and punctuation marks which appeared in the corpus (henceforth called the *ignore-list*) was assembled. Agreement and precision/recall have been calculated both on all words and on all words that do not appear in the ignore-list.

Annotator agreement was strong for Sure alignments and fairly weak for Possible alignments (considering only the 40 independently annotated pairs). When considering only Sure alignments, the kappa statistic (over 7.2 million items, 2 annotators and 2 categories) for agreement was 0.63. When words from the ignore-list were thrown out, this rose to 0.68. Carletta (1995) suggests that kappa values over 0.80 reflect very strong agreement and that kappa values between 0.60 and 0.80 reflect good agreement.

3.2 Machine Translation Experiments

In order to establish a baseline alignment model, we used the IBM Model 4 (Brown et al., 1993) and the HMM model (Stephan Vogel and Tillmann, 1996) as implemented in the GIZA++ package (Och and Ney, 2003). We modified this slightly to allow longer inputs and higher fertilities.

Such translation models require that input be in sentence-aligned form. In the summarization task, however, one abstract sentence often corresponds to multiple document sentences. In order to overcome this problem, each sentence in an abstract was paired with three sentences from the corresponding document, selected using the techniques described by Marcu (1999). In an informal evaluation, 20 such pairs were randomly extracted and evaluated by a human. Each pair was ranked as 0 (document sentences contain little-to-none of the information in the abstract sentence), 1 (document sentences contain some of the information in the abstract sentence) or 2 (document sentences contain all of the information). Of the twenty random examples, none were labeled as 0; five were labeled as 1; and 15 were labeled as 2, giving a mean rating of 1.75.

We ran experiments using the document sentences as both the source and the target language in GIZA++. When document sentences were used as the target language, each abstract word needed to produce many document words, leading to very high fertilities. However, since each target word is generated independently, this led to very flat rewrite tables and, hence, to poor results. Performance increased dramatically by using the document as the source language and the abstract as the target lan-

guage.

In all MT cases, the corpus was appended with one-word sentence pairs for each word where that word is translated as itself. In the two basic models, HMM and Model 4, the abstract sentence is the source language and the document sentences are the target language. To alleviate the fertility problem, we also ran experiments with the translation going in the opposite direction. These are called HMM-flipped and Model 4-flipped, respectively. These tend to out-perform the original translation direction. In all of these setups, 5 iterations of Model 1 were run, followed by 5 iterations of the HMM model. In the Model 4 cases, 5 iterations of Model 4 were run, following the HMM.

3.3 Cut and Paste Experiments

We also tested alignments using the Cut and Paste summary decomposition method (Jing, 2002), based on a non-trainable HMM. Briefly, the Cut and Paste HMM searches for long contiguous blocks of words in the document and abstract that are identical (up to stem). The longest such sequences are aligned. By fixing a length cutoff of n and ignoring sequences of length less than n, one can arbitrarily increase the precision of this method. We found that $n = 2$ yields the best balance between precision and recall (and the highest F-measure). The results of these experiments are shown under the header "Cut & Paste." It clearly outperforms all of the MT-based models.

3.4 PBHMM Experiments

While the PBHMM is based on a dynamic programming algorithm, the effective search space in this model is enormous, even for moderately sized document/abstract pairs. We selected the 2000 shortest document/abstract pairs from the Ziff-Davis corpus for training; however, only 12 of the hand-annotated documents were included in this set, so we additionally added the other 33 hand-annotate documents to this set, yielding 2033 document/abstract pairs. We then performed sentence extraction on this corpus exactly as in the MT case, using the technique of (Marcu, 1999). The relevant data for this corpus is in Table 4. We also restrict the state-space with a beam, sized at 50% of the unrestricted state-space.

The PBHMM system was then trained on this abstract/extract corpus. The precision/recall results are shown in Table 5. Under the methodology for combining the two human annotations by taking the union, either of the human scores would achieve a precision and recall of 1.0. To give a sense of how well humans actually perform on this task (in addition to the kappa scores reported earlier), we compare each human against the other.

System	SoftP	Recall	SoftF
Human$_1$	**0.727**	**0.746**	**0.736**
Human$_2$	0.680	0.695	0.687
HMM	0.120	0.260	0.164
Model 4	0.117	**0.260**	0.161
HMM-flipped	**0.295**	0.250	**0.271**
Model 4-flipped	0.280	0.247	0.262
Cut & Paste	**0.349**	**0.379**	**0.363**
PBHMM	**0.456**	**0.686**	**0.548**
PBHMM ^{O}	0.523	0.686	0.594

Table 5: Results on the Ziff-Davis corpus

One common precision mistake made by the PBHMM system is to accidentally align words on the summary side to words on the document side, when the summary word should be null-aligned. The PBHMM^O system is an oracle system in which system-produced alignments are removed for summary words that should be null-aligned (according to the hand-annotated data). Doing this results in a rather significant gain in SoftP score.

As we can see from Table 5, none of the machine translation models is well suited to this task, achieving, at best, an F-score of 0.298. The Cut & Paste method performs significantly better, which is to be expected, since it is designed specifically for summarization. As one would expect, this method achieves higher precision than recall, though not by very much. Our method significantly outperforms both the IBM models and the Cut & Paste method, achieving a precision of 0.456 and a recall nearing 0.7, yielding an overall F-score of 0.548.

4 Conclusions and Future Work

Despite the success of our model, it's performance still falls short of human performance (we achieve an F-score of 0.548 while humans achieve 0.736). Moreover, this number for human performance is a lower-bound, since it is calculated with only one reference, rather than two.

We have begun to perform a rigorous error analysis of the model to attempt to identify its deficiencies: currently, these appear to primarily be due to the model having a zeal for aligning identical words. This happens for one of two reasons: either a summary word should be null-aligned (but it is not),

or a summary word should be aligned to a different, non-identical document word. We can see the PBHMM^O model as giving us an upper bound on performance if we were to fix this first problem. The second problem has to do either with synonyms that do not appear frequently enough for the system to learn reliable rewrite probabilities, or with coreference issues, in which the system chooses to align, for instance, "Microsoft" to "Microsoft," rather than "Microsoft" to "the company," as might be correct in context. Clearly more work needs to be done to fix these problems; we are investigating solving the first problem by automatically building a list of synonyms from larger corpora and using this in the mixture model, and the second problem by investigating the possibility of including some (perhaps weak) coreference knowledge into the model.

Finally, we are looking to incorporate the results of this model into a real system. This can be done either by using the word-for-word alignments to automatically build sentence-to-sentence alignments for training a sentence extraction system (in which case the precision/recall numbers over full sentences are likely to be much higher), or by building a system that exploits the word-for-word alignments explicitly.

5 Acknowledgments

This work was partially supported by DARPA-ITO grant N66001-00-1-9814, NSF grant IIS-0097846, and a USC Dean Fellowship to Hal Daumé III. Thanks to Franz Josef Och and Dave Blei for discussions related to the project.

References

Michele Banko, Vibhu Mittal, and Michael Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL–2000)*, pages 318–325, Hong Kong, October 1–8.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Jean Carletta. 1995. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.

Hal Daumé III and Daniel Marcu. 2002a. A noisy-channel model for document compression. In *Proceedings of the Conference of the Association of Computational Linguistics (ACL 2002)*.

Hal Daumé III and Daniel Marcu. 2002b. A phrase-based HMM. Unpublished; available at `http://www.isi.edu/~hdaume/docs/daume02pbhmm.ps`, December.

J. Gauvain and C. Lee. 1994. Maximum a-posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE Transactions SAP*, 2:291–298.

Hongyan Jing and Kathleen R. McKeown. 1999. The decomposition of human-written summary sentences. In *Proceedings of the 22nd Conference on Research and Development in Information Retrieval (SIGIR–99)*, Berkeley, CA, August 15–19.

Hongyan Jing. 2002. Using hidden markov modeling to decompose human-written summaries. *Computational Linguistics*, 28(4):527 – 544, December.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1).

Christopher Manning and Hinrich Schutze. 2000. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Daniel Marcu. 1999. The automatic construction of large-scale corpora for summarization research. In *Proceedings of the 22nd Conference on Research and Development in Information Retrieval (SIGIR–99)*, pages 137–144, Berkeley, CA, August 15–19.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Hermann Ney Stephan Vogel and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, pages 836–841.