A Maximum Entropy Approach to Natural Language Processing†

Adam L. Berger
Stephen A. Della Pietra
Vincent J. Della Pietra

IBM Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Abstract

The concept of maximum entropy can be traced back along multiple threads to Biblical times. Only recently, however, have computers become powerful enough to permit the widescale application of this concept to real world problems in statistical estimation and pattern recognition. In this paper we describe a method for statistical modelling based on this concept. First we present a maximum-likelihood approach to constructing maximum entropy models. We then describe how to apply this approach, using as examples several problems in natural language processing. We contrast the maximum entropy modelling with other modelling techniques and show where it offers an advantage.

1 Introduction

Statistical modelling addresses the problem of constructing a parametrized model to predict the future behavior of a random process. In constructing this model, we typically have at our disposal a sample of output from the process. This output embodies our incomplete state of knowledge of the process; the modelling problem is to parlay this incomplete knowledge into an accurate representation of the process. Statistical modelling is rarely an end in itself, but rather a tool to aid in decision-making.

Baseball managers (who rank among the better paid statistical modellers) employ batting average, compiled from a history of at-bats, to gauge the likelihood that a player will succeed in his next appearance at the plate. Thus informed, they manipulate their lineups accordingly. Wall Street speculators (who rank among the *best* paid statistical modellers) build models based on past stock price movements to predict tomorrow's fluctuations and alter their portfolios to capitalize on the predicted future. At the other end of the pay scale reside natural language researchers, who design language and acoustic models for use in speech recognition systems and related tools.

The past few decades have witnessed significant progress toward increasing the predictive capacity of statistical models of natural language. For instance, decision tree models [1] and automatically inferred link grammars [13] address the task of modelling long range correlations in language. In parsing, Black [3] has described how to automatically extract grammatical rules from annotated text and incorporate these rules into statistical models of grammar. In speech

Research supported in part by ARPA under grant ONR N00014-91-C-0135.
†Correspondences regarding this article should be sent to aberger@watson.ibm.com or the above address.

recognition, Lucassen [18] has automatically discovered relevant features for the translation of word spelling to word pronunciation and then used these in decision tree models.

These efforts, while varied in specifics, all confront two essential tasks of statistical modelling. The first task is to determine a set of statistics which capture the behavior of the random process being considered. If the process is a sequence of words uttered by a speaker, one informative statistic is the average number of words in the speaker's sentences. Another statistic is the percentage of words that are adjectives. A third statistic (perhaps most informative) is the number of times the speaker uses each different word. Given some set of statistics such as these, representing a collection of facts about the process, the second task is to corral these facts into an accurate model of the process—a model capable of predicting the future output of the process. The first task is one of feature selection; the second is one of model selection. In the following pages we present a unified approach to these two tasks based on the maximum entropy philosophy.

The paper will proceed as follows. In Section 2 we give an overview of the maximum entropy philosophy and work through a motivating example. In Section 3 we describe the mathematical structure of maximum entropy models and give an efficient algorithm for estimating the parameters of such models. In Section 4 we discuss feature selection, and present an automatic method for discovering facts about a process from a sample of output from the process. Finally, in Section 5 we describe the application of maximum entropy ideas to several tasks in stochastic language processing: bilingual sense disambiguation, word reordering, and sentence segmentation.

2 A Maximum Entropy Overview

We introduce the concept of maximum entropy through a simple example. Suppose we wish to model an expert translator's decisions on the proper French rendering of an English word e. Our model p of the expert's decisions assigns to each French word or phrase f an estimate, $p(f)$, of the probability that the expert would choose f as a translation of e. To guide us in developing p, we collect a large sample of instances of the expert's decisions. Our goal is to extract a set of clues from the sample (this is the first task of modelling) that will aid us in constructing a model of the process (the second task).

One rather obvious clue we might glean from the sample is the list of allowed translations. If the English word e is *in*, for example, we might discover that the expert translator always chose among the following five French phrases: *dans*, *en*, *à*, *au cours de*, and *pendant*. With this information in hand, we can impose our first constraint on our model p:

$$p(dans) + p(en) + p(\grave{a}) + p(au\ cours\ de) + p(pendant) = 1$$

This equation represents our first statistic of the process; we can now proceed to search for a suitable model which obeys this equation. But there are an infinite number of models p for which this identity holds. One model sets $p(dans) = 1$; in other words, the model always predicts *dans*. Another model which obeys this constraint predicts *pendant* with a probability of 0.5, and *à* with a probability of 0.5. But both of these models offend our sensibilities: knowing only that the expert always chose from among these five French phrases, how can we justify either of these probability distributions? Each seems to be making rather bold assumptions, with no empirical justification. Put another way, these two models assume more than we actually know about the expert's decision-making process. All we know is that the expert chose exclusively from among these five French phrases; given this, the most (and only) intuitively appealing model is the following:

$$
\begin{aligned}
p(dans) &= 1/5 \\
p(en) &= 1/5 \\
p(\grave{a}) &= 1/5 \\
p(au\ cours\ de) &= 1/5 \\
p(pendant) &= 1/5
\end{aligned}
$$

This model, which allocates the total probability (1) evenly among the five possible outcomes, is the most uniform subject to our knowledge. It is not, however, the most uniform overall; that model would grant an equal probability to *every* possible French phrase.

We might hope to glean more clues from our sample about the expert's decision-making. Suppose we notice that the expert chose either *dans* or *en* 30% of the time. We could apply this knowledge to update our model of the translation process by requiring that p satisfy two constraints:

$$
\begin{gathered}
p(dans) + p(en) = 3/10 \\
p(dans) + p(en) + p(\grave{a}) + p(au\ cours\ de) + p(pendant) = 1
\end{gathered}
$$

Once again there are many probability distributions consistent with these two constraints. In the absence of any other knowledge, a reasonable choice for p is again the most uniform—that is, the distribution which allocates its probability as evenly as possible, subject to the constraints. A back-of-the-envelope calculation yields the following distribution:

$$
\begin{aligned}
p(dans) &= 3/20 \\
p(en) &= 3/20 \\
p(\grave{a}) &= 7/30 \\
p(au\ cours\ de) &= 7/30 \\
p(pendant) &= 7/30
\end{aligned}
$$

Say we inspect the data once more, and this time notice another interesting fact: in exactly half the cases, the expert chose either *dans* or *à*. We can incorporate this information into our model as an additional constraint, giving us three total:

$$
\begin{gathered}
p(dans) + p(en) = 3/10 \\
p(dans) + p(\grave{a}) = 1/2 \\
p(dans) + p(en) + p(\grave{a}) + p(au\ cours\ de) + p(pendant) = 1
\end{gathered}
$$

We can once again look for the most uniform p satisfying these constraints, but now the choice is not as obvious. As we have added complexity, we have encountered two difficulties at once. First, what exactly is meant by "uniform," and how can one measure the uniformity of a model? And, having determined a suitable answer to these related questions, how does one go about finding the most uniform model subject to a set of such constraints?

The maximum entropy method answers both these questions, as we will demonstrate in the next few pages. Intuitively, the maximum entropy idea is simple: model all that is known and assume nothing about that which is unknown. In other words, given a collection of constraining facts, choose a model consistent with all the constraints, but which otherwise is as uniform as possible. This is precisely the approach we took in selecting our model p at each step in the above example.

The maximum entropy concept has a long history. Adopting the least complex hypothesis possible is embodied in Occam's Razor: "Nunquam ponenda est pluralitas sine necesitate" [9], and even appears earlier, in the Bible and the writings of Herotodus [16]. Laplace might justly be considered the father of maximum entropy, having enunciated the underlying theme quite distinctly 200 years ago in his "Principle of Insufficient Reason": when one has no information to distinguish between the probability of two events, the best strategy is to consider them equally likely [14]. As E.T. Jaynes, a more recent pioneer of maximum entropy, put it [16]:

> *...the fact that a certain probability distribution maximizes entropy subject to certain constraints representing our incomplete information, is the fundamental property which justifies use of that distribution for inference; it agrees with everything that is known, but carefully avoids assuming anything that is not known. It is a transcription into mathematics of an ancient principle of wisdom...*

3 Maximum Entropy Modelling

3.1 Setup

We consider a random process which produces at each time sample an output value y which is a member of a finite set Y. For the translation problem just considered, the process is that of generating a translation of the word *in*, and the outcome y could be any word in the French vocabulary Y. In generating y, the process makes use of contextual information x, also a member of a finite set X. The *in* example lacked any contextual dependence for the sake of simplicity, though the model would have been able to predict the expert's preferred French translation in a given instance much more accurately by examining the context for hints. An expert will probably translate the *in* within the phrase *sometime in April* as the word *en*; on the other hand, the *in* from the phrase *perhaps in a month* is more likely to be translated as *dans*. Since any translator worth his salt uses context when selecting a translation, we cannot hope to model his decisions well with a context-ignorant model. (We are, of course, not saying anything revolutionary here. The fact that word senses are affected by neighboring words has been appreciated by machine translation researchers since the field was born half a century ago [15].)

Often, the contextual information x can be represented as a vector of values within a finite set. In the current example, the contextual information is the English words surrounding *in*; if *in* is the i^{th} word of the passage, we let x contain the identities of the words surrounding e_i. Figure 1 depicts x as a vector of size 10, containing the identities of the five words to either side of e_i. Here x_5 contains the left neighbor of *in*, and x_6 the right neighbor.

Given some sample of output from the process, our goal is to construct a parametrized conditional model which accurately represents the process. We denote the set of all conditional probability distributions which assign a probability to $y \in Y$ given $x \in X$ by $\mathcal{P}(Y|X)$ or simply

translation of e_i		e_{i-5}	e_{i-4}	e_{i-3}	e_{i-2}	e_{i-1}	e_{i+1}	e_{i+2}	e_{i+3}	e_{i+4}	e_{i+5}
y		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}

Figure 1: Example of a future y and context x

$\mathcal{P}$. One element of $\mathcal{P}$—a particular probability distribution—is denoted by $p(y|x)$. This is a slight abuse of notation, since $p(y|x)$ typically represents not an entire model, but rather the particular probability value assigned by the model to the event y given x. We will employ the same notation for both concepts; the proper interpretation in any instance will, we hope, be apparent.

3.2 Constraints and Features

To model the random process, we assume that we have gleaned some facts about the process from the sample. In the example of Section 2, we discussed several such facts: the frequency of either *dans* or *en* is $3/10$, the frequency of *dans* or *au cours de* is $1/2$, and so on. These particular facts are context-independent, but we can also consider facts which depend on x. For instance, suppose we notice an unmistakeable empirical trend that *in* translates to *en* very often when *April* is the word following *in*. We can acknowledge the importance of this fact by requiring that our model accord with it. More precisely, we constrain the expected value of the probability of *en* according to our model, given that the English word following *in* is *April*, to be the same as that in the data.

To capture this mathematically, it is convenient to introduce a binary-valued function $f(y, x)$ which takes the value 1 when y is *en* and x_6 is *April*, and 0 otherwise. That is,

$$f(y,x) = \begin{cases} 1 & \text{if } y = \textit{en} \text{ and } x_6 = \textit{April} \\ 0 & \text{otherwise} \end{cases}$$

We call f a *feature function*. The requirement that the model assign the same probability to y in the context of x as actually occurs in the data can be written as

$$\sum_x \tilde{p}(x) \sum_y p(y|x) f(y,x) = \sum_x \tilde{p}(x) \sum_y \tilde{p}(y|x) f(y,x) \tag{1}$$

Here we have adopted the convention, to be followed in the remainder of the paper, that the empirical distribution is distinguished from the model distribution by a tilde; thus, $\tilde{p}(y|x)$ represents the conditional distribution of y given x in the data; $p(y|x)$ represents the same distribution as predicted by the model. For ease of notation, we also adopt a shorthand*: $p(y,x) \equiv p(y|x)\tilde{p}(x)$.

Let $E_p[f]$ denote the expected value of $f(y, x)$ with respect to p:

$$E_p[f] \equiv \sum_{y,x} p(y,x) f(y,x)$$

Then equation (1) takes the simple form

*Unfortunately, this notation slightly obfuscates the fact that $p(y,x)$ depends not only on the model $p(y|x)$, but also on the empirical distribution $\tilde{p}(x)$. In navigating between the Scylla of unwieldy notation and Charybdis of oversimplification, we are sailing rather closer to to the latter.

$$E_p[f] = E_{\tilde{p}}[f] \tag{2}$$

We call an equation such as (1) or (2) a *constraint equation*, or simply a *constraint*. By constraining the expected values of feature functions to their empirically expected values, these equations allow us to incorporate into the model facts extracted from the sample. For the example at hand, the feature f gives rise to a constraint which enforces that the expected probability of *en*, given that the word following *in* is *April*, is the same according to the model as it appears in the sample.

We have come far since the beginning of the section: we now have a means of representing statistical phenomena inherent in a sample of data (namely, $E_{\tilde{p}}[f]$), and also a means of requiring that our model of the process exhibit these phenomena (via the constraint equations).

Typically, the collection of feature functions $f(y,x)$ is vast. By substituting any of the five other possibilities for *en* and all English words for *April*, we can realize $6V$ different features of a type akin to $f(y,x)$ above, where V is the vocabulary size of English. We can get another $6V$ by considering features that depend on the translation y and the word to the left of *in*. Or we can imagine features depending on a set of context words:

$$f(y,x) = \begin{cases} 1 & y = \textit{pendant} \text{ and } x_4,\ x_5, \text{ or } x_6 = \textit{weeks} \\ 0 & \text{otherwise} \end{cases}$$

For a vocabulary of size 20,000, one can easily imagine a universe of feature functions numbering in the millions. Trying somehow to incorporate all these functions into a model of the process seems a daunting task. Have we have bitten off more than we (or our computers) can chew?

Not exactly, for we have no intention whatsoever of using every single feature. The vast majority of features are, in fact, altogether meaningless. For instance, it hardly seems plausible that an expert translator is much influenced in his decision making by the fact that *banana* was the word which appeared three positions prior to *in*. Faced with a set of features—some important, some of marginal utility, and most useless—the methods of Section 4 will describe how to separate the wheat from the chaff. For the time being, we imagine that a genie has selected for us n of the "best" feature functions $f_1(y,x), f_2(y,x), f_3(y,x) \ldots f_n(y,x)$. Typically, n will number in the hundreds or perhaps thousands, rather than millions.

One final note about features and constraints bears repeating: though the words "feature" and "constraint" are often used interchangeably in the maximum entropy literature, we will be vigilant to distinguish the two and urge the reader to do likewise: a feature is a binary-valued function of (y,x); a constraint is an equation that imposes equality between the expected value of the feature function in the model and its expected value in the data.

3.3 The Maximum Entropy Principle

In this section we concentrate on the second task of modelling: presented with a set of constraints, how can we devise a model of the process based on these constraints?

We want the expected values with respect to the model p of each of the n feature functions to be the same as in the data. This implies a series of n linear equations, which can be written as

$$E_p[f_i] = E_{\tilde{p}}[f_i] \qquad i \in (1, 2, \ldots, n) \tag{3}$$

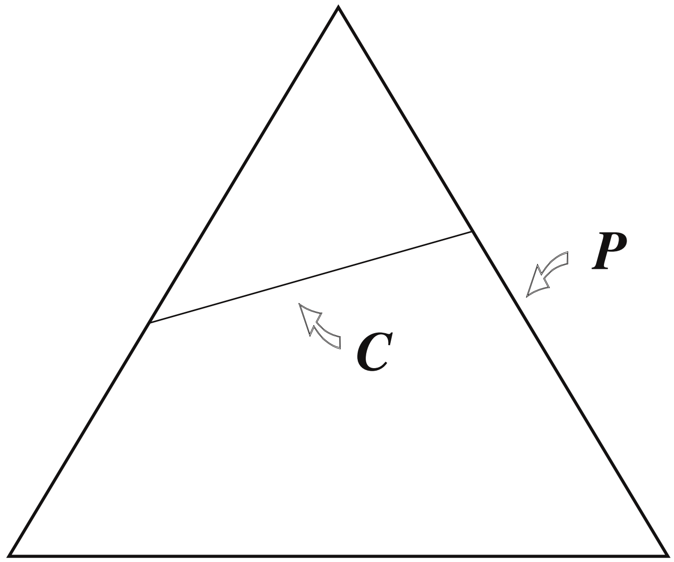

Figure 2: A linear subset $\mathcal{C}$ of the space of all probability distributions $\mathcal{P}$. In this illustration, $\mathcal{P}$ is the simplex of probability distributions on 3 points, and $\mathcal{C} \subset \mathcal{P}$ is determined by a single linear constraint.

We denote by $\mathcal{C}$ the set of all probability distributions satisfying the n linear constraints (3):

$$\mathcal{C} \equiv \{ \ p \in \mathcal{P} \mid E_p[f_i] = E_{\tilde{p}}[f_i] \ \} \qquad i \ \in \ (1, 2, \ldots, n) \tag{4}$$

Figure 2 illustrates the situation in the case that $\mathcal{P}$ is the space of probability distributions on 3 points and $\mathcal{C}$ is a subset of $\mathcal{P}$ determined by a single linear constraint.

The set $\mathcal{C}$ typically contains an infinite number of probability distributions. According to the maximum entropy philosophy, we seek that distribution which is most uniform. But now we face a question left open in Section 2: what does "uniform" mean?

A mathematical measure of the uniformity of p is provided by the Shannon entropy

$$H(p) = -\sum_{y,x} p(y,x) \log p(y|x) \tag{5}$$

[Recall that $p(y,x) = p(y|x)\tilde{p}(x)$]. From below, $H(p)$ is bounded by zero, the entropy of a model with no uncertainty at all, and from above by $H(U)$, the entropy of the uniform distribution over all possible y (where the number of distinct values y may take is given by $|Y|$):

$$0 \leq H(P) \leq H(U) = \log |Y| \tag{6}$$

We are now ready to present the principle of maximum entropy.

> To select a model p from a set $\mathcal{C}$ of allowed probability distributions, choose the distribution $p_{ME} \in \mathcal{C}$ with maximum entropy $H(p)$:
>
> $$p_{ME} = \operatorname{argmax}_{p \in \mathcal{C}} H(p) \tag{7}$$

If $\mathcal{C} = \mathcal{P}$, so that no constraints are imposed and all probability distributions are allowed, then the entropy-maximizing member of $\mathcal{C}$ is the entropy-maximizing member of $\mathcal{P}$; i.e., the uniform model U. In the opposite scenario, the constraints may be inconsistent—for instance, one constraint might require that the probability of *en* is 50% and another might require that the probability of either *en* or *pendant* is 35%. In this case, $\mathcal{C}$ is empty and no solution is possible. Typically, however, $\mathcal{C}$ is non-empty and its constraints are consistent. In this case there exists one and only one $p \in \mathcal{C}$ which satisfies equation (7).

3.4 Parametric Form

The maximum entropy principle requires us to solve a constrained optimization problem: find the p that maximizes the function $H(p)$ subject to the specified set of linear constraints. In simple cases, we can write the solution to this problem explicitly. Thus, in our example of the translations of *in*, it was easy to find the maximum entropy model in the cases of one or two constraints. Typically, however, the problem is more difficult. (The reader is invited to try to calculate the solution for the three-constraint example left unsolved in Section 2.)

Our approach to the problem is to write a parametric form for the solution and solve for the values of the parameters by using the well-known Lagrange multiplier method of constrained optimization. This technique is particularly powerful for the maximum entropy problem because the entropy function $H(p)$ is convex and the constraints $\mathcal{C}$ are linear. We will not describe the general theory here; rather, we only state the result of the analysis in the present context. The interested reader is invited to consult [12] for a mathematical account.

The bottom line of the Lagrange multiplier method (and this should be familiar to those who recall their college calculus) is to reduce the original constrained optimization problem to a simpler *unconstrained* optimization problem. To state the result, let β denote an vector of n components with one non-negative component $\beta_i \geq 0$ for each constraint function $f_i(y, x)$. Define a real-valued function $\Psi(\beta)$ by

$$\Psi(\beta) \equiv -\sum_x \tilde{p}(x) \log Z_\beta(x) + \sum_i E_{\tilde{p}}[f_i] \log \beta_i \tag{8}$$

$$Z_\beta(x) \equiv \sum_y \prod_i \beta_i^{f_i(y,x)} \tag{9}$$

Also, define a parametric family of probability models $p_\beta(y|x)$ by

$$p_\beta(y|x) \equiv \frac{1}{Z_\beta(x)} \prod_i \beta_i^{f_i(y,x)} \tag{10}$$

Note that $\sum_y p_\beta(y|x) = 1$ by virtue of the definition of $Z_\beta(x)$.

The result of the Lagrange multiplier analysis is:

> Suppose $\beta_\star$ maximizes $\Psi(\beta)$. Then $p_{\beta_\star}(y|x)$ is the model in $\mathcal{C}$ that maximizes $H(p)$.

In other words, the maximum entropy model subject to the constraints has the parametric form $p_{\beta_\star}(y|x)$, and we can find $\beta_\star$ by maximizing $\Psi(\beta)$.† The virtue of this result is that the

†It could be that the maximum of $\Psi(\beta)$ occurs at $\beta = \infty$. More precisely, there might not be any *finite* $\beta_\star$ that maximizes $\Psi(\beta)$. In this case, the maximum entropy model will not have the parametric form p_β for some finite β, but rather will be a limit of such models. More precisely, the following result is proved in [12]: If $\beta^{(n)}$ is any sequence such that $\Psi(\beta^{(n)})$ converges to the maximum of $\Psi(\beta)$, then $p_{\beta^{(n)}}$ converges to the model in $\mathcal{C}$ that maximizes $H(p)$.

task of maximizing $\Psi(\beta)$ is much simpler than the original constrained optimization problem. It can be attacked by a variety of numerical techniques; we will describe one such technique below.

In summary, a maximum entropy model is characterised by two components. First, the n features, which through their associated constraint functions compel the model to accord with the data in n different ways. Second, the n parameters β_i, one for each feature, which must be adjusted so that the constraints are satisfied.

3.5 Relation to Maximum Likelihood

It is easy to check that $\Psi(\beta)$ is in fact just the conditional log-likelihood $L(p_\beta)$ of the model p_β:

$$\Psi(\beta) \equiv L(p_\beta) = \sum_{y,x} \tilde{p}(y,x) \log p_\beta(y|x) = \log \prod_{y,x} p_\beta(y|x)^{\tilde{p}(y,x)} \tag{11}$$

(We view the log-likelihood as a measure of the ability of a model p to account for an empirical distribution $\tilde{p}$. We will write it simply as $L(p)$, omitting the $\tilde{p}$ unless a distinction needs to be made between different empirical distributions, in which case we will be more explicit, writing $L_{\tilde{p}}(p)$.)

With this interpretation, the results of the previous section can be rephrased as:

> The model $p \in \mathcal{C}$ with maximum entropy is the the model in the parametric family $p_\beta(y|x)$ that maximizes the likelihood of the sample $\tilde{p}(y,x)$.

In mathematical jargon, this is expressed by saying that the maximum entropy problem subject to linear constraints is *dual* to the maximum likelihood problem for an exponential family of models.

3.6 Computing the Parameters

Equation (10) has n parameters β_i; for all but the most simple problems, the values of these parameters that maximize $\Psi(\beta)$ cannot be found in closed form. Instead, we must resort to numerical methods. From the perspective of numerical optimization, the function $\Psi(\beta)$ is well behaved, since it is convex in the parameters $\log \beta_i$. As a result, a variety of numerical methods are available for calculating the optimal parameter values. The Brown algorithm [4] and the iterative scaling algorithm of Darroch and Ratcliff[11] are two noteworthy examples.

We present here a recent version of iterative scaling incorporating an improvement suggested by Della Pietra[12].[‡] The algorithm requires that the constraint functions be presented in such a way that

$$f_i(y,x) \geq 0 \qquad \text{for all } i, x, y$$

Condition 12 is always satisfied by the type of feature functions we are considering here—indicator functions which evaluate to one or zero. In general, given a collection of constraint

[‡]The improvement that Algorithm 1 incorporates is that in contrast to the Darroch-Ratcliff algorithm, Algorithm 1 *does not* require that

$$\sum_i f_i(y,x) = 1 \qquad \text{for all } i$$

Removing this restriction has significant practical performance implications. See [12].

functions not satisfying this condition, it is easy to construct an equivalent set of constraint functions for which condition (12) holds[11].

Algorithm 1 *Improved Iterative Scaling*

1. *Start with* $\beta_i = 1$ *for all* $i \in (1 \ldots n)$
2. *Repeat step* 3 *until all* β_i *values have converged to within a preset threshold.*
3. *For each* $i \in (1, 2, \ldots, n)$ *do the following:*
   (a) *Let* $\Delta\beta_i$ *be the solution to the following equation:*

$$\sum_{y,x} \tilde{p}(x) p_\beta(y|x) (\Delta\beta_i)^{\overline{f}(y,x)} f(y,x) = \sum_{y,x} \tilde{p}(y,x) f(y,x) \tag{12}$$

   where

$$\overline{f}(y,x) \equiv \sum_{i=1}^{n} f_i(y,x) \tag{13}$$

   (b) *Update the value of* β_i *according to:*

$$\beta_i \leftarrow \Delta\beta_i \cdot \beta_i$$

The value of $\Delta\beta_i$ that solves the equation in step 3 can be calculated in a variety of ways. For simple feature functions $f(y,x)$, the solution can sometimes be expressed in closed form. Otherwise, we can always resort to the well-known Newton method for finding the root of a polynomial.

4 Automatic Model Construction

In the introduction we divided the statistical modelling problem into two steps. The first is to find appropriate facts about the data; the second is to incorporate these facts into the model. Up to this point we have proceeded by assuming that the first task was somehow performed for us. Even in the simple example of section 2, we did not explicitly state how we selected those particular constraints: why the fact that *dans* or *à* was chosen by the expert translator 50% of the time is any more important than countless other facts contained in the data was left unaddressed. In fact, maximum entropy does not directly concern itself with this issue: it merely provides a recipe for combining constraints into a model. But the fact-selection problem is not to be sneezed at, particularly when the universe of possible constraints is in the thousands or even millions. In this section we introduce a method for automatically selecting the facts to be included in a maximum entropy model.

4.1 Feature Selection

We begin by giving a general algorithm for feature selection. Given a large sample of output from the random process we are modelling and a universe of feature functions $f(y,x)$, we could (if we had enough patience and paper) write down the constraint corresponding to each feature

$E_p[en]$	$=$	$E_{\tilde{p}}[en]$
$E_p[dans \mid x_4 = melon]$	$=$	$E_{\tilde{p}}[dans \mid x_4 = melon]$
$E_p[au\ cours\ de \mid x_2, x_6,\ or\ x_7 = around]$	$=$	$E_{\tilde{p}}[au\ cours\ de \mid x_2, x_6,\ or\ x_7 = around]$
$E_p[en \mid x_6 = trust]$	$=$	$E_{\tilde{p}}[en \mid x_6 = trust]$
$E_p[rhinocéros]$	$=$	$E_{\tilde{p}}[rhinocéros]$
$\vdots$		$\vdots$
$E_p[dans \mid x_7, x_8,\ or\ x_9 = month]$	$=$	$E_{\tilde{p}}[dans \mid x_7, x_8,\ or\ x_9 = month]$

Table 1: Subset of the universe of constraints for translation of *in*

function. If the process was an expert translator's French rendering of *in*, the result might look something like Table 1, only much more profuse.

Let us denote the complete set of constraints by $\mathcal{F}$. In an ideal world in which our knowledge is unequivocal, the maximum entropy principle specifies a *best* choice of model. It instructs us to choose the model with the greatest entropy consistent with all the constraints of $\mathcal{F}$.

Unfortunately, in the real world, our knowledge of the constraints of $\mathcal{F}$ comes from finite empirical data, and is usually far from unequivocal. Including all the constraints of $\mathcal{F}$ into our model, without regard to the certainty with which these constraints are known, would result in a model which suffers from the familar difficulty of any statistical model based on over-reliance on training data: the model would not generalize well to new data. Suppose that we inspected one thousand instances where an expert translated *in* into French, and we encounter one example in which *in* is translated as *rhinocéros*. We can postulate two reasons for this behavior. First, the particular example may be spurious (perhaps the translator had just returned from a memorable day at the zoo). Alternatively, this example may reflect some deep semantic truth relating the French word *rhinocéros* and the English word *in*.

We hope the reader will agree that the first interpretation is the more reasonable. From the fact that *rhinocéros* appeared as one of the translations of *in* it does not at all follow that the word *in* sometimes refers to a large, leathery water-dwelling mammal. In a similar vein, we should not accept every possible constraint suggested by the data as gospel. One should not draw conclusions about the underlying random process from one datum of output from the process.

In general, we would like to include in the model only a subset $\mathcal{S}$ of the constraints of $\mathcal{F}$. The choice of $\mathcal{S}$ must strike a balance between two competing objectives. On the one hand, $\mathcal{S}$ must be sufficiently rich to capture as much information about the random process as possible. On the other hand, $\mathcal{S}$ must be limited so that the model determined by $\mathcal{S}$ is robust enough to generalize to new data. Any statistic derived from an empirical sample will fall into one of two categories: either it is likely to occur in other samples of output from the same process, or it is just a random fluctuation of the data, a meaningless pattern. Our goal is to include in $\mathcal{S}$ as many constraints as possible of the first category, and as few as possible of the second.

We might first be inclined to include in $\mathcal{S}$ constraints corresponding to features which we have observed many times in our empirical data. This is not, however, an adequate criterion; even if every constraint in $\mathcal{S}$ derives from a feature that has been observed many times in data, the interplay between different constraints in $\mathcal{S}$ might implicitly impose additional constraints corresponding to features that are not observed sufficiently. We provide an example of this phenomenon in Appendix A.

A somewhat less direct approach to constraint selection turns out to be much more fruitful. We adopt an incremental approach, similar to the strategy used for growing decision trees [1].

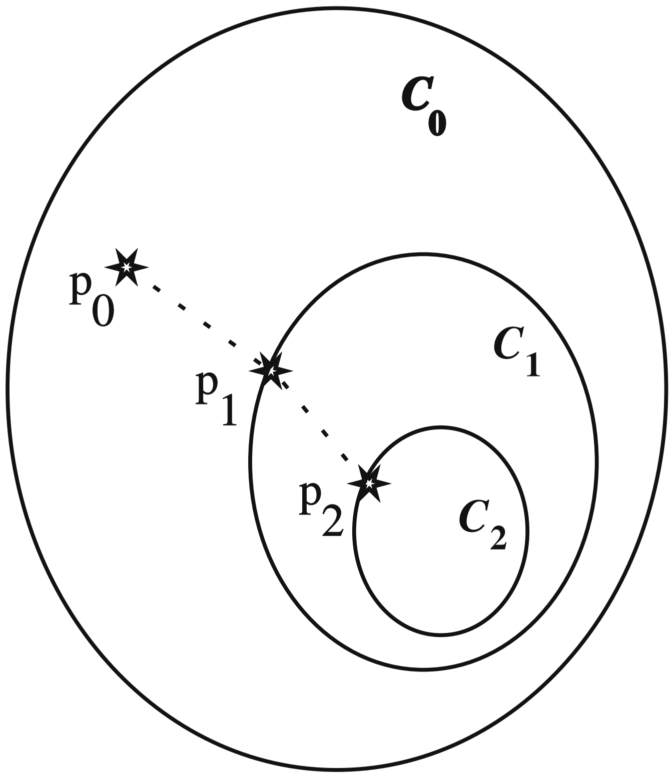

Figure 3: A nested sequence of subsets $\mathcal{C}_0 \supset \mathcal{C}_1 \supset \mathcal{C}_2 \ldots$ of $\mathcal{P}$ corresponding to increasing sets of constraints $\mathcal{S}_0 \subset \mathcal{S}_1 \subset \mathcal{S}_2 \ldots$. The point p_i represents the maximum entropy distribution in $\mathcal{C}_i$.

The idea is to build up $\mathcal{S}$ by gradually adding more and more constraints determined from the training data. As the set $\mathcal{S}$ grows, the space $\mathcal{C}$ of models allowed by the constraints in $\mathcal{S}$ shrinks, and the model p_{ME} in $\mathcal{C}$ with the greatest entropy reflects ever-increasing knowledge and thus (hopefully) becomes a more accurate representation of the process. The situation is illustrated in Figure 3, which depicts a sequence of nested subsets $\mathcal{C}_0 \supset \mathcal{C}_1 \supset \mathcal{C}_2 \ldots$ of $\mathcal{P}$ corresponding to ever-larger constraint sets $\mathcal{S}_0 \subset \mathcal{S}_1 \subset \mathcal{S}_2 \ldots$ In the figure, the point p_i represents the maximum entropy distribution consistent with all the constraints in $\mathcal{S}_i$. At every stage of the growing process we select from the remaining constraints that one which, when added to the present model, would result in the greatest increase in the conditional log-likelihood on the training data. We stop growing $\mathcal{S}$ when we can no longer find any constraint which if added to $\mathcal{S}$ results in a statistically significant improvement. We will have more to say about the stopping criterion in Section 5.3.

Algorithm 2 *Incremental Feature Selection*

1. *Begin with $\mathcal{S}$ empty*
2. *Compute the maximum entropy model $p^{\mathcal{S}}(y|x)$ that is consistent with all the constraints in $\mathcal{S}$.*
3. *For each constraint f not in $\mathcal{S}$, evaluate the gain $G(f)$ of adding f to $\mathcal{S}$ as follows:*
   (a) *Compute the maximum entropy model $p^{\mathcal{S}\cup f}$ that is consistent with f and the constraints already in $\mathcal{S}$.*
   (b) *Let $G(f)$ be the increase in conditional log-likelihood: $G(f) \equiv L(p^{\mathcal{S}\cup f}) - L(p^{\mathcal{S}})$.*
4. *Select the constraint f with the largest gain. If this gain is statistically significant, then add f to $\mathcal{S}$ and go to step 2. Otherwise, stop.*

4.2 Efficiently Computing Gains

Algorithm 2 is not a practical algorithm for incremental feature selection. For each new feature f considered in Step 3, we must compute the maximum entropy model $p^{\mathcal{S}\cup f}$, a task that is computationally costly even with the efficient iterative scaling algorithm described in Section 3.6.

We now describe a modification to the algorithm which is greedy but much more practical. We replace the exorbitant (albeit exact) computation of step $3a$ with an easier-to-compute approximation. Recall that to each constraint f_i of our model there corresponds a parameter β_i. The model $p^{\mathcal{S}}$ has one parameter for each feature in $\mathcal{S}$; the model $p^{\mathcal{S}\cup f}$ contains this set of parameters of $p^{\mathcal{S}}$ and in addition a single new parameter $\beta(f)$ corresponding to f. Given this structure, we might hope that the optimal values for the parameters of $p^{\mathcal{S}\cup f}$ corresponding to the constraints in $\mathcal{S}$ are the same as the optimal values for these parameters for the model $p^{\mathcal{S}}$. Were this the case, then imposing an additional constraint would require only the mundane task of adjusting the sole parameter $\beta(f)$ to maximize $G(f)$. Unfortunately, it so happens that all β-values change with the imposition of a new constraint.

Nevertheless, it is often reasonable to make the approximation that the addition of a new feature does not affect the optimal values for the parameters associated with the other features. Under this approximation, the model $p^{\mathcal{S}\cup f}$ can be expressed in terms of the model $p^{\mathcal{S}}$ as

$$p^{\mathcal{S}\cup f}(y|x) = \frac{1}{Z_\beta(x)} p^{\mathcal{S}}(y|x)\beta^{f(y,x)}$$

where

$$Z_\beta(x) = \sum_y p^{\mathcal{S}}(y|x)\beta^{f(y,x)}$$

The optimal value of β is determined by maximizing the likelihood of the empirical data with respect to $p^{\mathcal{S}\cup f}$. This yields

$$\sum_{y,x} f(y,x)p^{\mathcal{S}}(y|x)\beta^{f(y,x)} = E_{\tilde{p}}[f] \tag{14}$$

It is far easier to solve equation (14) for a single parameter β than to do the entire optimization problem for all the parameters in $p^{S \cup f}$. Once β is determined, the gain $G(f)$ can be computed using

$$G(f) = -\sum_x \tilde{p}(x) \log Z_\beta(x) + \log \beta E_{\tilde{p}}[f] \tag{15}$$

The actual algorithm we employ is presented in appendix B.

5 Case Studies

5.1 Background

One of the primary components of a stochastic translation system like the one described in [6] is a model that estimates the probability that a passage in one language is the translation of a passage in another language. We begin with a review of this model, henceforth referred to as the translation model. For concreteness, we consider the task of assigning probabilities to French translations of English passages.

We assume, somewhat naively, that each word in the English passage *independently* translates to zero or more French words, and these words are subsequently ordered to yield the resulting French passage. We will denote the English passage by E and the French passage by F; and individual words within each passage by lowercase letters. The association between English and French words is referred to as an *alignment* and denoted by A. Figure 4 depicts a typical alignment. For every French word $f_i \in F$, a_i is the position of the word in the English sentence which gives rise to f_i. If $a_i = j$, we say that the English word e_j *generates* the French word f_i; if the English word e_j generates n French words, we say that e_j has a *fertility* of n.

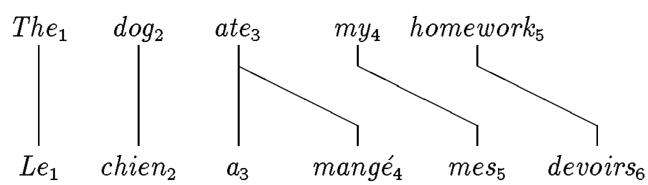

Figure 4: Alignment of a French–English sentence pair. The subscripts give the position of each word in its sentence. Here $a_1 = 1$, $a_2 = 2$, $a_3 = a_4 = 3$, $a_5 = 4$, and $a_6 = 5$.

To compute $p(F|E)$, the probability of the French passage given the English passage, we first write down an intermediate expression $p(F, A|E)$, which denotes the probability of a French passage *and* an alignment, given an English passage. To obtain $p(F|E)$, we could in principle sum this expression over all possible alignments A:

$$p(F|E) = \sum_A p(F, A|E) \tag{16}$$

However, the sum in equation (16) is computationally unwieldy; if the French and English passages have lengths $|F|$ and $|E|$, respectively, the sum extends over $|E|^{|F|}$ different alignments A. For this reason we make the simplifying assumption that there exists one extremely probable alignment $\hat{A}$, called the "Viterbi alignment," for which

$$p(F|E) \approx p(F, \hat{A}|E) \tag{17}$$

Equation (17) is sometimes referred to as the Viterbi approximation to the true conditional probability of F given E; to employ this approximation is to consider only the most likely alignment between F and E when estimating $p(F|E)$.

Given some alignment A (Viterbi or otherwise) between E and F, Brown *et. al.* has shown [5] that one can estimate $p(F, A|E)$ as a product of three terms:

$$p(F, A|E) = \prod_{i=1}^{|F|} p(n(e_{a_i})|e_{a_i}) \cdot \prod_{i=1}^{|F|} p(f_i|e_{a_i}) \cdot d(F, A|E) \tag{18}$$

Equation (18) has the following parameters:

- $p(n|e)$, the probability that the English word e generates n French words.
- $p(f|e)$, the probability that the English word e generates the French word f.
- "distortion" probabilities $d(F, A|E)$, which model how the words are reordered in the French sentence after they are generated.

Brown *et. al.* also describe a method for estimating the parameters $p(n|e)$, $p(f|e)$, and $d(F, A|E)$, given only a large bilingual corpus of unaligned words. They employ the EM algorithm, a well-known iterative method for estimating parameters of a model in the face of some "hidden" information. In this case, the hidden information is the Viterbi alignment between the French and English passages in the corpus. Starting from a uniform distribution for each of the three probability distributions comprising equation (18), the EM algorithm can simultaneously discover the Viterbi alignment $\hat{A}$ between the English and French words and determine maximum likelihood estimates for the parameters of the model. Henceforth, we will refer to this translation model as the "Brown model."

We employed the EM algorithm to estimate the translation model's parameters on 3,603,971 bilingual pairs of sentences (a little under 100 million words in each language). The data were drawn exclusively from the proceedings of the Canadian parliament, which are recorded bilingually and are referred to, for historical reasons, as "Hansards." Reviving our simple example from Section 2, we then examined the resulting model's estimates for $p(f|in)$. As Table 2 illustrates, the EM-algorithm manages this task ably. Given only the bilingual corpus, with no additional knowledge of the languages or any relation between them, the algorithm is able to learn highly plausible translations.

Though powerful, the Brown model has one major shortcoming: its probabilities are context-free. That is, the model takes no cognizance of surrounding words in estimating the probability that an expert will prefer one French phrase over another as the translation of a given English word. As we asserted in Section 3, this is not how language works. The best French translation of *in* is a function of the surrounding English words: if *a month's time* are the subsequent words, *dans* might be more likely, but if *the fiscal year 1992* are what follows, then *au cours de* is more likely. The Brown model is blind to context, always assigning a probability of 0.3004 to *dans* and 0.0044 to *au cours de*.

A method for employing the EM-estimated translation model as a component in a French-to-English automatic translation system is presented in [2]. As described there, the $p(F|E)$ model is employed in the task of translating from French to English. This "backwardness" comes about from viewing translation in the so-called source-channel formalism. Without introducing the entire formalism here, we mention only that given the French sentence, the translation task is to find an English sentence which maximizes $p(E|F)$ or, equivalently, $p(F|E) \cdot p(E)$. We are presently considering only the $p(F|E)$ term.

Translation	Probability
dans	0.3004
à	0.2275
de	0.1428
en	0.1361
pour	0.0349
(OTHER)	0.0290
au cours de	0.0233
,	0.0154
sur	0.0123
par	0.0101
pendant	0.0044

Table 2: Most frequent French translations of *in* as estimated from EM-training. (OTHER) represents a catch-all classifier for any French phrase not listed (none of which had a probability exceeding 0.0043).

Je dirais même que les chances sont supérieures à 50%.

↓

I would even say that the odds are superior to 50%.

Il semble que Bank of Boston ait pratiquement achevé son réexamen de Shawmut.

↓

He appears that Bank of Boston has almost completed its review of Shawmut.

Figure 5: Typical errors encountered in using EM-based model of Brown *et. al* in a French-to-English translation system

f	e_{i-3}	e_{i-2}	e_{i-1}	e_{i+1}	e_{i+2}	e_{i+3}
y	x_1	x_2	x_3	x_4	x_5	x_6

Figure 6: (y, x) for word translation modelling. e_i is the English word being translated; f is the putative French translation.

By way of illustration, we present in Figure 5 two examples of typical errors of the resulting system. The errors in translation are not surprising in view of the fact that the system is employing a context-ignorant model for $p(F|E)$. With no knowledge of the context, an expert translator is also quite likely to select *superior* as the English word which generates *supérieures*. But if the expert was privy to the fact that *50%* was among the next few words, he might prefer *greater* or *higher*. Similarly, for the second example, no reasonable translation system should overlook the fact that the next word is *appears* when deciding how to translate *Il*.

5.2 Word Translation

In the hope of rectifying these errors, we consider modelling $p(f|e)$ using maximum entropy methods. We envision, in practice, a separate maximum entropy model, $p_e(f|x)$, for each English word e, where $p_e(f|x)$ represents the probability that an expert translator would choose f as the French rendering of e, given the surrounding English context x. Coming up with a reasonable model $p_e(f|x)$ is a slightly recast version of a longstanding problem in computational linguistics, namely sense disambiguation—the determination of a word's sense from its context. We take the context x to be the three words on either side of e, as depicted in Figure 6.

We retain the notion of alignments, and with them the restrictive view that f_i is generated by e_{a_i} alone. But we now view the *probability* that e_{a_i} generates f_i to be a function not only of e_{a_i} itself, but also of a six-word window around e_{a_i} as well. Our hope is that by joining forces with the words around it, e_{a_i} will be better equipped to predict its French rendering.

How does this improvement fit into the general scheme of statistical machine translation? By replacing in equation (18) the term $\prod_{i=1}^{|F|} p(f_i|e_{a_i})$ by the more general term $\prod_{i=1}^{|F|} p_{e_{a_i}}(f_i|x)$, we hope to improve our estimate of $p(F, A|E)$. Ultimately, this will lead to a system that selects better translations.

Having defined the context x, all that remains is to define the set of possible features. For this application, we have settled on a simple form for the features: the indicator function f_i evaluates to 1 if and only if y is some prescribed value, and some other prescribed value appears within a subcontext of x. The two examples of features given in Section 3.2 are representative of the types of features we consider; we repeat them here for convenience.

$$f(y, x) = \begin{cases} 1 & y = en \text{ and } x_4 = April \\ 0 & \text{otherwise} \end{cases}$$

$$f(y, x) = \begin{cases} 1 & y = pendant \text{ and } x_4,\ x_5, \text{ or } x_6 = weeks \\ 0 & \text{otherwise} \end{cases}$$

Of course, these are just two examples of the types of features under consideration: the set of all possible features is vast. We thus adopt a shorthand notation when listing sets of features, displayed in Table 3. The $\Diamond$ symbols are placeholders; to enumerate all possible features, replace each $\Diamond$ by all possible values in the specified range. If V_e and V_f are the sizes of the English and

Template	Number of actual features	(y, x) Feature		
1	V_f	$y = \Diamond$		
2	$V_f \cdot V_e$	$y = \Diamond$	and	$x_4 = \Diamond$
3	$V_f \cdot V_e$	$y = \Diamond$	and	$x_3 = \Diamond$
4	$V_f \cdot V_e$	$y = \Diamond$	and	x_1, x_2,or $x_3 = \Diamond$
5	$V_f \cdot V_e$	$y = \Diamond$	and	x_4, x_5,or $x_6 = \Diamond$

Table 3: (y, x) feature templates for word-translation modelling. V_e is the size of the English vocabulary; V_f the size of the French vocabulary.

y	$x[1]$	$x[2]$	$x[3]$		$x[4]$	$x[5]$	$x[6]$
dans	*the*	*committee*	*stated*	·	*a*	*letter*	*to*
à	*work*	*was*	*required*	·	*respect*	*of*	*the*
au cours de				·	*the*	*fiscal*	*year*
dans	*by*	*the*	*government*	·	*the*	*same*	*postal*
à	*of*	*diphtheria*	*reported*	·	*Canada*	*,*	*by*
de	*not*	*given*	*notice*	·	*the*	*ordinary*	*way*

Table 4: Several actual "training events" for the maximum entropy translation model for *in*

French vocabularies respectively, then $y = \Diamond, x_3 = \Diamond$ is a shorthand for $V_e \cdot V_f$ different features. The table lists five "feature templates," where each template represents many different actual features.

Recall that a constraint is an equation which enforces that the expected value of a feature according to the model equals its expected value in some sample of data. For this experiment, the sample is a collection of 10,000 occurrences of the English word *in* culled from the Hansard corpus. Each event consists of x (the context of *in*) and the French phrase f aligned with *in* according to some previously-computed alignment between the French and English phrases. A few actual examples of events for *in* are depicted in Table 4.

We now return our attention to Table 3. Template 1 features, of which there are V_f, give rise to constraints that enforce equality between the probability of any French translation f according to the model and the probability of that translation in the actual empirical distribution. If in the Hansard events there appear n different French renderings of *in*, the Template 1 features require that

$$E_p[f_i] = E_{\tilde{p}}[f_i] \qquad i \in (1, 2, \ldots, n) \tag{19}$$

Creating a maximum entropy model using only Template 1 features amounts to predicting each French translation f_i of the English word *in* with a probability of $\tilde{p}(f_i)$; that is, the probability, estimated from the empirical data for *in*, of the French word f_i. The Template 1 constraints on the model $p_{in}(f|x)$ are:

$$
\begin{aligned}
E_p[f_i = dans] &= E_{\tilde{p}}[f_i = dans] \\
E_p[f_i = \grave{a}] &= E_{\tilde{p}}[f_i = \grave{a}] \\
E_p[f_i = de] &= E_{\tilde{p}}[f_i = de] \\
E_p[f_i = en] &= E_{\tilde{p}}[f_i = en] \\
&\vdots
\end{aligned}
$$

Template 1 features are independent of x, which means that a maximum entropy model which employs only constraints derived from Template 1 features takes no account of contextual information in assigning a probability to f. This is exactly the approach of the EM algorithm. Creating a maximum entropy translation model for *in* which matches the EM-estimated model $p(f|in)$ of Table 2 requires imposing 12 Template 1 constraints—one for each possible French rendering.

When we include constraints derived from Template 2 features, we take our first step away from the Brown model. Rather than simply constraining the expected probability of a French word f_i to equal its empirical probability, these constraints require that the expected joint probability of the English word e_k appearing in the third context position (the word immediately following *in*) and the French rendering of *in* being f_i be equal to its empirical probability. For example,

$$E_p[f_i = pendant, \quad x_5 = several] = E_{\tilde{p}}[f_i = pendant, \quad x_5 = several]$$

By adding one feature template, we have suddenly gained the ability to condition the model's probability distribution on not only e_{a_i}, but the word after e_{a_i} as well. Applying this constraint will allow the model to predict *pendant* with higher or lower probability, depending on whether *several* appears two words after *in*. We have thus achieved context-sensitive translation modelling.

Templates 3, 4 and 5 consider, each in a different way, different parts of the context. For instance, Template 5 constraints allow us to model how an expert translator is biased by the appearance of a word *somewhere* in the three words following to the word he is translating. For instance, if *house* appears within the next three words (as in the phrases *in the house* and *in the red house*), then *dans* is a more likely translation. On the other hand, if *year* appears within the same window of words (as in *in the year 1941* or *in that fateful year*), then *au cours de* is more likely. Together, the five constraint templates allow the model to condition its assignment of probabilities on a window of six words around e_{a_i}.

We constructed a maximum entropy model $p_{in}(f|x)$ by the iterative model-growing method described in Section 4. The automatic feature selection algorithm first selected a Template 1 constraint for each of the translations of *in* seen in the sample (12 in all), thus constraining the model's expected probability of each of these translations to its empirical probability. The next few constraints selected by the algorithm are shown in Table 5. The first column gives the identity of the feature whose expected value is constrained; the second column gives $G(f)$, the increase in the model's log-likelihood on the data as a result of imposing this constraint; the third column gives $L(p)$, the resulting log-likelihood.

Let us consider the fifth row in the table. This constraint requires that the model's expected probability of *dans*, if one of the three words to the right of *in* is the word *speech*, is equal to that

(y, x) *Feature*			$G(f)$	$L(p)$
y=*à*	x_4 =	*Canada*	0.041581	−2.9674
y=*à*	x_4 =	*House*	0.036177	−2.9281
y=*en*	x_4 =	*the*	0.022113	−2.8944
y=*pour*	x_4 =	*order*	0.022441	−2.8703
y=*dans*	x_4, x_5 or x_6 =	*speech*	0.019059	−2.8525
y=*dans*	x_4, x_5 or x_6 =	*area*	0.015329	−2.8377
y=*de*	x_1, x_2 or x_3 =	*increase*	0.015196	−2.8209
y=*[verb marker]*	x_4 =	*my*	0.014162	−2.8034
y=*dans*	x_4, x_5 or x_6 =	*case*	0.011694	−2.7918
y=*au cours de*	x_4, x_5 or x_6 =	*year*	0.010489	−2.7792

Table 5: Maximum entropy model to predict French translation of *in*. Features shown here were the first non Template 1 features selected. *[verb marker]* denotes a morphological marker inserted to indicate the presence of a verb as the next word.

(y, x) *Feature*			$G(f)$	$L(p)$
y=*épuiser*	x_4, x_5, or x_6 =	*out*	0.0252	−4.8499
y=*manquer*	x_4, x_5, or x_6 =	*out*	0.0221	−4.8201
y=*écouler*	x_4, x_5, or x_6 =	*time*	0.0157	−4.7969
y=*accumuler*	x_4 =	*up*	0.0149	−4.7771
y=*nous*	x_3 =	*we*	0.0140	−4.7582
y=*aller*	x_4, x_5, or x_6 =	*counter*	0.0131	−4.7445
y=*candidat*	x_4, x_5, or x_6 =	*for*	0.0124	−4.7295
y=*diriger*	x_4, x_5, or x_6 =	*the*	0.0123	−4.7146

Table 6: Maximum entropy model to predict French translation of *to run*: top-ranked features not from Template 1

in the empirical sample. Before imposing this constraint on the model during the iterative model-growing process, the log-likelihood of the current model on the empirical sample was −2.8703 bits. The feature selection algorithm described in Section 4 calculated that if this constraint were imposed on the model, the log-likelihood would rise by approximately 0.019059 bits; since this value (the "gain") was higher than for any other constraint considered, this constraint was selected. After being subjected to this additional constraint, the model's resulting likelihood was −2.8525 bits, a reduction of 0.0178 bits.

The "Hansard flavor"—the rather specific domain of parliamentary discourse related to Canadian affairs—is easy to detect in many of these features. Table 6 lists the same for the model for translating the English word *run*.

As stated above, the Template 1 constraints allow the translation models to subsume their EM-estimated predecessors. Therefore, we can expect a translation system which employs the maximum entropy models to perform at least as well as one which employs the EM-based models. Figure 7, which illustrates the type of improvements one gains in employing a maximum-entropy approach, displays the results of using the maximum entropy-based system to translate the test sentences provided earlier.

Though the maximum entropy translation model has the advantage over the EM-based

---

Je dirais même que les chances sont supérieures à 50%.

↓

I would even say that the odds are greater than 50%.

Il semble que Bank of Boston ait pratiquement achevé son réexamen de Shawmut.

↓

It appears that Bank of Boston has almost completed its review of Shawmut .

Figure 7: Improved French-to-English translations resulting from maximum entropy-based system

approach of Brown *et. al* in its ability to take context into account, one ought not dismiss the EM algorithm's utility in translation modeling. In determining the (y, x) "training events" for maximum entropy modeling, one needs a Viterbi alignment between the French and English words; this alignment is provided by the EM algorithm. So maximum entropy modelling, at least in this application, cannot hope to replace the EM algorithm. Instead, the two methods act in a complementary fashion: the EM algorithm discovers the Viterbi alignment between the French and English words; this alignment may then be employed to generate training events for estimating the parameters of maximum entropy models.

5.3 Segmentation

Though an ideal machine translation system could process input sentences of unrestricted length, a typical stochastic system must split its input into polite lengths before digesting them. If the processing time is exponential in the length of the input passage (as is the case with the translation system described in [2]), then not splitting the input into reasonably-sized segments would result in an exponential slowdown in translation.

Thus, a common task in machine translation is to find safe positions at which to split input sentences in order to speed the translation process. What we mean by the word "safe" is inexact; roughly speaking, a safe position to split the text is one which results in semantically and linguistically coherent segments. Splitting (or "segmenting") a sentence between its subject and verb is not safe; in all likelihood, the resulting segments are missing information crucial to translating them accurately. Similarly, splitting between an adjective from the noun it modifies may not be safe.

In general, *no* split is fully safe; the words of a coherent sentence represent, as any linguist will attest, nodes of a complex web of semantic and linguistic dependencies. We must cut, but our goal is to minimize damage to the web.

We consider a model which assigns to each location in a French sentence a score which is a measure of the safety (an inverse measure of the danger) in cutting the sentence at that location. Though we would like to consider the aforementioned web when deciding where to make a cut, the web is not explicit; that is, we have no web data. All that is available is a bilingual corpus with a Viterbi alignment $\hat{A}$ between the French and English. The goal is this: learn from the aligned bilingual data how to perform a "safe" cut; then, use this knowledge to segment French sentences prior to translating them.

We define a *rift* as a position i in a French sentence such that for all $j < i$, $a_j \leq a_i$ and for

$r(i)$	e_{a_i-3}	$\ldots$	e_{a_i+3}	$\mathrm{tag}(\mathrm{e}_{a_i-3})$	$\ldots$	$\mathrm{tag}(\mathrm{e}_{a_i+3})$	$\mathrm{class}(\mathrm{e}_{a_i-3})$	$\ldots$	$\mathrm{class}(\mathrm{e}_{a_i+3})$
y	$x[1]$		$x[6]$	$x[7]$		$x[12]$	$x[13]$		$x[18]$

Figure 8: (y, x) for sentence segmentation

all $j > i$, $a_j \geq a_i$. If one were to draw the alignments as in Figure 4, this condition requires that no lines cross the line connecting f_i and e_{a_i}. In other words, the words to the left of f_i are generated by words to the left of e_{a_i}, and the words to the right of f_i are generated by words to the right of e_{a_i}. In that alignment, there are rifts at positions 1, 2, 4 and 5 in the French sentence.

We make no assertions about any linguistic or semantic meaning of these rifts: we do not mean to suggest that a rift marks the boundary between two wholly independent concepts. On the other hand, it does provide us with one piece of evidence that position i is a reasonable place to divide the sentence. Lacking a better measure, we make the approximation that the probability of a rift is a gauge of the "goodness" of a split.

We now describe a method for constructing a maximum entropy model that predicts, given only a French sentence, the likelihood of a rift at each position in the sentence. Let r_i be an indicator function which evaluates to 1 if and only if a rift belongs at f_i. Thus we are modelling $p(r_i = 1|x)$, which we will denote by $p(\mathit{rift}_i|x)$.

The conditioning information x is reminiscent of that employed in the word translation application described earlier. We employ a six-word window of French words, three to the left of f_i and three to the right of f_i. The intuition behind this choice of context is not profound: to guess whether there will be any crossing alignment lines above the French word f_i, it helps to know what the surrounding French words are. In addition to these six informants, the context also includes the part-of-speech tags for these surrounding words as assigned by a stochastic part-of-speech tagger as described in [19], and the classes of these words as assigned by a mutual-information word clustering algorithm as described in [7]. The complete (y, x) pair is illustrated in Figure 8.

In creating $p(\mathit{rift}_i|x)$, we are (at least in principle) modelling the decisions of an expert French segmenter. We have a sample of his work in our empirical data $\tilde{p}$ (once again drawn from the Hansard data), and we measure of the worth of a model by the log-likelihood objective function $L(p)$. During the iterative model-growing procedure, the algorithm selects constraints on the basis of how much they increase this objective function. As the algorithm proceeds, more and more constraints are imposed on the model p, bringing it into ever-stricter compliance with the empirical data $\tilde{p}$. This is useful to a point; insofar as the empirical data embodies the expert knowledge of the French segmenter, we would like to incorporate this knowledge into a model. But the data contains only so much expert knowledge; the algorithm should terminate when it has extracted this knowledge. Otherwise, the model p will begin to fit itself to quirks in the empirical data.

A standard approach in statistical modelling to avoid this problem—the problem of "overfitting" the training data—is to separate the empirical data $\tilde{p}$ into two segments. Call one the training data, denoted by $\tilde{p}_r$, and the other held-out data, denoted by $\tilde{p}_h$. We will use only $\tilde{p}_r$ in the model-growing process; that is, we will select features based on how much they increase $L_{\tilde{p}_r}(p)$. As the algorithm progresses, $L_{\tilde{p}_r}(p)$ will thus increase monotonically. As long as each new constraint imposed allows p to better account for the random process which generated both $\tilde{p}_r$ and $\tilde{p}_h$, the quantity $L_{\tilde{p}_h}(p)$ will also increase. At the point when overfitting begins, however, the new constraints no longer help p model the random process, but instead require p to model

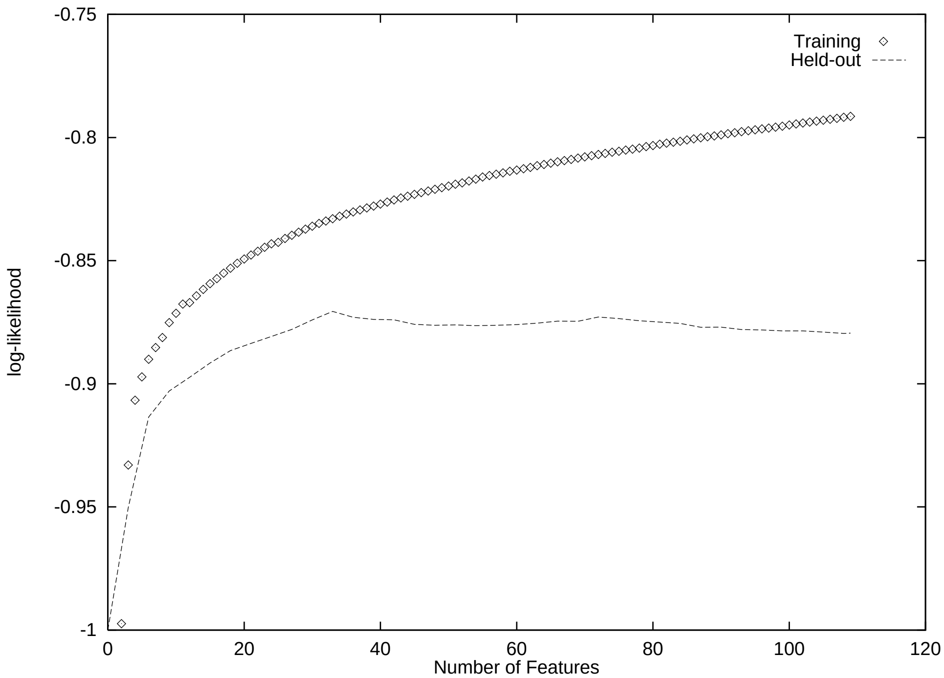

Figure 9: Change in log-likelihood during segmenting model-growing. (Overtraining begins to occur at about 40 features)

the noise in the sample $\tilde{p}_r$ itself. At this point, $L_{\tilde{p}_r}(p)$ continues to rise, but $L_{\tilde{p}_h}(p)$ no longer does. To answer a question left open at the end of Section 4.1, it is at this point that the the algorithm should terminate. The technique of checking one's progress on a held-out corpus of data is known as "cross-validation."

Figure 9 illustrates the change in log-likelihood of training data $L_{\tilde{p}_r}(p)$ and held-out data $L_{\tilde{p}_h}(p)$. Had the algorithm terminated when the log-likelihood of the held-out data stopped increasing, the final model p would contain slightly less than 40 features.

We have employed this segmenting model as a component in a French-English machine translation system in the following manner. The model assigns to each position in the French sentence a score, $p(\mathit{rift}_i|x)$, which is a measure of how appropriate a split would be at that location. A dynamic programming algorithm then selects, given the "appropriateness" score at each position and the requirement that no segment may contain more than 10 words, an optimal (or, at least, reasonable) splitting of the sentence. Table 10 shows the system's segmentation of four sentences selected at random from the Hansard data.

5.4 Word Reordering

For our third example, we consider the problem of translating French phrases of the form NOUN *de* NOUN . For example, the French phrase *taux d'intérêt* is best rendered in English as *interest rate*; the French words have switched positions and the *de* has disappeared. On the other hand, one should almost always translate *conflit d'intérêt* as *conflict of interest*, and not *interest conflict*. Table 7 gives several more examples of these types of phrases along with their English translations. To non-native speakers of English (alas, this includes computers), predicting whether the French phrase will invert seems about as difficult as predicting whether a fair penny will come up heads when dropped.

Thus, having an automatic means of determining whether a NOUN *de* NOUN phrase is in-

Monsier l'Orateur

,

j'aimerais poser une question au
Ministre des Transports.

A quelle date le
nouveau règlement devrait il entrer en vigeur?

Quels furent les crit'eres utilisés
pour l'évaluation
de ces biens.

Nous
savons que si nous pouvions contrôler la folle avoine
dans l'ouest du Canada, en
un an nous
augmenterions notre rendement en
céréales de 1 milliard de dollars.

Figure 10: Maximum entropy segmenter behavior on four sentences selected at random from the Hansard data

French	English
somme d'argent	*sum of money*
pays d'origin	*country of origin*
question de privilège	*question of privilege*
conflit d'intérêt	*conflict of interest*
bureau de poste	*post office*
taux d'intérêt	*interest rate*
compagnie d'assurance	*insurance company*
gardien de prison	*prison guard*

Table 7: NOUN *de* NOUN phrases and their English equivalents

invert?		NOUN_L	NOUN_R
y		$x[1]$	$x[2]$

Figure 11: (y, x) for prediction of English word order in translating NOUN *de* NOUN phrases

verted in its English form would certainly be a useful translation tool. For instance, one could employ this tool to preprocess a French sentence before translating it, rearranging the order of certain nouns to bring the French into closer agreement with the word-ordering rules of English. We now describe a maximum entropy-based attack on this problem.

We desire a model which, given some French NOUN *de* NOUN phrase and perhaps some other contextual information, assigns a probability to the event that in rendering an English translation of the phrase, an expert translator inverts the two nouns and drop the *de*. In contrast to the previous applications, where the model has surveyed the surrounding words in order to guide its decision, the linguistic landscape is, we assert, almost completely uninformative here: *bureau de poste* is best translated as *post office*, no matter what words precede or follow the phrase. We therefore restrict our attention to the values of the two constituent nouns. In other words, y is binary-valued (taking on the value *invert* or *don't invert*) and $x = (\text{NOUN}_L, \text{NOUN}_R)$. The setup is shown in Figure 11.

As a notational aside, we remark that in contrast to the word translation models, in which we envisioned a separate model for each English word (thus, we write $p_{in}(y|x)$ for the model which assigns a probability to *in* translating to the French word y, given the context x, and $p_{run}(y|x)$ for the corresponding model for *run*), here we imagine one model which handles *all* possible NOUN *de* NOUN phrases. In the word-translation example, including the English word in x would be redundant, since each English word e has a dedicated model p_e; here, storing the identities of NOUN_L and NOUN_R in x is essential.

Though the surrounding words are not usually helpful in predicting whether the phrase ought to be inverted, what does have a bearing on whether inversion is likely is whether the NOUN *de* NOUN pair in question has been inverted within a sample of decisions made by an expert translator. That is, a model which assigns a probability of inversion to a NOUN *de* NOUN phrase ought to assign a high probability if the phrase was found to be inverted in an empirical sample. Unfortunately, the sample may have few or no occurrences of the NOUN *de* NOUN phrase in question. For this reason we also consider what the sample says about NOUN_L alone: did, for instance, NOUN_L ever appear as the first noun in a NOUN *de* NOUN pair which inverted? If so, this may indicate that phrases which begin with NOUN_L are more likely to invert. For example, NOUN *de* NOUN phrases which begin with *système* (such as *système de surveillance* and *système de quota*) are more likely to be inverted in their English rendering. Likewise, NOUN *de* NOUN phrases which begin with *mois* (such as *mois de mai* and *moi de bonheur*) are more likely not to be inverted.

Of course, trying to determine how a particular NOUN *de* NOUN phrase ought to be handled based on past occurrences of either or both of the two nouns which comprise it is an uncertain business. For example, acting as a NOUN_R, *intérêt* can be part of an inverted phrase (*taux d'intérêt*) and also a non-inverted phrase (*conflit d'intérêt*). Relying on the past behavior of the constituent nouns is evidently not as reliable as using the past behavior of the entire NOUN *de* NOUN phrase, yet we employ it to supplement the conditioning information for the model.

Given an empirical collection of NOUN *de* NOUN phrases along with an expert's translation of each phrase (once again drawn from the Hansard corpus), we can construct overlapping sets

Set	*Noun site*	*English inverted?*
1	NOUN_L	No
2	NOUN_R	No
3	NOUN_L	Yes
4	NOUN_R	Yes
5	NOUN_L, NOUN_R	No
6	NOUN_L, NOUN_R	Yes

Table 8: Sets used in NOUN *de* NOUN word reordering

of nouns and noun pairs as depicted in Table 8. To illustrate, Set 1 contains all nouns found in the NOUN_L position of a NOUN *de* NOUN phrase in the sample whose English translation was not inverted; Set 4 contains all nouns found in the NOUN_R position of a phrase which was inverted in English. Set 5 contains all *pairs* of nouns which appeared as a NOUN *de* NOUN phrase and did not invert in English. In modelling the expert's decision, we employ these these sets, which encapsulate the information contained in a history of past occurrences, to influence the (*invert* or *don't invert?*) decision.

From the identities of the NOUN_L and NOUN_R and these sets, we can construct features of the following type:

$$f_i(y,x) = \begin{cases} 1 & \text{if } y = \textit{invert} \text{ and } \text{NOUN}_L \in \text{ Set 1} \\ 0 & \text{otherwise} \end{cases}$$

This feature constrains the model's expected probability of inversion in the case that NOUN_L is an element of Set 1 (the set of all nouns found in the NOUN_L position which did not invert) to be equal to the expected empirical value. Another type of feature checks NOUN_R:

$$f_i(y,x) = \begin{cases} 1 & \text{if } y = \textit{invert} \text{ and } \text{NOUN}_R \in \text{ Set 2} \\ 0 & \text{otherwise} \end{cases}$$

This feature constrains the model's expected probability of inversion in the case that NOUN_R is an element of Set 2 (the set of all nouns found in the NOUN_R position which did not invert) to be equal to the expected empirical value. In addition, we can consider both nouns together:

$$f_i(y,x) = \begin{cases} 1 & \text{if } y = \textit{invert} \text{ and } (\text{NOUN}_L, \text{NOUN}_R) \in \text{ Set 5} \\ 0 & \text{otherwise} \end{cases}$$

This feature constrains the model's expected probability of inversion in the case that the pair (NOUN_L, NOUN_R) is an element of Set 5 to be equal to the expected empirical value.

The model-growing algorithm terminated after imposing 358 constraints, and the progression of log-likelihood against test and training data is depicted in Figure 12.

We can consider this model as decision-making component of a French-English machine translation system in the following manner. If the model $p(\textit{invert}) > 0.5$, the model suggests inversion; otherwise, the model suggests retaining the French order. Since in a corpus of Hansard data the inversion ratio was 29.8%, a reasonable first approach to making this decision would be never to invert. Table 9 compares the error rate of the two methods on a previously-unseen portion of the Hansard data.

Table 10 shows some randomly-chosen NOUN *de* NOUN phrases extracted from this test suite along with $p(\textit{invert} \mid x)$, the probability which the model assigned to inversion. At the top of the

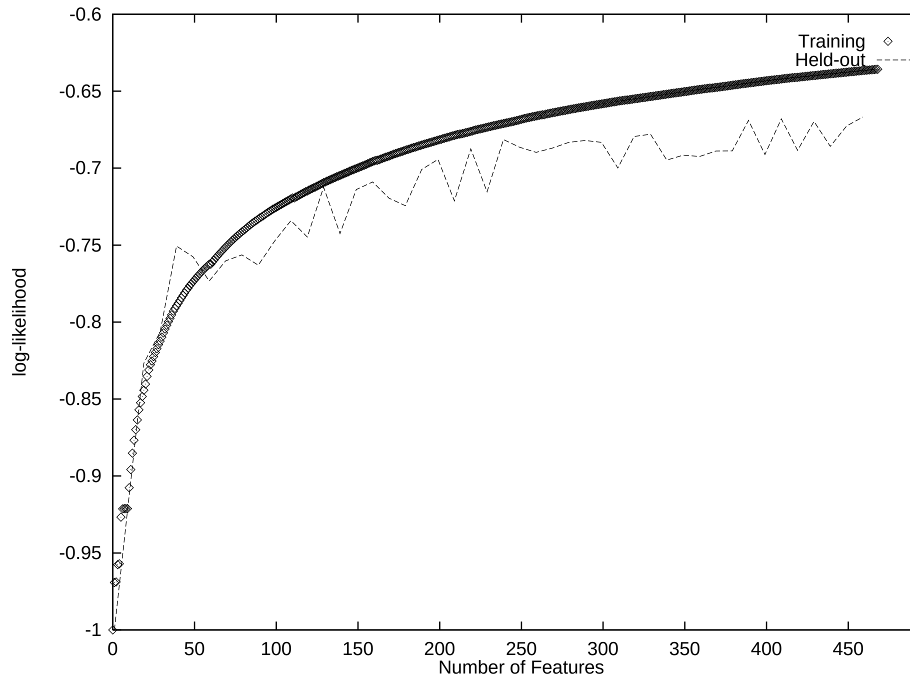

Figure 12: Change in log-likelihood during word-reordering model-growing

Test data	Simple Model Accuracy	Maximum Entropy Model Accuracy
50,229 not inverted	100%	93.5%
21,326 inverted	0%	49.2%
71,555 total	70.2%	80.4%

Table 9: NOUN *de* NOUN model performance: simple approach vs. maximum entropy

Phrase	$p(\text{invert} \mid x)$
saison d'hiver	0.997
taux de base	0.922
plan d'urgence	0.911
système de surveillance	0.851
bureau de poste	0.845
taux d'intérêt	0.724
système de quota	0.723
coût de transport	0.555
taux d'inflation	0.440
ficelle d'emballage	0.375
déclaration de candidature	0.254
niveau d'inflation	0.224
construction d'habitation	0.206
acquisition de terre	0.195
état de santé	0.167
quantité de pétrole	0.080
mois de mai	0.043
genre de travail	0.032
somme d'argent	0.018
abus de privilège	0.012
chambre de commerce	0.006

Table 10: Performance of the NOUN *de* NOUN inverter

chart are phrases such as *saison d'hiver* for which the model strongly predicted an inversion. At the bottom are phrases which the model strongly prefers not to invert, such as *somme d'argent*, *abus de privilège* and *chambre de commerce*. Perhaps most intriguing are those phrases which lie in the middle, such as *taux d'inflation*, which can translate either to *inflation rate* or *rate of inflation*.

Acknowledgments

The authors wish to thank Harry Printz and John Lafferty for suggestions and comments on a preliminary draft of this paper, and Jerome Bellegarda for providing expert French knowledge.

References

[1] Bahl, L., Brown, P., de Souza, P., Mercer, R. (1989) A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 7.

[2] Berger, A., Brown, P., Della Pietra, S., Della Pietra, V., Gillett, J., Lafferty, J., Printz, H., Ureš, L. (1994) The Candide system for machine translation. To appear in *Proceedings of the ARPA Conference on Human Language Technology*. Plainsborough, New Jersey.

[3] Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R. and Roukos, S. (1992), Towards history-based grammars: using richer models for probabilistic parsing. *Proceedings of the DARPA Speech and Natural Language Workshop*. Arden House, New York.

[4] Brown, D. (1959) A note on approximations to discrete probability distributions. *Information and Control*, vol. 2, pp. 386-392.

[5] Brown, P., Della Pietra, S., Della Pietra, V. Mercer, R. (1993), The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* vol. 19, no. 2, pp. 263-311.

[6] Brown, P., Cocke, J. Della Pietra, S., Della Pietra, V., Jelinek, F., Lafferty, J., Mercer, R., and Roossin, P. (1990), A statistical approach to machine translation. *Computational Linguistics*, vol. 16, pp. 79-85.

[7] Brown, P., Della Pietra, V, de Souza, P., Mercer, R. (1990) Class-based n-gram models of natural language. *Proceedings of the IBM Natural Language ITL*, pp. 283-298.

[8] Brown, P., Della Pietra, S., Della Pietra, V., Mercer, R. (1991) A statistical approach to sense disambiguation in machine translation. *DARPA Workshop on Speech and Natural Language*, pp. 146-151.

[9] Cover, T. and Thomas, J. (1991) *Elements of Information Theory*. John Wiley & Sons.

[10] Csiszár, L. and Tusnády, G. (1984) Information geometry and alternating minimization procedures. *Statistics & Decisions, Supplement Issue*, no. 1, pp. 205-237.

[11] Darroch, J.N. and Ratcliff, D. (1972) Generalized Iterative Scaling for log-linear models. *Annals of Mathematical Statistics*, no. 43, pp. 1470-1480.

[12] Della Pietra, S. and Della Pietra, V. (1994) Statistical modelling using maximum entropy. *IBM Research Report*, in preparation.

[13] Della Pietra, S., Della Pietra, V., Gillett, J., Lafferty, J., Printz, H., Ureš, L. (1994) Inference and estimation of a long-range trigram model. To appear in *Second International Symposium on Grammatical Inference*, Alicante, Spain.

[14] Guiasu, S. and Shenitzer, A. (1985) The principle of maximum entropy. *The Mathematical Intelligencer*, vol. 7, no. 1.

[15] Hutchins, W.J. (1986) *Machine Translation: Past, Present, Future*. John Wiley & Sons.

[16] Jaynes, E.T. (1990) Notes on present status and future prospects. In Grandy, W.T., and Schick, L.H. *Maximum Entropy and Bayesian Methods*. pp. 1-13. Kluwer.

[17] Jelinek, F. and Mercer, R. L. (1980) Interpolated estimation of Markov source parameters from sparse data. In *Proceedings, Workshop on Pattern Recognition in Practice*, Amsterdam, The Netherlands.

[18] Lucassen, J. and Mercer, R. (1984), An information theoretic approach to automatic determination of phonemic baseforms. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. 42.5.1-42.5.4. San Diego, CA.

[19] Merialdo, B. (1990) Tagging text with a probabilistic model. *Proceedings of the IBM Natural Language ITL*, Paris, France, pp. 161-172.

[20] Nádas, A. Mercer, R., Bahl, L., Bakis, R., Cohen, P., Cole, A., Jelinek, F., and Lewis, B. (1981), Continuous speech recognition with automatically selected acoustic prototypes obtained by either bootstrapping or clustering. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. pp. 1153-1155. Atlanta, GA.

Appendix

A The Danger of Implicit Constraints

In section 4.1, we discussed methods for selecting an appropriate subset of features to include in a model. Here we elaborate on why selecting all features which manifest themselves often enough in the empirical sample is not an adequate scheme.

We illustrate the problem with an example. As usual, we take up the task of translating *in* to French, given a 10-word context x. Suppose we have at our disposal an empirical sample of 2000 instances of an expert's translations. We define feature functions f_1, f_2 and $f_{1\cup 2}$ as

$$f_1(y,x) = \begin{cases} 1 & \text{if } y = \textit{dans} \text{ and } x_7 = \textit{the} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(y,x) = \begin{cases} 1 & \text{if } y = \textit{en} \text{ and } x_1 = \textit{rhinoceros} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{1\cup 2}(y,x) = \begin{cases} 1 & \text{if } (y = \textit{dans} \text{ and } x_7 = \textit{the}) \\ & \text{or } (y = \textit{en} \text{ and } x_1 = \textit{rhinoceros}) \\ 0 & \text{otherwise} \end{cases}$$

Suppose that in the sample, the feature f_1 evaluates to 1 exactly 1000 times, and the event f_2 evaluates to 1 only once. The feature $f_{1\cup 2}$ thus evaluates to 1 exactly 1001 times. The constraints corresponding to these features are:

$$E_p[f_1] = 1000/2000 \qquad E_p[f_1] = 1/2 \qquad E_p[f_{1\cup 2}] = 1001/2000$$

If we incorporate all constraints whose corresponding features evaluate to 1 often enough in the data, then incorporating f_1 forces us to take $f_{1\cup 2}$ as well. However, taking $f_{1\cup 2}$ along with f_1 implicitly constrains the expected probability of f_2 to be 1/2000. But the latter is *not* a constraint we wish to be present in our model; we have only seen the feature f_2 one time in the empirical sample, and (as we have argued) we have little basis for concluding that 1/2000 is the proper value to which to constrain the probability of this feature.

Clearly the "evaluates to 1 often enough" criterion will not suffice for determining membership in $\mathcal{S}$.

B Feature Selection Algorithm

Here we present the feature selection algorithm referred to in Section 4.

The algorithm below is based upon solving equation (14) for each using Newton's method. For a more complete discussion, see [12].

Algorithm 3 *Feature Selection*

1. *For each $f \in \mathcal{F}$, calculate the expected value of f in the training data according to*

$$\tilde{E}_0[f] = \sum_{y,x} \tilde{p}(y,x) f(y,x)$$

2. *For each x, perform steps* 3-4*.*

3. *Determine the set $\mathcal{F}(x) \subset \mathcal{F}$ of those f that are* active *for x; i.e.,*

$$\mathcal{F}(x) \equiv \{ f \in \mathcal{F} \mid f(y,x)p(y|x)\tilde{p}(x) > 0 \text{ for some } y \}$$

4. *For each $f \in \mathcal{F}$, initialize $\beta(f) \leftarrow 1$*

5. *For each f, determine $r(f)$ by*

$$r(f) = \begin{cases} 1 & \text{if } \tilde{E}_0[f] \leq E_0[f] \\ -1 & \text{if } \tilde{E}_0[f] > E_0[f] \end{cases}$$

where

$$E_0[f] = \sum_x \tilde{p}(x) \sum_y p_0(y|x) f(y,x)$$

6. *Repeat steps* 7-10 *until $\beta(f)$ converges for each $f \in \mathcal{F}$*

7. *For each $f \in \mathcal{F}$, initialize*

$$\begin{aligned} a(f) &\leftarrow \tilde{E}_0[f] \\ b(f) &\leftarrow 0 \end{aligned}$$

8. *For each x perform step* 9*.*

9. *For each $f \in \mathcal{F}(x)$, update $a(f)$ and $b(f)$ by*

$$\begin{aligned} a(f) &\leftarrow a(f) - \tilde{p}(x)\ E_{\beta,x}[f] \\ b(f) &\leftarrow b(f) - \tilde{p}(x)\ E_{\beta,x}[f - E_{\beta,x}[f]]^2 \end{aligned}$$

where

$$E_{\beta,x}[f] = \sum_y p_\beta(y|x) f(y,x)$$

10. *For each $f \in \mathcal{F}$ update $\beta(f)$ by*

$$\log \beta(f) \leftarrow \log \beta(f) + \frac{1}{r(f)} \log \left(1 - \frac{1}{r(f)} \frac{a(f)}{b(f)} \right)$$