Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence

Silviu Cucerzan and **David Yarowsky**
Department of Computer Science
Center for Language and Speech Processing
Johns Hopkins University
Baltimore, Maryland, 21218
{silviu,yarowsky}@cs.jhu.edu

Abstract

Identifying and classifying personal, geographic, institutional or other names in a text is an important task for numerous applications. This paper describes and evaluates a language-independent bootstrapping algorithm based on iterative learning and re-estimation of contextual and morphological patterns captured in hierarchically smoothed trie models. The algorithm learns from unannotated text and achieves competitive performance when trained on a very short labelled name list with no other required language-specific information, tokenizers or tools.

1 Introduction

The ability to determine the named entities in a text has been established as an important task for several natural language processing areas, including information retrieval, machine translation, information extraction and language understanding. For the 1995 Message Understanding Conference (MUC-6), a separate named entity recognition task was developed and the best systems achieved impressive accuracy (with an F-measure approaching 95%). What should be underlined here is that these systems were trained for a specific domain and a particular language (English), typically making use of hand-coded rules, taggers, parsers and semantic lexicons. Indeed, most named entity recognizers that have been published either use tagged text, perform syntactical and morphological analysis or use semantic information for contextual clues. Even the systems that do not make use of extensive knowledge about a particular language, such as Nominator (Choi et al., 1997), still typically use large data files containing lists of names, exceptions, personal and organizational identifiers.

Our aim has been to build a maximally language-independent system for both named-entity identification and classification, using minimal information about the source language. The applicability of AI-style algorithms and supervised methods is limited in the multilingual case because of the cost of knowledge databases and manually annotated corpora. Therefore, a much more suitable approach is to consider an EM-style bootstrapping algorithm. In terms of world knowledge, the simplest and most relevant resource for this task is a database of known names. For each entity class to be recognized and tagged, it is assumed that the user can provide a short list (order of one hundred) of unambiguous examples (seeds). Of course the more examples provided, the better the results, but what we try to prove is that even with minimal knowledge good results can be achieved. Additionally some basic particularities of the language should be known: capitalization (if it exists and is relevant – some languages do not make use of capitalization; in others, such as German, the capitalization is not of great help), allowable word separators (if they exist), and a few frequent exceptions (like the pronoun "*I*" in English). Although such information can be utilised if present, it is not required, and no other assumptions are made in the general model.

1.1 Word-Internal and Contextual Information

The algorithm relies on both *word internal* and *contextual* clues as relatively independent evidence sources that drive the bootstrapping algorithm. The first category refers to the morphological structure of the word and makes use of the paradigm that for certain classes of entities some prefixes and suffixes are good indicators. For example, knowing that "*Maria*", "*Marinela*" and "*Maricica*" are feminine first names in Romanian, the same classification may be a good guess for "*Mariana*", based on common prefix. Suffixes are typically even more informative, for example "*-escu*" is an almost perfect indicator of a last name in Romanian, the same applies to "*-wski*" in Polish, "*-ovic*" and "*-ivic*" in Serbo-Croatian, "*-son*" in English etc. Such morphological information is automatically learned during bootstrapping.

Contextual patterns (e.g. "*Mr.*", "*in*" and "*mayor of*" in left context) are also clearly crucial to named entity identification and classification, especially for names that do not follow a typical morphological pattern for their word class, are of foreign origin or polysemous (for example, many places or institutions are named after persons, such as "*Washington*" or "*Madison*", or, in some cases, vice-versa: "*Ion Popescu Topolog*" is the name of a Romanian writer, who added to his name the name of the river "*Topolog*").

Clearly, in many cases, the context for only one occurrence of a new word and its morphological information is not enough to make a decision. But, as noted in Katz (1996), a newly introduced entity will be repeated, "if not for breaking the monotonous effect of pronoun use, then for emphasis and clarity". Moreover, he claims that the number of instances of the new entity is not associated with the document length but with the importance of the entity with regard to the subject/discourse. We will use this property in conjunction with the *one sense per discourse* tendency noted by Gale, Church and Yarowsky (1992b), who showed that words strongly tend to exhibit only one sense in a document/discourse. By gathering contextual information about the entity from each of its occurrences in the text and using morphological clues as well, we expect to classify entities more effectively than if they are considered in isolation, especially those that are very important with regard to the subject. When analyzing large texts, a segmentation phase should be considered, so that all the instances of a name in a segment have a high probability of belonging to the same class, and thus the contextual information for all instances within a segment can be used jointly when making a decision. Since the precision of the segmentation is not critical, a language independent segmentation system like the one presented by Amithay, Richmond and Smith (1997) is adequately reliable for this task.

1.2 Tokenized Text vs. Plain Text

There are two basic alternatives for handling a text. The first one is to tokenize it and classify the individual tokens or group of tokens. This alternative works for languages that use word separators (such as spaces or punctuation), where a relatively simple set of separator patterns can adequately tokenize the text. The second alternative is to classify entities simply with respect to a given starting and ending character position, without knowing the word boundaries, but just the probability (that can be learned automatically) of a boundary given the neighboring contexts. This second alternative works for languages like Chinese, where no separators between the words are typically used. Since for the first class of languages we can define *a priori* probabilities for boundaries that will match the actual separators, this second approach represents a generalization of the one using tokenized text.

However, the first method, in which the text is tokenized, presents the advantage that statistics for both tokens and types can be kept and, as the results show, the statistics for types seem to be more reliable than those for tokens. Using the second method, there is no single definition of "type", given that there are multiple possible boundaries for each token instance, but there are ways to gather statistics, such as considering what we may call "probable types" according to the boundary probabilities or keeping statistics on sistrings (semi-infinite strings). Some other advantages and disadvantages of the two methods will be discussed below.

2 The Basic Model

Before describing the algorithm, we will present a brief overview of some of its goals:

- a language independent core model
- the ability to exploit basic language-specific features
- the ability to learn from small named entity lists (on the order of 100 total training names)
- the capability to handle both large and small texts
- good class-scalability properties (the possibility of defining as many named entity types as desired, so that for different languages or different purposes the user can choose different classes of words to be recognized)
- the capability to store the information learned from each instance for further use

Three important concepts are used in our model:

2.1 Trie structures are used for both morphological and contextual information

Tries provide an effective, efficient and flexible data structure for storing both contextual and morphological patterns and statistics. First, they are very compact representations. Second, they support a natural hierarchical smoothing procedure for distributional class statistics. We consider character-based tries, in which each node contains a probability distribution (when working with tokenized text, two distributions are considered in each node, one for tokens and one for types). The distribution stored at each node contain the probability of each name class given the history ending at that node. Each distribution also has two standard classes, named "questionable" (unassigned probability mass in terms of entity classes, to be motivated below) and "non-entity".

To simplify the notations, we will refer to a start and end point bounded portion of text being analyzed (in order to determine if it represents a named entity or not) as a token.

Two tries are used for context (left and right) and two for internal morphological patterns of tokens.

Figure 1 shows an example of a morphological prefix trie, which stores the characters of tokens from

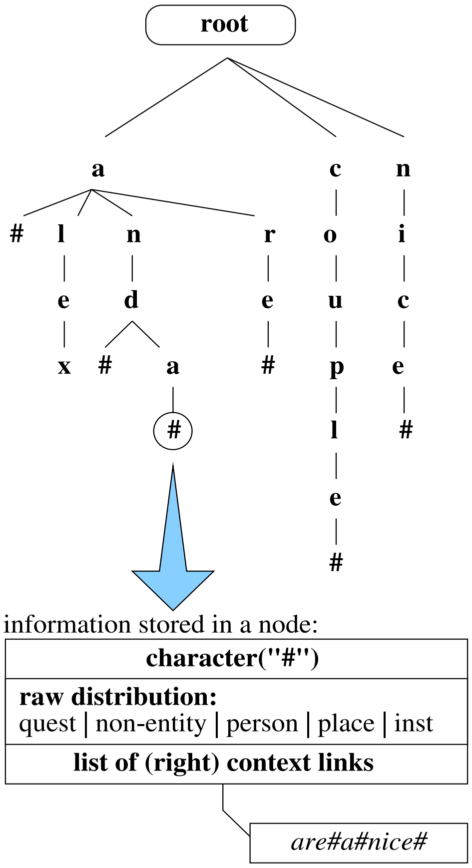

Figure 1: Morphological prefix trie for "Alex and Anda are a nice couple"

left to right from given starting points (with optional word boundaries indicated by "#").

Suffix tries (typically more informative) have equivalent structure but reversed direction. The left and right context tries have the same structure as well, but the list of links refers now to the tokens which have the particular context represented by the path from the root to the current node. For right context, the letters are introduced in the trie in normal order, for left context they are considered in the reversed order (in our example, "*Anda*" has as left context "dna#xela#"). Similarly, nodes of the context tries contain links to the tokens that occurred in the particular contexts defined by the paths. Two bipartite graph structures are created in this way by these links.

For reasons that will be explained later, raw counts are kept for the distributions.

The probability of a token/context as being in or indicating a class is computed along the whole path from the root to the terminal node of the token/context. In this way, effective smoothing is realized for rare tokens or contexts.

Considering a token/context formed from characters $l_1 l_2 ... l_n$, (i.e. the path in the trie is $root - l_1 - l_2 - ... - l_n$) the general smoothing model is:

$$\begin{aligned} P(class_j | l_1 l_2 ... l_n) &= \sum_{i=1}^{n} \lambda_i P(class_j | l_1 l_2 ... l_i), \\ \text{where } \lambda_i &\in [0,1] \text{ and } \sum_{i=1}^{n} \lambda_i = 1 \end{aligned}$$

It is reasonable to expect that smaller lambdas should correspond to smaller indices, or even that $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. In order to keep the number of parameters low, we used the following model:

$$\begin{aligned} F(class_j | l_1 l_2 ... l_n) &= \beta F(class_j | l_1) \\ &\quad + \sum_{i=2}^{n} \alpha^{n-i} F(class_j | l_1 l_2 ... l_i) \\ \text{where } \alpha, \beta &\in (0,1), \beta \text{ having a small value} \end{aligned}$$

The symbol F is used instead of P since we have raw distributions (frequencies) and a normalization step is needed to compute the final probability distribution.

A simpler model can use just one parameter (setting $\beta = \alpha^n$), but this has limited flexibility in optimizing the hierarchical inheritance - the probability of a class given the first letter is often not very informative for some languages (such as English or Romanian) or, by contrast, may be extremely important for others (e.g. Japanese).

2.2 EM-style bootstrapping

The basic concept of this bootstrapping procedure is to iteratively leverage relatively independent sources of information. Beginning with some seed names for each class, the algorithm learns contextual patterns that are indicative for those classes and then iteratively learns new class members and word-internal morphological clues. Through this cycle, probability distributions for class given token, prefix/suffix or context are incrementally refined. More details are given when describing stage 2 of the algorithm.

2.3 Unassigned probability mass as opposed to the classical maximum entropy principle

When faced with a highly skewed observed class distribution for which there is little confidence due to small sample size, a typical response to this uncertainty in statistical machine learning systems is to backoff or smooth to the more general class distribution, which is typically more uniform. Unfortunately, this representation is difficult to distinguish from a conditional distribution based on a very large sample (and hence estimated with confidence) that just happens to have a similar fairly uniform true distribution. One would like a representation that does not obscure this distinction, and represents the uncertainty of the distribution separately.

We resolve this problem while retaining a single probability distribution over classes by adding a separate "questionable" (or unassigned) cell that reflects the uncertainty of the distribution. Probability mass continues to be distributed among the remaining class cells proportional to the observed distribution in the data, but with a total sum (≤ 1) that reflects the confidence in the distribution and is equal to $1 - P(questionable)$.

This approach has the advantage of explicitly representing the uncertainty in a given class distribution, facilitating the further development of an interactive system, while retaining a single probability distribution that simplifies trie architecture and model combination. Incremental learning essentially becomes the process of gradually shifting probability mass from questionable/uncertain to one of the primary categories.

3 The Algorithm

The algorithm can be divided into five stages, which are summarized below.

Stage 0: build the initial training list of class representatives

Stage 1: read the text and build the left and right morphological and context tries

Stage 2: introduce the training information in the tries and re-estimate the distributions by bootstrapping

Stage 3: identify and classify the named entities in the text using competing classifiers

Stage 4: update the entity and context training space, using the new extracted information

Stage 0:

This stage is performed once for each language/task and consists of defining the classes and filling in the initial class seed data with examples provided by the user. The list of class training names should be as unambiguous as possible and (ideally) also relatively common. It is also necessary to have a relatively large unannotated text for bootstrapping the contextual models and classifying new named entities. Examples of such training seeds and text for Romanian language are given in Tables 1 and 2[1]. For the primary experiments reported in this paper, we have studied a relatively difficult 3-way named entity partition between First (given) names, Last (family) names and Place names. The first two tend to be relatively hard to distinguish in most languages. A simpler person/place-based distinction more comparable to the MUC-6 EMAMEX task is evaluated in Table 3(d).

Training Data (seed wordlists):		
F.NAME	L.NAME	PLACE
Andrei	Iliescu	Abrud
Adam	Popescu	Alba-Iulia
Alexandru	Ionescu	Arad
Aurel	Nitu	Bacău
Bogdan	Tănase	Botosani
Cosmin	Tudose	Bucuresti
Constantin	Rotariu	Brasov
Cătălin	Ciurea	Brăila
Costin	Bucur	Buzău
Claudiu	Gherman	Calafat

Table 1: Sample training wordlists for Romanian

Target Evaluation Text (labels not used for training)
Primarul comunei <place> Rosia Montană </place> judetul <place> Alba </place> <fname> David </fname> <lname> Botar </lname> a intrat în legendă datorită unor întîmplări de-a dreptul penibile, relatate în "Evenimentul zilei". Practic, primul gospodar al celei mai bogate comune în aur din <place> Muntii Apuseni </place> este mai tot timpul beat-crită, drept pentru care, la oficierea unei căsătorii, a sărutat mîna mirelui, a strîns mîna miresei si a întocmit certificat de deces în locul celui de căsătorie. Recent, <fname> Andrei </fname> <lname> Păunescu </lname> fiul poetului, a intentionat să achizitioneze gospodăria unei bucurestence care se stabilise de o vreme în <place> Rosia Montană </place> La primărie însă, turmentatul primar l-a trimis pe fiul lui <fname> Adrian </fname> <lname> Păunescu </lname> să-i cumpere ceva de băut, pentru a se putea concentra îndeajuns asupra hîrtiilor tranzactiei imobiliare. <fname> LUCIAN </fname> <lname> DOBRATER </lname>

Table 2: Sample test data for Romanian

Stage 1:

There are two ways to start this stage, either by tokenizing the text or considering it in raw form.

When tokenization is used, each token is inserted in the two morphological tries: one that keeps the letters of the tokens in the normal (prefix) order, another that keeps the letter in the reverse (suffix) order. For each letter on the path, the raw distributions are changed by adding the *a priori* probability

[1]The text refers to the mayor of a small town of Alba county, who was so drunk while officiating at a wedding that he shook the bride's hand and kissed the groom.

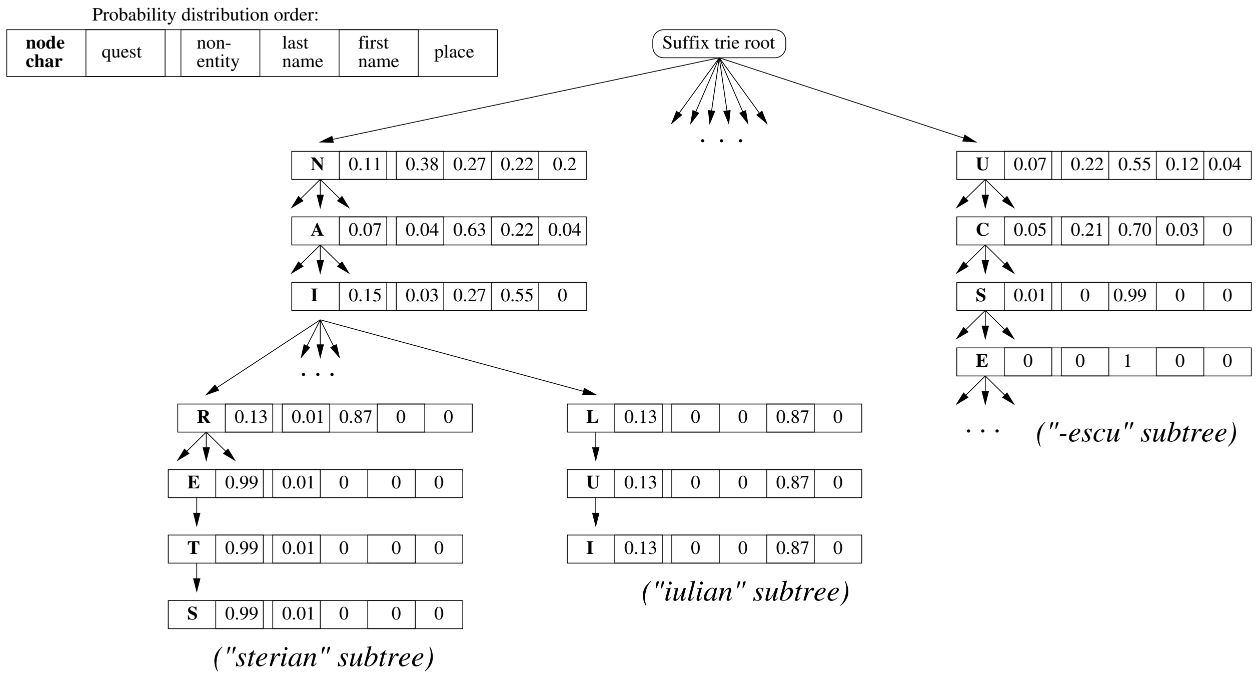

Figure 2: An example of normalized but unsmoothed distributions from the suffix morphological trie for Romanian. The paths shown are for *Iulian*, a "first name" entity, contained in the training word list; *Sterian* a "last name", not in the training data; and a partial path for the tokens ending in *-escu*.

of the token belonging to each class (language dependent information may be used here). For example, in the case of Indo-European languages, if the token starts with an upper-case letter, we add 1 full count (all probability mass) to the "questionable" sum, as this entity is initially fully ambiguous. If the token starts with lower-case (and hence is an unlikely name) in this case we add the bulk of the probability mass δ (e.g.$\delta \geqslant 0.9$) to "non-entity" and the remainder (1-δ) to "questionable" (otherwise unassigned). Other language-specific orthographic clues could potentially affect this initial probability mass assignment.

When no tokenization is applied, we have to consider possible starting and ending points. Therefore, the strings (which, for simplicity, we will refer as well as tokens) introduced in the prefix morphological trie and the ones introduced in the suffix trie may differ.

The left context of each token is introduced, letters in reverse order, in the left context trie, with pointers to the token in the morphlogical prefix trie; the right context of each token is introduced, in normal order, in the right context trie, keeping pointers to the token in the suffix trie. The distributions along the paths are modified according to the *a priori* distribution of the targeted token.

Stage 2:

This stage is the core bootstrapping phase of the algorithm. In essence, as contextual models become better estimated, they identify additional named entities with increasing confidence, allowing reestimation and improvement of the internal morphological models. The additional training data that this yields allows the contextual models to be augmented and reestimated, and the cycle continues until convergence. One approach to this bootstrapping process is to use a standard continuous EM (Expectation-Maximization) family of algorithms (Baum, 1972; Dempster et al., 1977). The proposed approach outlined below is a discrete variant that is much less computationally intensive, and has the advantage of distinguishing between unknown probability distributions and those which are simply evenly distributed. The approach is conservative in that it only utilizes the class estimations for newly classified data in the retraining process if the class probability passes a confidence threshold, as defined below. The concept of confidence threshold can be captured through the following definitions of dominant and semi-dominant.

Let us consider a discrete finite probability distribution $P = (p_1, ..., p_n)$. We say that P has a *dominant* if there is an i in $\{1...n\}$ such that $p_i > 0.5$, or in other words if

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} p_j < p_i.$$

We say that P has an α*-semi-dominant* with respect to an event k, where $\alpha > 1$, if it does not have k as dominant and there exist i in $\{1...n\}$ such that

$$\alpha \sum_{\substack{j=1 \\ j\neq i \\ j\neq k}}^{n} p_j < p_i.$$

A few comments about these definitions are necessary: it can be easily observed that not every distribution has a dominant, even though it has a maximum value. The second definition, of α-semi-dominant, makes sense if we consider a particular event k that is not relevant (or the result cannot be measured). By removing this event and normalizing the rest of the values, we obtain a new distribution (of size n-1) having an α-dominant.

The core of stage 2 is the bootstrapping procedure. The known names (either from the original training list or otherwise learned data) are inserted sequentially into the morphological tries, modifying the probability distributions of the nodes on the paths accordingly (the data structure is illustrated in Figures 1 and 2) . If the new distribution in one of the nodes on the path of a known token gains a dominant (for example "place") then the effect of this change is propagated by reestimating other node distributions given this change. Each distribution on the context paths in which that token occurred in the text is modified, by subtracting from the "questionable" mass a quantity proportional to the number of times the respective token was found in that context and adding it to the dominant-position (e.g. "place") mass. For the newly obtained distributions that gained a dominant (in our example "place") in the context trie, the bootstrapping procedure is called for all tokens that occurred in that context, and so on, recursively. Here it is very important that we consider raw distributions and not normalize them. For example, if word "*Mariana*" occurs x times with the right context "*merge*" (meaning "*goes*") and the distribution for "*mariana#*" has now been identified with the dominant "first name", then x units from the "questionable" mass can be moved to "first name" mass along the path of "*merge#*" in the right context trie. If semi-dominants are used instead of dominants then we have to account for the fact that the semi-dominants may change over time, so the probability mass must be moved either from "questionable" position or previous semi-dominant position, if a semi-dominant state has been reached before.

It may be easily observed that stage 2 has a sequential characteristic, because the updating is done after reading each name incrementally. When using dominants the order does not affect the process, because of the fact that once a dominant state is reached, it cannot change to another dominant state in the future (probability mass is moved only from "questionable"). In the case of semi-dominants, the data ordering in the training file does influence the learning procedure.

The more conservative strategy of using dominants rather then semi-dominants has, on the other hand, the disadvantage of cancelling or postponing the utilisation of many words. For example, if both "questionable" and "first name" have 49% of the mass then subsequent reestimation iterations are not initiated for this data, even though the alternative name classes are very unlikely.

Considering those advantages and disadvantages, we used the less conservative semi-dominant approach as the default model.

Stage 3:

In this stage the text is re-analysed sequentially, and for each token (given a start-end point pair) a decision is made. Here the bipartite structure of the two pairs of tries has a central role: during stage 2, the left context and prefix tries interact with each other and so do the right context and suffix tries, but there's no interference between the two pairs during the bootstrapping stage. Therefore, for each instance of a token in the text, four classifiers are available, a different one given by each trie. The decision with regard to the presence of an entity and its classification is made by combining them. Comparative trials indicate that higher performance is achieved by initially having the classifiers vote. Results indicate that the most accurate classifications are obtained from the two independently bootstrapped morphological tries (they incorporate the morphological information about the token to be classified, and, during the bootstrapping, they also incorporate information from all the contexts in which the token occurred). If the two agree (they have semi-dominants and they are the same) then the corresponding class is returned. Otherwise, agreement is tested between other paired independent classifiers (in order of empirically measured reliability). If no agreement is found, then a simple linear combination of all four is considered for the decision. This approach yields 6% higher F-measure than the simple interpolation of classifiers for the default parameters.

Stage 4:

The newly classified tokens and contexts are saved for future use as potential seed data in subsequent named-entity classification on new texts.

4 Results

The basic measures for evaluation of this work are precision and recall. Precision (P) represents the percentage of the entities that the system recognized which are actually correct. Recall (R) represents the percentage of the correct named entities in the text that the system identified. Both measures are incorporated in the F-measure, $F = 2PR/(P+R)$.

It would be inappropriate to compare the results of a language independent system with the ones designed for only one language. As Day and Palmer (1997) observed, "the fact that existing systems perform extremely well on mixed-case English newswire corpora is certainly related to the years of research and organized evaluations on this specific task in this language. It is not clear what resources are required to adapt systems to new languages."

It is important to mention that the F-measure for the human performance on this task is about 96%, (Sundheim 1995). Our experiments on Romanian text were consistent with this figure.

4.1 Baseline measures

In order to obtain a baseline performance for this method we considered the performance of a system that tags only the examples found in one of the the original training wordlists. We consider this to be a plausible lower bound measure if the training words have not been selected from the test text. Day and Palmer (1997) showed that a baseline F-measure score for the ENAMEX task varies from 21.2% for English to 73.2% for Chinese. It is important to mention that, when they computed these figures, they trained their language independent system on large annotated corpora (e.g. the Wall Street Journal for English).

The fact that the precision obtained by the baseline approach is not 100% indicates that the seed training names for each class are not completely unambiguous, and that a certain degree of ambiguity is generally unavoidable (in this case, mainly because of the interference between first names and last names).

Another significant performance measure is forced classification accuracy, where the entities have been previously identified in the text and the only task is selecting their name class. To obtain baseline performance for this measure, we considered a system that uses the original training word labels if there is an exact match, with all other entities labeled with a default "last name" tag, the most common class in all languages studied. The baseline accuracy was measured at 61.18% for Romanian. System accuracies range from 77.12% to 91.76% on this same data.

4.2 Evaluation of basic estimation methods

The results shown in Table 3 were obtained for a Romanian text having 12320 words, from which 438 were entities, using a training seed set of 300 names (115 first names, 125 last names, and 60 city/country names).

The baseline measures and default system (a) are as described above.

In configuration (b), the based parameters of the system have been optimized for Romanian, using greedy search on an independent development test (devtest) set, yielding a slight increase in F-measure.

Configuration (c) used the default parameters, but the more conservative "dominant" criterion was utilized, clearly favoring precision at the expense of recall.

Configuration (d), which is relevant for the ENAMEX task, represents the performance of the system when classes "first name" and "last name" are combined into "person" (whenever two or more such entities are adjacent, we consider the whole group as a "person" entity).

Configuration (e) shows contrastive performance when using standard continuous EM smoothing on the same data and data structures.

4.3 Evaluation by language and knowledge source

Table 4 shows system performance for 5 fairly diverse languages: Romanian, English, Greek, Turkish and Hindi. The initial 4 rows provide some basic details on the training data available for each language. Note that when annotators were generating the lists of 150-300 seed words, they had access to a development test from which to extract samples, but they were not constrained to this text and could add additional ones from memory. Furthermore, it was quite unpredictable how many contexts would actually be found for a given word in the development texts, as some appeared several times and many did not appear at all. Thus the total number of contextual matches for the seed words was quite variable, from 113-249, and difficult to control. It is also the case that not all additional contexts bring comparable new benefit, as many secondary instances of the same word in a given related document collection tend to have similar or identical surrounding contexts to the first instance (e.g. "*Mayor of* XXX" or "XXX *said*"), so in general it is quite difficult to control the actual training information content just by the number of raw seed word types that are annotated.

For each of these languages, 5 levels of information sources are evaluated. The baseline case is as previously described for Table 3. The context-only case restricts system training to the two (left and right) contextual tries, ignoring the prefix/suffix morphological information. The morphology only case, in contrast, restricts the system to only the two (prefix and suffix) morphological models. These can be estimated from the 3 training wordlists (150-300 words total), but without an independent source of information (e.g. context) via which bootstrapping can iterate, there is no available path by which these

	Romanian			
	Precision	**Recall**	**F-measure**	**Accuracy**
Baseline:	98.67	34.02	50.58	61.18
System Performance using:				
(a) default settings (using semi-dominants)	**76.95**	**64.99**	**70.47**	**78.49**
(b) re-estimated smoothing parameters	80.17	62.93	70.51	78.93
(c) learning with dominants	91.06	51.38	65.69	78.26
(d) ENAMEX-like "Person" / "Place" classes	**82.38**	**69.57**	**75.43**	**91.76**
(e) continuous EM approach	74.02	60.64	66.67	77.12

Table 3: Comparison of the performance of basic estimation methods on Romaninan

Language	**Romanian**		**English**		**Greek**		**Turkish**		**Hindi**	
Training text size	12320		15738		10445		5207		18806	
Total training seed words	300		190		210		150		150	
Contextual matches for seeds	149		205		113		133		249	
Labeled entities for testing	438		204		210		203		303	
Baseline:	98.67	34.01	83.33	12.25	100	18.09	86.66	19.21	94.42	20.26
(Precision/Recall//F-measure)	50.58		21.36		30.64		31.45		33.23	
Context Only:	96.59	19.45	82.35	6.86	87.50	3.33	53.33	11.82	84.21	9.58
(Precision/Recall//F-measure)	32.38		12.67		6.42		19.35		17.20	
Morphology Only:	73.79	52.97	84.42	31.86	78.35	36.19	75.32	28.57	89.01	24.25
(Precision/Recall//F-measure)	61.67		46.26		49.51		41.43		38.12	
Context and Morphology:	78.38	53.09	82.95	35.78	77.06	40.00	60.99	42.36	81.37	24.85
(Precision/Recall//F-measure)	63.30		50.00		52.66		50.00		38.07	
Full bootstrapping:	76.95	64.99	83.67	40.20	76.47	43.33	60.38	47.29	83.04	27.84
(Precision/Recall//F-measure)	70.47		54.30		55.32		53.04		41.70	
Baseline classification accuracy	61.18		62.74		55.23		61.08		61.79	
System classification accuracy	78.93		78.43		79.05		72.91		79.04	

Table 4: Comparison of performance by language and knowledge source

models can learn the behaviour of previously unseen affixes and conquer new territory. Thus the model is entirely static on just the initial training data. For the same reasons, the context only model is also static. In this case there is a possible bootstrapping path using alternating left and right context to expand coverage to new contexts, but this tends to be not robust and was not pursued. Interestingly, recall for morphology only is typically much higher than in the context only case. The reason for this is that the morphology models are full hierarchically smoothed character tries rather than word token tries, and hence have much denser initial statistics for small training data sets, proving greater partial matching potential for previously unseen words.

In an effort to test the contribution of the full iterative boostrapping, the "context and morphology only" results are based on the combination of all 4 tries, but without any bootstrapping. Thus they are trained exclusively on the 150-300 training examples. Performance for the combined sources is in all cases greater than for the morphology or context source used alone. Furthermore, the full iterative bootstrapping clearly yields substantial improvement over the static models, almost exclusively in the form of increased recall (and its corresponding boost the the F-measure).

Cross-language analysis yields further insight. First, recall is much higher for the 4 languages in which case is explicitly marked and is a clue for named entity identification (Romanian, English, Greek and Turkish) than for a language like Hindi, where there are no case distinctions and hence any word could potentially be a named entity. A language such as German would be roughly in the middle, where lower-case words have low probability as named entities, but capitalized words are highly ambiguous between common and proper nouns. Because approximately 96% of words in the Hindi text are not named entities, without additional orthographic clues the prior probability for "non-entity" is so strong that the morphological or contextual evi-

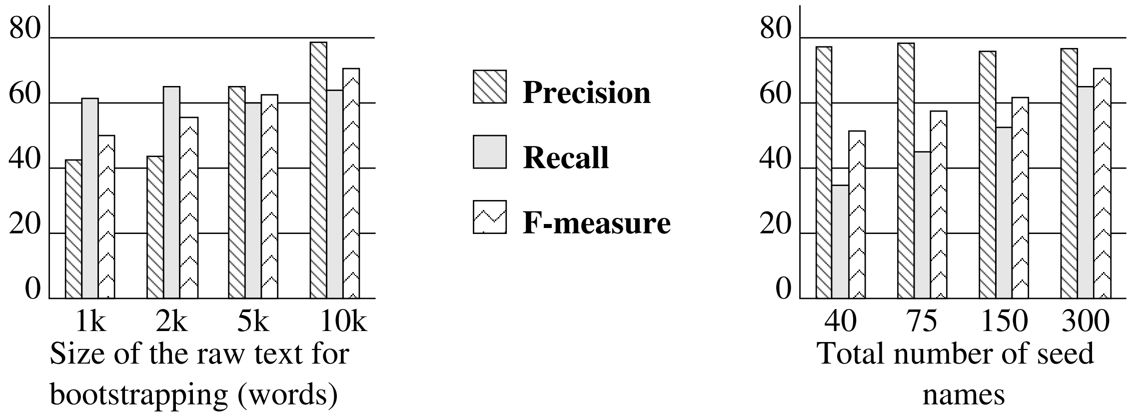

Figure 3: Learning curves for Romanian

dence in favor of one of the named entity classes must be very compelling to overcome this bias. With only 50 training words per context this is difficult, and in the face of such strong odds against any of the named entity classes the conservative nature of the learning algorithm only braves an entity label (correctly) for 38% more words than the baseline model. In contrast, its performance on entity *classification* rather than identification, measured by forced choice accuracy in labelling the given entities, is comparable to all the other languages, with 79% accuracy relative to the 62% baseline.[2]

4.4 Evaluation at different training set sizes

Figure 3 demonstrates that the performance of the algorithm is highly sensitive to the size of the training data. Based on Romanian, the first graph shows that as the size of the raw text for bootstrapping increases, F-measure performance increases roughly logrithmically, due almost exclusively to increases in precision. (Approximately the same number of unique entities are being identified, but due to the increased number of examples of each, their classification is more accurate). This is a very encouraging trend, as the web and other online sources provides virtually unlimited raw text in most major languages, and substantial on-line text for virtually all languages. So extrapolating far beyond the 10K word level is relatively low cost and very feasible.

The second graph shows that F-measure performance also increases roughly logrithmically with the total length of the seed wordlists in the range 40-300. This increase is due entirely to improved recall, which doubles over this small range. This trend suggests that there is considerable benefit to be gained by additional human annotation, or seed wordlist acquisition from existing online lexicons. However, relative to case of raw text acquisition, such additional annotations tend to be much costlier, and there is a clear cost-benefit tradeoff to further investment in annotation.

In summary, however, these evaluation results are satisfying in that they (a) show clear and consistent trends across several diverse languages, (b) show clear trends for improvement as training resources grow, and (c) show that comparable (and robust) classification results can be achieved on this diversity of languages.

5 Future work

For future work, natural next steps include incorporating a language independent word segmentation phase like the one proposed by Amitay, Richmond and Smith (1997), to improve the performance on large texts. Different statistics can be pre-computed for different languages and language families and stored in external files. For example, the *a priori* probability of a named entity given the set of characteristics of its representation in the text, such as position, capitalization, and relative position of other entities (e.g.: first name followed by last name). A further step is the implementation of a supervised active learning system based on the present algorithm, in which the most relevant words for future disambiguation is presented to the user to be classified and the feedback used for bootstrapping. The selection of candidate examples for tagging would be based on both the unassigned probability mass and the frequency of occurrence. *Active learning* strategies (Lewis and Gale, 1994) are a natural path for efficiently selecting contexts for human annotation.

[2]Note again that this baseline is more competitive than typical, as it not only assigns the majority tag ("last name"), but when there is an exact match with the training wordlist (e.g. "deepak"), a common occurrence given repeated high-frequency names in the Hindi data, the training classification is used as the baseline answer

6 Conclusion

This paper has presented an algorithm for the minimally supervised learning of named entity recognizers given short name lists as seed data (typically 40-100 example words per entity class). The algorithm uses hierarchically smoothed trie structures for modeling morphological and contextual probabilities effectively in a language independent framework, overcoming the need for fixed token boundaries or history lengths. The combination of relatively independent morphological and contextual evidence sources in an iterative bootstrapping framework converges upon a successful named entity recognizer, achieving a competitive 70.5%-75.4% F-measure (measuring both named entity identification and classification) when applied to Romanian text. Fixed k-way classification accuracy on given entities ranges between 73%-79% on 5 diverse languages for a difficult firstname/lastname/place partition, and approaches 92% accuracy for the simpler person/place discrimination. These results were achieved using only unannotated training texts, with absolutely no required language-specific information, tokenizers or other tools, and requiring no more than 15 minutes total human effort in training (for short wordlist creation) The observed robust and consistent performance and very rapid, low cost rampup across 5 quite different languages shows the potential for further successful and diverse applications of this work to new languages and domains.

7 Acknowledgements

The authors would like to thank Eric Brill, Radu Florian, Shankar Kumar, Murat Saraclar, Dimitra Vergyri and Jun Wu for both their feedback on this work and their help in annotating the named-entity data for the languages studied.

References

Amithay, E., K. Richmond, and A. Smith. 1997. Detecting Subject Boundaries within Text: A Language Independent Statistical ApproachIn *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 47-54.

Aone, C., S. Benett, and C. Lovell, 1997. Learning to Tag Multilingual Texts Through Observation. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 109-116.

Baum, L. 1972. An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process. *Inequalities*, 3:1–8.

Choi, M., Y. Ravin, and N. Wacholder. 1997. Disambiguation of Proper Names in Text. In *Proceeding of the Fifth Conference on Applied Natural Language Processing*, pp. 202-208.

Day, D., and D. Palmer. 1997. A Statistical Profile of the Named Entity Task. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 190-193.

Dempster, A., N. Laird and D. Rubin. 1977. Maximum Likelihood from Incomplete Data Via the EM Algorithm. *Journal of the Royal Statistical Society*, 39:1–38.

Gale, W., K. Church and D. Yarowsky. 1992. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26(5):415-439.

Gale, W., K. Church, and D. Yarowsky. 1992b. One Sense per Discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, pp. 233-237.

Gale, W., K. Church and D. Yarowsky. 1995. Discrimination Decisions for 100,000 Dimensional Spaces. *Annals of Operation Research*, 55:323-344.

Katz, S.M. 1996. Distribution of Context Words and Phrases in Text and Language Modeling. *Natural Language Engineering* 2(1):15-59.

Lewis, D. and W. Gale. 1994. A Sequential Algorithm for Training Text Classifiers. In *Proceedings of SIGIR'94*, pp. 3-12, Dublin.

Sundheim, B.M. 1995. Overview of Results of the MUC6 Evaluation. *Proceedings of the Sixth Message Understanding Conference*, pp. 13-31.

Yarowsky, D. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189-196.