# Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition

*Andrew Borthwick, John Sterling, Eugene Agichtein and Ralph Grishman*
Computer Science Department
New York University
715 Broadway, 7th floor
New York, NY 10003, USA
{borthwic,sterling,agichtn,grishman}@cs.nyu.edu

## Abstract

This paper describes a novel statistical named-entity (i.e. "proper name") recognition system built around a maximum entity framework. By working within the framework of maximum entropy theory and utilizing a flexible object-based architecture, the system is able to make use of an extraordinarily diverse range of knowledge sources in making its tagging decisions. These knowledge sources include capitalization features, lexical features, features indicating the current section of text (i.e. headline or main body), and dictionaries of single or multi-word terms. The purely statistical system contains no hand-generated patterns and achieves a result comparable with the best statistical systems. However, when combined with other handcoded systems, the system achieves scores that exceed the highest comparable scores thus-far published.

## 1 INTRODUCTION

Named entity recognition is one of the simplest of the common message understanding tasks. The objective is to identify and categorize all members of certain categories of "proper names" from a given corpus. The specific test bed which will be the subject of this paper is that of the Seventh Message Understanding Conference (MUC-7), in which the task was to identify "names" falling into one of seven categories: person, organization, location, date, time, percentage, and monetary amount.

This paper describes a new system called "Maximum Entropy Named Entity" or "MENE" (pronounced "meanie"). By working within the framework of maximum entropy theory and utilizing a flexible object-based architecture, the system is able to make use of an extraordinarily diverse range of knowledge sources in making its tagging decision. These knowledge sources include capitalization features, lexical features, and features indicating the current section of text. It makes use of a broad array of dictionaries of useful single or multi-word terms such as first names, company names, and corporate suffixes, and automatically handles cases where words are in more than one dictionary. Our dictionaries required no manual editing and were either downloaded from the web or were simply "obvious" lists entered by hand.

This system, built from off-the-shelf knowledge sources, contained no hand-generated patterns and achieved a result which is comparable with that of the best statistical systems. Further experiments showed that when combined with handcoded systems from NYU, the University of Manitoba, and IsoQuest, Inc., MENE was able to generate scores which exceeded the highest scores thus-far reported by any system on a MUC evaluation.

Given appropriate training data, we believe that this system is highly portable to other domains and languages and have already achieved good results on upper-case English. We also feel that there are plenty of avenues to explore in enhancing the system's performance on English-language newspaper text.

## 2 MAXIMUM ENTROPY

Given a tokenization of a test corpus and a set of $n$ (for MUC-7, $n = 7$) tags which define the name categories of the task at hand, the problem of named entity recognition can be reduced to the problem of assigning one of $4n + 1$ tags to each token. For any particular tag $x$ from the set of $n$ tags, we could be in one of 4 states: x_start, x_continue, x_end, and x_unique. In addition, a token could be tagged as "other" to indicate that it is not part of a named entity. For instance, we would tag the phrase [Jerry Lee Lewis flew to Paris] as [person_start, person_continue, person_end, other, other, location_unique]. This approach is essentially the same as (Sekine et al., 1998).

The 29 tags of MUC-7 form the space of "futures" for a maximum entropy formulation of our N.E. problem. A maximum entropy solution to this, or any other similar problem allows the computation of $p(f|h)$ for any $f$ from the space of possible futures, $F$, for every $h$ from the space of possible histories, $H$. A "history" in maximum entropy is all of the conditioning data which enables you to make a decision among the space of futures. In the named

entity problem, we could reformulate this in terms of finding the probability of $f$ associated with the token at index $t$ in the test corpus as:

$$p(f|h_t) = p\left(f|\begin{matrix}\text{Information derivable from the}\\ \text{test corpus relative to token } t\end{matrix}\right)$$

The computation of $p(f|h)$ in M.E. is dependent on a set of "features" which, hopefully, are helpful in making a prediction about the future. Like most current M.E. modeling efforts in computational linguistics we restrict ourselves to features which are binary functions of the history and future. For instance, one of our features is

$$g(h,f) = \left\{\begin{matrix} & \text{if} & \text{current-token-} \\ 1 & : & \text{capitalized(h)} = \text{true} \\ & & \text{and } f = \text{location\_start} \\ 0 & : & \text{else} \end{matrix}\right\} \quad (1)$$

Here "current-token-capitalized(h)" is a binary function which returns true if the "current token" of the history $h$ (the token whose tag we are trying to determine) has an initial capitalized letter.

Given a set of features and some training data, the maximum entropy estimation process produces a model in which every feature $g_i$ has associated with it a parameter $\alpha_i$. This allows us to compute the conditional probability as follows (Berger et al., 1996):

$$P(f|h) = \frac{\prod_i \alpha_i^{g_i(h,f)}}{Z_\lambda(h)} \quad (2)$$

$$Z_\lambda(h) = \sum_f \prod_i \alpha_i^{g_i(h,f)} \quad (3)$$

The maximum entropy estimation technique guarantees that for every feature $g_i$, the expected value of $g_i$ according to the M.E. model will equal the empirical expectation of $g_i$ in the training corpus. In other words:

$$\sum_{h,f} \tilde{P}(h,f)\cdot g_i(h,f) = \sum_h \tilde{P}(h)\cdot \sum_f P_{ME}(f|h)\cdot g_i(h,f) \quad (4)$$

Here $\tilde{P}$ is an empirical probability and $P_{ME}$ is the probability assigned by the M.E. model.

More complete discussions of M.E. as applied to computational linguistics, including a description of the M.E. estimation procedure can be found in (Berger et al., 1996) and (Della Pietra et al., 1995). The following are some additional references which are useful as introductions and examples of applications: (Ratnaparkhi, 1997b) (Ristad, 1998) (Jaynes, 1996). As many authors have remarked, though, the most useful thing about maximum entropy modeling is that it allows the modeler to concentrate on finding the features that characterize the problem while letting the M.E. estimation routine worry about assigning the relative weights to the features.

# 3 SYSTEM ARCHITECTURE: Histories and Futures

MENE consists of a set of C++ and Perl modules which forms a wrapper around a publicly available M.E. toolkit (Ristad, 1998) which computes the values of the $\alpha$ parameters of equation 2 from a pair of training files created by MENE. MENE's flexibility is due to its object-based treatment of the three essential components of a maximum entropy system: histories, futures, and features (Borthwick et al., 1997).

History objects in MENE act as containers for a list of "history views". The history view classes each represent a different type of information about the history object. When the features attempt to determine whether or not they fire on a given history, they request an appropriate history view object from the history object and then query the history view object to determine whether their firing conditions are satisfied. Note that these history views generally hold information about a limited window around the current token. If the current token is denoted as $w_0$, then our model only holds information about tokens $w_{-1} \ldots w_1$ for all history views except the lexical ones. For these views, the window is $w_{-2} \ldots w_2$.

Future objects, on the other hand, are trivial in that their only piece of data is an integer indicating which of the 29 members of the future space they represent.

# 4 FEATURES

Features are implemented as binary valued functions which query the history and future objects to determine whether or not they "fire". In the following sections, we will look at each of MENE's feature classes in turn.

## 4.1 Binary Features

While all of MENE's features have binary-valued output, the "binary" features are features whose associated history-view can be considered to be either on or off for a given token. Examples are "the token begins with a capitalized letter" or "the token is a four-digit number". Equation 1 gives an example of a binary feature. The 11 binary history-views used by MENE's binary features are very similar to those used in BBN's Nymble/Identifinder system (Bikel et al., 1997) with two exceptions:

- Nymble used a feature for "significant" (i.e. non-sentence-beginning) capitalization. We didn't include this, believing that MENE could make these judgments from the surrounding lexical content.
- Nymble's features were non-overlapping. I.e. the all-cap feature took precedence over the initial-cap feature. Given two features, $a$ and

$b$, when the (history,future) space on which feature $b$ activates must be a subset of the space for feature $a$, it can be shown that the M.E. model will yield the same results whether $a$ and $b$ are included as features or if $(a - b)$ and $b$ are features. Consequently, MENE allows all features to fire in overlapping cases. For instance, in MENE the initial cap features activate on the histories "Clinton", "IBM", and "ValuJet" while in Nymble the feature would only be active on "Clinton" because the "All-Cap" feature would take precedence on "IBM" and an "Initial-and-internal-cap" feature would take precedence on "ValuJet".

### 4.2 Lexical Features

To create a lexical history view, the tokens at $w_{-2} \ldots w_2$ are compared with a vocabulary and their vocabulary indices are recorded. For a given training corpus, we define the vocabulary to be all tokens with a count of three or more. Words not found in the vocabulary are assigned a distinguished "Unknown" index. Lexical feature example:

$$g(h,f) = \left\{ \begin{array}{lcl} & & \text{if Lexical-History-View}(token_{-1}(\text{h})) = \text{"Mr"} \\ 1 & : & \text{and } f = \text{person\_unique} \\ 0 & : & \text{else} \end{array} \right\}$$

- Correctly predicts: Mr **Jones**

A more subtle feature picked up by MENE: preceding word is "to" and future is "location_unique". Given the domain of the MUC-7 training data (aviation disasters), "to" is a weak indicator, but a real one. This is an example of a feature which MENE can make use of but which the constructor of a hand-coded system would probably regard as too risky to incorporate. This feature, in conjunction with other weak features, can allow MENE to pick up names that other systems might miss.

As discussed later, these features are automatically acquired and the system can attain a very high level of performance using these features alone. This is encouraging since these lexical features are not dependent on any external knowledge source or linguistic intuition and thus are completely portable to new domains.

### 4.3 Section Features

The New York Times articles which constituted the MUC-7 test and training corpora were composed of six distinct sections including "Date", "Preamble", and "Text". Section features activate according to which of these sections the current token is in. Example feature:

$$g(h,f) = \left\{ \begin{array}{lcl} & & \text{if Section-View}(token_0(\text{h})) \\ 1 & : & = \text{"Preamble" and } f = \\ & & \text{person\_unique} \\ 0 & : & \text{else} \end{array} \right\}$$

Activation example: CLINTON WARNS HUSSEIN ABOUT IRAQI DEFIANCE. Note that, assuming that this headline is in the preamble, the above feature will fire on *all* of these words. Of course, this feature's prediction will only be correct on "CLINTON" and "HUSSEIN".

Section features establish the background probability of the occurrence of the different futures. For instance, in NYU's evaluation system, the $\alpha$ value assigned to the feature which predicts "other" given a current section of "main body of text" is 7.9 times stronger than the feature which predicts "person_unique" in the same section. Thus the system predicts "other" by default. On the other hand, in the preamble (which contains headline, author, etc. information), the feature predicting "other" is much weaker in most cases. It is only about 2.6 times as strong as "organization_start" and "organization_end", for instance.

### 4.4 Dictionary Features

Multi-word dictionaries are a key element of MENE. Each entry in a MENE dictionary consists of a term which is one or more tokens long. Dictionaries can be case-sensitive or not on a dictionary-by-dictionary basis. A pre-processing step summarizes the information in the dictionary on a token-by-token basis by assigning to every token one of the following five tags for each dictionary: start, continue, end, unique, other. I.e. if "British Airways" was in our dictionary, a dictionary feature would see the phrase "on British Airways Flight 962" as "other, start, end, other, other". Table 1 lists the dictionaries used by MENE in the MUC-7 evaluation. Below is an example of a dictionary feature:

$$g(h,f) = \left\{ \begin{array}{lcl} & & \text{if First-Name-Dictionary-View}(token_0(\text{h})) = \\ 1 & : & \text{"unique" and } f = \text{per-son\_start} \\ 0 & : & \text{else} \end{array} \right\}$$

- Example: **Richard** M. Nixon–assuming that "Richard" is in the first name dictionary.

Note that, similar to the case of overlapping binary features, we don't have to worry about words appearing in the dictionary which are commonly used in another sense. I.e. we can leave dangerous-looking names like "April" in the first-name dictionary because whenever the first-name feature fires on "April", the lexical and date-dictionary features for "April" will also fire and, assuming that the use of April as "date" exceeded the use of April as person_start or person_unique, we can expect that the lexical feature will have a high enough $\alpha$ value to outweigh the first-name-dictionary feature. This was confirmed in our test runs: no instance of "April" was tagged as a name, including one case, "The

| Dictionary | Number of Entries | Data Source | Examples |
|---|---|---|---|
| first names | 1245 | www.babyname.com | John, Julie, April |
| corporate names | 10300 | www.marketguide.com | Exxon Corporation Motorola, Inc. |
| corporate names without suffixes | 10300 | "corporate names" processed through a Perl script | Exxon; Motorola |
| colleges and universities | 1225 | http://www.utexas.edu/world/univ/alpha/ | New York University; Oberlin College |
| Corporate Suffixes | 244 | Tipster resource | Inc.; Incorporated; AG |
| Dates and times | 51 | Hand Entered | Wednesday, April, EST, a.m. |
| 2-letter State Abbreviations | 50 | www.usps.gov | NY, CA |
| World Regions | 14 | www.yahoo.com | Africa, Central America, Caribbean, Pacific Rim |

Table 1: Dictionaries used in MENE

death of Ron Brown in April in a similar plane crash ..." which could be thought of as somewhat tricky because the month was not followed by a specific date. Note that the system isn't foolproof: if a "dangerous" dictionary word appeared in only one dictionary and did not appear often enough in the training corpus to be included in the vocabulary, but did appear in the test corpus, we would probably mistag it.

### 4.5 External System Features

For NYU's official entry in the MUC-7 evaluation, MENE took in the output of an enhanced version of the more traditional, hand-coded "Proteus" named-entity tagger which we entered in MUC-6(Grishman, 1995). In addition, subsequent to the evaluation, the University of Manitoba (Lin, 1998) and IsoQuest, Inc. (Krupka and Hausman, 1998) shared with us the outputs of their systems on our training corpora as well as on various test corpora. The output sent to us was the standard MUC-7 output, so our collaborators didn't have to do any special processing for us. These systems were incorporated into MENE as simply three more history views by the following 2 step process:

1. Each system's output is tokenized by MENE's tokenizer and cross-system tokenization discrepancies are resolved.
2. The tag assigned to each token by each system is noted. This tag will be one of the 29 tags mentioned above (i.e. person_start, location_continue, etc.)

The result of all this is that the "futures" produced by the three external systems become three "external system histories" for MENE. Here is an example feature:

$$g(h,f) = \begin{cases} 1 & : \text{if Proteus-System-View}(token_0(\text{h})) = \text{"person\_start"} \text{ and } f = \text{person\_start} \\ 0 & : \text{else} \end{cases}$$

- Example: **Richard** M. Nixon, in a case where Proteus has correctly tagged "Richard".

It is important to note that MENE has features which predict a different future than the future predicted by the external system. This can be seen as the process by which MENE learns the errors which the external system is likely to make. An example of this is that on the evaluation system the feature which predicted person_unique given a tag of person_unique by Proteus had only a 76% higher weight than the feature which predicted person_start given person_unique. In other words, Proteus had a tendency to chop off multi-word names at the first word. MENE learned this and made it easy to override Proteus in this way. In fact, an analysis of the differences between the Proteus output and the MENE + Proteus output turned up a significant number of instances in which MENE extended or contracted name boundaries in this way. Given proper training data, MENE can pinpoint and selectively correct the weaknesses of a handcoded system.

## 5 Compound Features

MENE currently has no direct ability to learn compound features or "patterns"–the "history" side of a lexical feature activates based on only a single word, for instance. A sort of pattern-like ability comes into the system from multiple features firing at once. I.e. to predict that "York" in the name "New York" is the end of a location, we will have two features firing: one predicts location_end when $token_{-1}$ is

"new". The other predicts location_end when $token_0$ is "york".

Nevertheless, it is possible that compound features would behave differently from two simultaneously firing "atomic" features. We integrated this into the model in an *ad hoc* manner for the external system features, where we constructed features which essentially query the external system history and the section history simultaneously to determine whether they fire. I.e. a particular feature might fire if Proteus predicts person_start, the current section is "main body of text", and the future is "person_start". This allows MENE to assign a lower $\alpha$ to a Proteus prediction in the preamble vs. a prediction in the main body of text. Proteus, like many hand-coded systems, is more accurate in the main body of the text than in headline-type material. We found that this compound feature gave the system slightly higher performance than we got when we just used section features and external system features separately.

It seems reasonable that adding an ability to handle fully general compound features (i.e. feature $A$ fires if features $B$ and $C$ both fire) would improve system performance based on this limited experiment. In addition to allowing us to predict futures based on multi-word patterns, it would also let us use other promising combinations of features such as distinguishing between capitalization in a headline vs. in the main body of the text. Unfortunately, this experiment will have to wait until we deploy a more sophisticated method of feature selection, as discussed in the next section.

## 6 FEATURE SELECTION

Features are chosen by a very simple method. All possible features from the classes we want included in our model are put into a "feature pool". For instance, if we want lexical features in our model which activate on a range of $token_{-2} \ldots token_2$, our vocabulary has a size of $V$, and we have 29 futures, we will add $(5 \cdot (V+1) \cdot 29)$ lexical features to the pool. The $V+1$ term comes from the fact that we include all words in the vocabulary plus the unknown word. From this pool, we then select all features which fire at least three times on the training corpus. Note that this algorithm is entirely free of human intervention. Once the modeler has selected the classes of features, MENE will both select all the relevant features and train the features to have the proper weightings.

We deviate from this basic algorithm in three ways:

1. We exclude features which activate on some sort of "default" value of a history view. Many history views have some sort of default value which they display for the vast majority of tokens. For instance, a first-name-dictionary history view would say that the current token is not a name in over 99% of the cases. Rather than adding features which activate both when the token in question is and when it is not a first name, we only include features which activate when the token is a first name. A feature which activated when a token was not a first name, while theoretically not harmful, would have practical disadvantages. First of all, the feature would probably be redundant, because if the frequency of a future given a first-name-dictionary hit is constrained (by equation 4), then the future frequency given a non-hit is also implicitly constrained. Secondly, since this feature would fire on nearly every token, it would slow down run-time performance. Finally, while maximum entropy models are designed to handle feature overlap, a very high degree of overlap requires more iterations of the maximum entropy estimation routine and can lead to numerical difficulties (Ristad, 1998).
2. Features which predict the future "other" have to fire six times to be included in the model rather than three. Experiments showed that doing this had no impact on performance and reduced the size of the model by about 20%.
3. As another way of reducing the model size, lexical features which activate on $token_{-2}$ and $token_2$ are excluded if they predict "other". Like the previous heuristic, this is based on the idea that features predicting named entities are more useful than features predicting the default.

Note that this method of feature selection would probably break down if we tried to incorporate general compound features into our model as described in the previous section. The model currently has about 24,000 features when trained on 350 articles of text. If we even considered all pairs of features as potential compound features, the $O(n^2)$ compound features which we could build from our atomic features would undoubtedly yield an unacceptable slowdown in the model's performance. Clearly a more sophisticated feature selection routine such as the ones in (Berger et al., 1996), or (Berger and Printz, 1998) would be required in this case.

## 7 DECODING and VITERBI SEARCH

After having trained the features of an M.E. model and assigned the proper weight ($\alpha$ values) to each of the features, decoding (i.e. "marking up") a new piece of text is a fairly simple process:

1. Tokenize the text.

2. Compute each of the history views by looking up words in the dictionary, checking the output of the external systems, checking whether words are capitalized or not, etc.
3. For each article of the text
   (a) For each token of the text, check each feature to see whether it fires, and combine the $\alpha$ values of the firing features according to equation 2. This will give us a conditional probability for each of the 29 futures for each token in the article.
   (b) Run a Viterbi search to find the highest probability legal path through the lattice of conditional probabilities.

The Viterbi search is necessary because simply taking the highest-probability future assigned to each token would result in incompatible assignments. For instance, an assignment of [person_start, location_end] to two consecutive tokens would be invalid. The Viterbi search finds the highest probability path in which there are no two tokens in which the second one cannot follow the first, as defined by a table of all such invalid transitions (a similar approach to (Sekine et al., 1998)).

## 8 RESULTS

MENE's maximum entropy training algorithm gives it reasonable performance with moderate-sized training corpora or few information sources, while allowing it to really shine when more training data and information sources are added. Table 2 shows MENE's performance on the MUC-7 "dry run" corpus, which consisted of 25 articles mostly on the topic of aviation disasters. All systems shown were trained on 350 articles on the same domain (this training corpus consisted of about 270,000 words, which our system turned into 321,000 tokens).

Note the smooth progression of the scores as more data is added to the system. Also note that, when combined under MENE, the three weakest systems, MENE, Proteus, and Manitoba outperform the strongest single system, IsoQuest's. Finally, the top score of 97.12 from combining all three systems is a very strong result. On a different set of data, the MUC-7 formal run data, the accuracy of the two human taggers who were preparing the answer key was tested and it was discovered that one of them had an F-Measure of 96.95 and the other of 97.60 (Marsh and Perzanowski, 1998). Although we don't have human performance measures on the dry run test set, it seems that we have attained a result which is at least competitive with that of a human.

We also did a series of runs to examine how the systems performed with different amounts of training data. These experiments are summarized in table 3. Note the 97.38 all-systems result which we achieved by adding 75 articles from the formal-run test corpus to the basic 350-article training data. In addition to being an outstanding performance figure, this number shows MENE's responsiveness to good training material. A few other conclusions can be drawn from this data. First of all, MENE needs at least 20 articles of tagged training data to get acceptable performance on its own. Secondly, there is a minimum amount of training data which is needed for MENE to improve an external system. For Proteus and the Manitoba system, this number seems to be about 80 articles, because they show a degradation of performance at 40. Since the IsoQuest system was stronger to start with, MENE required 150 articles to show an improvement. Note the anomaly in comparing the 250 and 350 article columns. Proteus shows only a very small gain and IsoQuest shows a deterioration. These last 100 articles added to the system were tagged by us at NYU, and we would humbly guess that we tagged them less carefully than the rest of the data which was tagged by BBN and Science Applications International Corporation (SAIC).

| Systems | F-Measure | Precision | Recall |
|---|---|---|---|
| MENE (ME) | 92.20 | 96 | 89 |
| IsoQuest (IQ) | 96.27 | 98 | 94 |
| Manitoba (Ma) | 93.32 | 94 | 92 |
| Proteus (Pr) | 92.24 | 95 | 90 |
| MENE + IsoQuest | 96.55 | 98 | 95 |
| MENE + Proteus | 95.61 | 97 | 94 |
| MENE + Manitoba | 95.49 | 97 | 94 |
| ME + Ma + IQ | 96.81 | 98 | 95 |
| ME + Pr + IQ | 96.78 | 98 | 96 |
| ME + Pr + Ma | 96.48 | 97 | 95 |
| ME + Pr + Ma + IQ | 97.12 | 98 | 96 |

Table 2: Combined systems on unseen data from the MUC-7 dry-run test set

MENE has also been run against all-uppercase data. On this we achieved an F-measure of 88.19 for the MENE-only system and 91.38 for the MENE + Proteus system. The latter figure matches the best currently published result (Bikel et al., 1997) on within-domain all-caps data. On the other hand, we scored lower on all-caps than BBN's Identifinder in the MUC-7 formal evaluation for reasons which are probably similar to the ones discussed in section 9 in the comparison of our mixed case performances (Miller et al., 1998) (Borthwick et al., 1998). We have put very little effort into optimizing MENE on

| Systems | 425 | 350 | 250 | 150 | 100 | 80 | 40 | 20 | 10 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|
| MENE | 92.94 | 92.20 | 91.32 | 90.64 | 89.17 | 87.85 | 84.14 | 80.97 | 76.43 | 63.13 |
| MENE + Proteus | 95.73 | 95.61 | 95.56 | 94.46 | 94.30 | 93.44 | 91.69 | | | |
| MENE + Manitoba | 95.60 | 95.49 | 95.26 | 94.86 | 94.50 | 94.15 | 93.06 | | | |
| MENE + IsoQuest | 96.73 | 96.55 | 96.70 | 96.55 | 96.11 | | | | | |
| ME + Pr + Ma + IQ | 97.38 | 97.12 | | | | | | | | |

Table 3: Systems' performances with different numbers of articles

this type of corpus and believe that there is room for improvement here.

In another experiment, we stripped out all features other than the lexical features and still achieved an F-measure of 88.13. Since these features do not rely on any external knowledge sources and are automatically generated, this result is a strong indicator of MENE's portability.

The MUC-7 formal evaluation involved a shift in topic which was not communicated to the participants beforehand–the training data focused on airline disasters while the test data was on missile and rocket launches. MENE fared much more poorly on this data than it did on the within-domain data quoted above, achieving an F-measure of only 88.80 for the MENE + Proteus system and 84.22 for the MENE-only system. While 88.80 was still the fourth highest score out of the twelve participants in the evaluation, we feel that it is necessary to view this number as a cross-domain portability result rather than as an indicator of how the system can do on unseen data within its training domain. We believe that if the system had been allowed to train on missile/rocket launch articles, its performance on these articles would have been much better. More MENE test results and discussion of the formal run can be found in (Borthwick et al., 1998).

## 9 RELATED WORK

M.E. has been successfully applied to many other tasks in computational linguistics. Some recent work for which there are solid comparable benchmarks is the work of Adwait Ratnaparkhi at the University of Pennsylvania. He has achieved state-of-the art results by applying M.E. to parsing (Ratnaparkhi, 1997a), part-of-speech tagging (Ratnaparkhi, 1996), and sentence-boundary detection (Reynar and Ratnaparkhi, 1997). Other recent work has applied M.E. to language modeling (Rosenfeld, 1994), machine translation (Berger et al., 1996), and reference resolution (Kehler, 1997). M.E. was first applied to named entity recognition at the MUC-7 conference by (Borthwick et al., 1998) and (Mikheev and Grover, 1998).

Note that part-of-speech tagging is, in many ways, a very similar task to that of named-entity recognition. Ratnaparkhi's tagger is similar to MENE, in that his features look at the surrounding two-word lexical context, but his system makes less use of dictionaries. On the other hand, his system looks at word suffixes and prefixes in the case of unknown words, which is something we haven't tried with MENE and looks at its own output by looking at its previous two tags when making its decision. We do this implicitly through our requirement that the futures we output be consistent, but we found that an attempt to do this more directly by building a consistency feature directly into the model had no effect on our results.

At the MUC-7 conference, there were two other interesting systems using statistical techniques from the Language Technology Group/University of Edinborough (Mikheev and Grover, 1998) and BBN (Miller et al., 1998). Comparisons with the LTG system are difficult since it was a hybrid model in which the text was passed through a five-stage process, only three of which involved maximum entropy and over half of the system's recall came from the two non-statistical phases. The LTG system demonstrated superior performance on the formal run relative to the MENE-Proteus hybrid system (93.39 vs 88.80), but it isn't clear whether their advantage came from superior handcoded rules or superior statistical techniques, because their system is not as easily broken down into separate components as is MENE-Proteus. It is also possible that tighter system integration between the statistical and handcoded components was responsible for some of LTG's relative advantage, but note that MENE-Proteus appears to have an advantage over LTG in terms of portability. We are currently experimenting with porting MENE to Japanese, for instance, and expect that it could be combined with a pre-existing Japanese handcoded system, but it isn't clear that this could be done with the LTG system. Nevertheless, one of our avenues for future research is to look at tighter multi-system integration methods which won't compromise MENE's essential portability.

Table 4 gives a comparison of BBN's HMM-based Identifinder (Miller et al., 1998) and NYU's MENE and MENE-Proteus systems on different training and test sets. We are not sure why MENE-Proteus was hurt more badly by the evaluation-time switch from aviation disaster articles to mis-

| Conditions | Identifinder | MENE | MENE + Proteus |
|---|---|---|---|
| Trained on official training data Tested on dry run (within domain) | 92.5 | 89.17 | 94.30 |
| Each organization trained on all of its own data and tested on dry run | 95.1 | 92.20 | 95.61 |
| Same as above, but run against official MUC-7 data | 90.44 | 84.22 | 88.80 |

Table 4: Comparison of BBN and NYU statistical systems

sile/rocket launch articles, but suspect that it may have been due to Identifinder's greater quantity and quality of training data. BBN used 790,000 words of training data to our 321,000. The quality advantage may have come from selecting sentences from a larger corpus for their annotators to tag which were chosen so as to increase the variety of training data.

When MENE-only and Identifinder are compared training on the same number of articles and testing on within-domain data, Identifinder still has an edge. We speculate that this is due to the dynamic updating of Identifinder's vocabulary during decoding when person or organization names are recognized, which gives the system a sort of long-distance reference resolution which is lacking in MENE. In addition, BBN's HMM-based system implictly predicts named entities based on consecutive pairs of words rather than based on single words, as is done in MENE, because each type of name has its own bigram language model. In the decoding process, the Viterbi algorithm chooses the sequence of names which yields the highest joint probability of names, words, and features associated with each word.

In comparing the maximum entropy and HMM-based approaches to named entity recognition, we are hopeful that M.E. will turn out to be the better method in the end. We think it is possible that some of Identifinder's current advantage can be neutralized by simply adding the just-mentioned features to MENE. On the other hand, we have a harder time seeing how some of MENE's strengths can be integrated into an HMM-based system. It is not clear, for instance, how a wide variety of dictionaries could be added to Identifinder or whether the system could be combined with a handcoded system as was done with our system and the one from LTG.

## 10 CONCLUSIONS AND FUTURE WORK

MENE is a very new, and, we feel, still immature system. Work started in October, 1997, and the system described above was not in place until mid-February. 1998. We believe that we can push the score of the MENE-only system higher by incorporating long-range reference-resolution on MENE's output. We are also missing a large number of acronyms which could be picked up by dynamically building them from entities which MENE had tagged elsewhere and then pulling that data in as a new class of feature. The other key element missing from the current system is a set of general compound features, which, as discussed above, would require the use of a more sophisticated feature selection algorithm. All three of these elements are present in systems such as IsoQuest's (Krupka and Hausman, 1998), and their absence from MENE probably explains much of the reason why the MENE-only system failed to perform at the state-of-the-art. We intend to add all of these elements to MENE in the near future to test this hypothesis.

Nevertheless, we believe that we have already demonstrated some very useful results. MENE is highly portable, as we have already demonstrated with our result on upper-case English text and even in its current state, its results are already comparable to that of the only other purely statistical English NE system which we are aware of (Miller et al., 1998). As shown with our result on running MENE with only the lexical features that it learns from the training corpus, porting MENE can be done with very little effort if appropriate training data is provided–it isn't even necessary to provide it with dictionaries to generate an acceptable result. We are working on a port to Japanese NE to further demonstrate MENE's flexibility.

However, we believe that the results on combining MENE with other systems are some of the most intriguing. We would hypothesize that, given sufficient training data, any handcoded system would benefit from having its output passed to MENE as a final step. MENE also opens up new avenues for collaboration whereby different organizations could focus on different aspects of the problem of N.E. recognition with the maximum entropy system acting as an arbitrator. MENE also offers the prospect of achieving very high performance with very little effort. Since MENE starts out with a fairly high base score just on its own, we speculate that a MENE user could then construct a hand-coded system which only focused on MENE's weaknesses, while skipping the areas in which MENE is already strong.

Finally, one can imagine a user acquiring licenses

to several different N.E. systems, generating some training data, and then combining it all under a MENE-like system. We have shown that this approach can yield performance which is competitive with that of a human tagger.

## 11 ACKNOWLEDGMENTS

We would like to thank Troy Straszheim for writing the Viterbi search routine used in this work.

## References

Adam Berger and Harry Printz. 1998. A comparison of criteria for maximum entropy/minimum divergence feature selection. In Nancy Ide and Atro Boutilainen, editors, *Proceedings of the Third Conference on Empirical Methods in Natural Language Processing*, pages 97–106. The Association for Computation Linguistics, June.

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Fifth Conference on Applied Natural Language Processing.*

Andrew Borthwick, Radu Florian, and Kishore Papineni. 1997. The system architecture of MENE was strongly influenced by the architecture of a maximum entropy language model jointly developed by these three authors at IBM Watson Labs, Yorktown Heights, New York, in the summer of 1997.

Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. 1998. Nyu: Description of the mene named entity system as used in muc-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7).*

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1995. Inducing features of random fields. Technical Report CMU-CS-95-144, Carnegie Mellon University.

Ralph Grishman. 1995. The nyu system for muc-6 or where's the syntax? In *Proceedings of the Sixth Message Understanding Conference.* Morgan Kaufmann, November.

Edwin T. Jaynes. 1996. Probability theory: The logic of science. Manuscript for book. Available at ftp://bayes.wustl.edu/, May.

Andrew Kehler. 1997. Probabilistic coreference in information extraction. In *Empirical Methods in Natural Language Processing*, volume 2, August.

George R. Krupka and Kevin Hausman. 1998. Isoquest: Description of the netowl(tm) extractor system as used in muc-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7).*

Dekang Lin. 1998. Using collocation statistics in information extraction. In *Proceedings of the Seventh Message Understanding Conference (MUC-7).*

Elaine Marsh and Dennis Perzanowski. 1998. Muc-7 evaluation of i.e. technology: Overview of results. In *Proceedings of the Seventh Message Understanding Conference (MUC-7).*

Andrei Mikheev and Claire Grover. 1998. Ltg: Description of the ne recognition system used for muc-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, April.

Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and the Annotation Group. 1998. Algorithms that learn to extract information–bbn: Description of the sift system as used for muc-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, April.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*, pages 133–142. University of Pennsylvania, May.

Adwait Ratnaparkhi. 1997a. A linear observed time statistical parser based on maximum entropy models. In *Conference on Empirical Methods in Natural Language Processing.* University of Pennsylvania, June.

Adwait Ratnaparkhi. 1997b. A simple introduction to maximum entropy models for natural language processing. Technical Report 97-08, Institute for Research in Cognitive Science, University of Pennsylvania, May.

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence structures. In *Fifth Conference on Applied Natural Language Processing*, pages 16–19, April.

Eric Sven Ristad. 1998. Maximum entropy modeling toolkit, release 1.6 beta. http://www.mnemonic.com/software/memt, February. Includes documentation which has an overview of MaxEnt modeling.

Ronald Rosenfeld. 1994. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach.* Ph.D. thesis, Carnegie Mellon University. CMU Technical Report CMU-CS-94-138.

Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. 1998. A decision tree method for finding and classifying names in japanese texts. In *Proceedings of the Sixth Workshop on Very Large Corpora.*