Entity Models: Construction and Applications

James Allan and Hema Raghavan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{allan,hema}@cs.umass.edu

ABSTRACT

We propose *entity language models*, a probabilistic representation of the language used to describe a named entity (person, organization, or location). The model is purely statistical and constructed from snippets of text surrounding mentions of an entity. We evaluate the effectiveness of entity models in three tasks: fact-based question answering, classification into pre-defined groups, and description of the relationship between two entities. The results on all tasks are promising.

1. INTRODUCTION

To find out who someone is, we ask friends, read books, search libraries, browse the Web, etc., looking for information that describes the person. The more information we have gathered, the better a picture we develop. We might find out the person's career, what they are known for, who they have associated with, when they lived, and so on. Our picture of a person's "meaning" is constructed from numerous passages of text.

Inspired by that idea, we propose *entity models*, models of people, places, and other entities, based on how they are described. Our model is completely unstructured and based only on the text in our corpus. We do not employ any deep natural language processing beyond simple techniques for locating likely names nor do we use a knowledge base to improve our representation.

Fundamentally, an entity model is a probabilistic unigram language model of the way that a name is discussed. We collect all references to a name and consider the text surrounding the mention. That data provides us with an estimate of the likelihood that a word will be used in the context of a person. Our hypothesis is that the high probability words will provide a useful representation of who a person is.

We explore our hypothesis primarily on entity models constructed for people, and to a limited degree for locations. We consider three tasks to see whether these models are appropriate:

1. We find that the effectiveness of our model when used to find answers to questions where the answer is known to be a person's name or a location is comparable with state-of-the-art question answering systems.
2. We consider how well we are able to group people into classes. For example, determining whether someone is a politician, a movie star, a terrorist, etc. We find that our approach provides high accuracy in comparison to other state-of-the-art techniques.
3. We explore the ability of these models to provide meaningful descriptions of how—and if—two people are related. We present an evaluation that suggests that we can determine with high accuracy whether there is a relationship between people, and that the "descriptions" we generate are generally good.

Our modeling approach provides an interesting new way to represent a person (or other entity), and it has broad applicability. Because it uses the text directly, without deep processing, we expect that it can be ported to new domains (i.e., not just news) with little difficulty. We demonstrate that the models are successful at addressing a few tasks, suggesting that they may help in an even broader set of activities.

In the next section we formally describe entity models and overview related work toward modeling entities. In Section 3, we describe a set of experiments that help us determine the parameter settings for building good entity models. Section 4 discusses the application of entity models to question answering, Section 5 presents our work on classifying entities, and Section 6 describes our work on linking entities and describing their relationship. We conclude and outline future directions for the work in Section 7.

2. ENTITY LANGUAGE MODELS

We define an entity language model (ELM) to be a probability distribution of words that are likely to be used to describe the named entity. For example, an entity model for *George W. Bush* would have *president*, *republican*, *conservative*, and other such words with high probability. It would also include names of strongly associated people (e.g., *Dick Cheney*), places (*Texas*), actions (*cut taxes*), and so on.

Given a large corpus of text, we construct a model for a named entity E as follows. First we find all occurrences of E in the corpus. We use a named entity extraction system to locate the entities and to provide an entity type (e.g., person, location, organization). If a name occurs as more than one type, we treat each type separately—e.g., *Ford* as an organization (company) and *Ford* as a person (Henry Ford).

The model is then computed from a maximum of m occurrences of E in the text and a window of $\pm n$ words surrounding each mention. We call these $(2n + 1)$ word windows *snippets*.

If we pool these m snippets into a combined "bag of words," we can calculate a maximum likelihood estimate for any word that

appears around mentions of E.

We remove stop words from the model and use Jelinek-Mercer smoothing with a background model constructed from the entire corpus of documents:

$$P(w|E) = \lambda P_{ml}(w|E) + (1 - \lambda)P_{ml}(w|C) \quad (1)$$

where P_{ml} represents a maximum likelihood estimator (the proportion of word occurrences in the "bag of words" E or collection C that are w). Setting $\lambda = 0.6$ was determined empirically to work well for all tasks described here. Throughout the rest of this study, $P(w)$ or $P(w|E)$ refers to a smoothed probability distribution as in Equation 1. Whenever the maximum likelihood estimate is used the notation P_{ml} will be used.

2.1 Related work

It is common in Web retrieval to represent a Web page by the way that it is described in other pages—that is, by the anchor text associated with links to the page [2]. Such approaches typically use a vector space model constructed from the combined anchor text.

Conrad and Utt [8] used fixed-sized windows around a name to build a pseudo-document of text that represents the name. They used these pseudo-documents as a method to retrieve names in response to queries and as a way to find connections between names. The pseudo-documents form a type of model that is very similar to the entity models that we develop here, and which clearly demonstrated some potential value in the approach. Our work provides a formal framework for the idea, extends the number of uses to which the models can be put, and introduces more rigorous evaluations of the tasks.

Researchers have studied lexical, phrasal, co-occurrence and dependency relations that can be pulled out from spans of text [11, 5]. Pseudo-relevance feedback methods find words that are related to the query terms to improve the effectiveness of retrieval [24, 19, 14] . Those uses of statistics and others like them are similar to our building of ELMs in that they find related words.

We know of no work that viewed the probability distribution of text around a word as a model of its meaning.

3. MODEL QUALITY

In the above discussion we said that an ELM is built from m occurrences or mentions of an entity, using text snippets of size $(2n + 1)$ around the mention of the entity. We now consider ways to determine suitable values of n and m so as to obtain a good model.

There are a number of ways to capture the information content of a model [21]: entropy, perplexity and clarity are a few that have been used in the past. We chose clarity because it has been shown to be related to effectiveness of information retrieval tasks [9]. Clarity is the Kullback-Leibler divergence of a model with respect to the corpus. If the distribution of words in a model is identical to that of the entire corpus, it will show very low clarity. Models that are highly focused will have very high scores. Clarity is defined as:

$$\text{clarity} = \sum_{w \in V} P(w|E) log_2 \frac{P(w|E)}{P_{ml}(w|C)} \quad (2)$$

where V is the vocabulary of the corpus C.

We investigated the relationship between clarity scores and the parameters m and n as follows. We ran BBN's IdentiFinder™[3] on the TREC-8 corpus from TREC volume 4 & 5 to find names of people, organizations, and locations. That collection of 525,000 documents was gathered from the Foreign Broadcast Information Service (1996), Federal Register (1994), Financial Times (1992-1994) and Los Angeles Times (1989-1990). IdentiFinder extracted 1,691,654 unique named entities from 14,688,360 occurrences.

	Values of n		
Entity	12	25	50
Marilyn Monroe	1.92	1.46	1.01
Martina Navratilova	2.25	1.68	1.50
Magic Johnson	1.68	1.39	1.11
Dick Cheney	2.09	1.28	0.86
Bill Clinton	1.49	1.43	0.62

Figure 1: Clarity of five different models where snippets include n words to either size of the entity.

Name	No. of variants	Clarity
Alice	3,844	0.85
Betty	506	0.79
Janet	518	0.44
Janet Jackson		2.32
Janet Weiss		2.69
Janet Reno		0.90

Figure 2: Clarity scores for some common first names and the much higher scores for *Janet* when combined with surnames.

We studied how the clarity of the models changed with different parameter values. Figure 1 shows how the size of the snippet impacts clarity for a sample of 5 models (in all cases $m = \infty$, so all snippets were used). As n increases, the clarity of the model decreases, presumably because it is more likely to include extraneous information from the surrounding text. When n is held constant at 12 and the value of m is varied from 50 to 1000, we found some decrease in the clarity scores.

Some entity names are not specific enough to convey any information. In other words, models for common names like *Janet*, *John*, and *Smith* do not represent a single entity and contain mostly noise. Of course, although *Janet* may be a noisy model, the name *Janet Jackson* represents a more specific entity. Noisy or non-specific models carry no useful information. Figure 2 lists the clarity scores for five such models and also shows what happens to the clarity of *Janet* when it is combined with a surname. Figure 2 also indicates the number of variants of a given first name, where a variant of E is any other entity that contains E as a substring.

An obvious remaining question is whether a high clarity score corresponds to an effective model. In the following sections we will see that optimal values of m and n can be selected based on the task's evaluation measure. We will find that the parameters suggested by the clarity measure will provide generally good performance, but are not always ideal.

4. QUESTION ANSWERING

We hypothesize that entity language models can be used to find the answer to questions that expect an entity as the answer. Given a question Q, we are looking for the entity E such that:

$$E = \text{argmax}_{E_i \in \text{entities}} P(Q|E_i)$$

That is, which of the entities is most likely to generate the text of the question.

Because we are only constructing models for some classes of named entities, we cannot use a complete question answering evaluation corpus. In the TREC-8 QA track [22] 48.5% of the questions

For each query Q
  Identify *AnswerType*
  Retrieve D, the top 50 docs returned by
    the document retrieval system
  Find set $A = \{a_i\}$ of all entities such that
      $a_i \in D$
      type of a_i is *AnswerType*
  Remove from A all noisy entities (no. of variants > 1000)
  For each $a_i \in A$
    Find $P(Q| < a_i, AnswerType >)$
  Rank the elements of A

Figure 3: Algorithm for finding candidate entities that might answer a question.

had named entities for answers (28% were people, 18.5% locations, 2% organizations) [1].

Most question answering techniques (e.g., [6]) start by finding complete documents or passages of documents that are likely to produce the query, $P(Q|D)$. They then sort through the high ranking passages to find entities that are plausible answers. In contrast, we build a model of every entity using *all* (up to m) of the passages describing the entity and then look to see which entities are most likely to be a plausible answer.

4.1 Approach

Theoretically, our goal is to estimate $P(Q|E_i)$ for all entity models in the system so that we can rank them. If our system is accurate, the correct answer will always appear with the highest probability. Of course, ranking all 1.6 million entities in the TREC-8 collection is computationally infeasible.

To reduce the search space, we use a document retrieval system and a question classification system. (The latter is used by virtually every question answering system to determine the likely answer type for a question.) We use the algorithm in Figure 3 to select and rank possible answers.

The final step of the algorithm requires ranking the candidate entities. The simplest approach would be to borrow directly from the query likelihood model to rank in the order of decreasing $P(Q|E_i)$. However we found that the rank of the document from which the entities were obtained can also contribute to the scores.

Therefore, the entities were also given scores equal to the minimum rank in the ordered set D in which the entity occured. This is called the *document score* of an entity. Naturally there are plenty of ties in this document scoring mechanism and therefore document scores are a necessary but not sufficient component of the total score.

We tried a range of combinations of the two scores and eventually settled on:

$$R_E = 0.9 \cdot R_{P()} + 0.1 * R_{ds}$$

where $R_{P()}$ is the rank of the entity when sorted by decreasing $P(Q|E)$ and R_{ds} is the document score of the entity.

4.2 Experimental setup

To evaluate the effectiveness of entity models for question answering, we turned to the TREC-8 question answering task. A system was required to present its top five candidate answers for each question. It was evaluated by the reciprocal rank of the correct answer, yielding a score of 1, 1/2, 1/3, 1/4, 1/5, or zero for questions that were not answered in the top 5. The mean reciprocal rank (MRR) was calculated by averaging over all queries.

We selected 30 of the evaluation questions: 25 questions where the answer was the name of a person and 5 where the answer was a location. Because we are only building models for entities, it did not make sense to consider other types of questions at this time.

To find candidate answer entities we used the algorithm of Figure 3. Rather than actually running a retrieval system, we used the top 50 documents provided by the TREC evaluation. We did not do any automatic question classification, instead relying on our knowledge of how the 30 questions were classified. Although this implies perfect question classification, we felt it was not unreasonable considering the types of questions that we are considering. We used IdentiFinder[3] to pull out all candidate entities of the right type.

When constructing entity models, we used *all* snippets that included the entity i.e., $m = \infty$. In question answering, the risk associated with using only a fraction of the mentions for the entity models is that the snippets that contain the answer might get left out of the model. We did not remove stop words.

To set the value of n, we used a set of five training questions. The MRR was found to be 0.38 and 0.55 at $n = 25$ and $n = 50$, respectively. Hence, n was chosen as 50.

System	MRR
Cymphony	0.78
ATT	0.67
SMU	0.65
ELM QA system	**0.52**
IBM	0.48
Xerox	0.37
median	0.28

Figure 4: MRR of the top 5 systems in the TREC-8 QA task on our 30 questions, sorted by descending MRR.

4.3 Evaluation

Our system found the correct answer in the top five for 23 of the 30 questions (thirteen at rank 1, three each at ranks 2, 3, and 5, and one at rank 4). This corresponds to a MRR of 0.52 (loosely speaking that means that on average we got the answer at rank of about 2).

We compared our system to the top five performing systems at TREC-8. The reciprocal ranks of each of these systems and for the median system for each question were obtained from the TREC proceedings [22]. Figure 4 tabulates the MRR of each of these systems for our test set of 30 questions. It also includes our system, which would have placed fourth in that year's evaluation (on those questions and with perfect classification).

In most cases where the answer was not found, it was because the answer was way below in the ranked list of answers. However in two cases, the answer was not found because of tagging errors. For Q64, *Who was the voice of Miss Piggy?* (*Frank Oz*), the error was caused by IdentiFinder's tagging *Oz* as a location, and the name *Frank* was not considered as a candidate answer because we filter out noisy entities. For Q73, *Where is the Taj Mahal?* (*Agra*), IdentiFinder tagged all instances of *Agra* as an organization, and our algorithm only considers terms tagged as locations as possible answers to this question. Both of these errors suggest that a better mechanism would be to consider all *AnswerTypes* in the algorithm of Figure 3, perhaps giving higher weight to the type identified by the classifier. We leave that effort for future work.

5. CATEGORIZATION

Analogous to document categorization (classification), which is the automated assignment of documents to certain predefined categories (classes), we consider entity categorization. We believe that just like document categorization, entity categorization has applications in retrieval, organization, and browsing. This section strives to determine whether or not we can do a good job of classifying people into different categories like *movie stars*, *sports figures*, etc., by means of our entity models.

We approach this task in four different ways. The first is a language modeling variant of a K-nearest neighbors (KNN) algorithm. The second, which we call the class models approach, is a variant of centroid based clustering [12]. In addition to the above two approaches, we also used the Rainbow toolkit [16] for categorization using naive Bayes [15] and maximum entropy [18] as baselines. These have been shown to have good performance.

5.1 Approach

The 5 Nearest Neighbors (5NN) approach is as follows. The entities in a training set are initially assigned to their classes on the basis of human judgments. An entity E to be classified is compared to each of the elements of the training set to obtain its 5 nearest neighbors. The confidence of a class i is determined by $\frac{C_i}{k}$ where $k = 5$ (for 5NN) and C_i is the number of entities among the k nearest neighbors that belongs to class i. E is put into the class with highest confidence. Ties are broken arbitrarily.

In our second approach, the class models approach, given training entities E_1 to E_T for a given category C, the combined class model for C is defined as:

$$P^C_{ml}(w) = \frac{\sum_{i=1}^{T} P^i_{ml}(w)}{\sum_{w' \in V} \sum_{i=1}^{T} P^i_{ml}(w')} \quad (3)$$

where $P^C_{ml}(w)$ is the maximum likelihood probability of word w in the model for class C, and $P^i_{ml}(w)$ is the maximum likelihood probability of a word w in the model for entity E_i. This maximum likelihood model is smoothed using Jelinek Mercer smoothing (Equation 1). Class models are built for each of the predefined classes. Given an entity E' to be categorized, the similarity of its distribution to that of each of the classes is computed. The entity is put into the class it is most similar to.

Both approaches involve computing the similarity of two probability distributions. A number of similarity measures exist for this purpose. A popular measure is the KL-divergence (which we negate to form a similarity):

$$D(p||q) = \sum_{w \in V} p(w) log \frac{p(w)}{q(w)} \quad (4)$$

It measures the average inefficiency of using one distribution to encode another. KL divergence can be used for the class based approach. This measure does not allow for zero probabilities and hence smoothed probabilities have to be used.

For the KNN approach we needed a symmetric similarity measure. Variants on KL divergence (e.g., Jensen Shannon) did not perform well, so we instead considered measures that treat two distributions as vectors and apply geometrically motivated functions such as Euclidean distance, cosine or L_1. The L_1 metric is given as,

$$L_1(p, q) = \sum_{w \in V} |q(w) - p(w)| \quad (5)$$

The L_1 distance metric is symmetric and is applicable to both algorithms (when converted to a similarity). Whenever the L_1 metric was used the maximum likelihood estimates were used. Experiments suggested that L_1 performed well in measuring the distance between two entities.

In addition, naive Bayes and maximum entropy classification were used as for document classification. All the snippets corresponding to a given entity were collapsed into a single document. In this way there was one document for each entity containing all the snippets that contribute to its entity language model. Then the Rainbow toolkit was used just as in a document classification task.

Stop words were discarded from the snippets when building the entity models.

5.2 Evaluation model

We used the TREC-8 corpus just as in the question answering task. The following categories were chosen by inspecting a subset of the list of entities in the corpus. A total of 162 entities were obtained by searching the Internet and scouring the corpus. The categories are as follows:

Politics	Political figures (48 entities)
Pop	Pop or rock music stars (12 entities)
Composers	Classical music composers (13 entities)
Actors	Movie actors (37 entities)
Sports	Tennis and basketball stars (52 entities)

The data was first split into two sets

- *Train*, a training set (55 entities). The (arbitrarily chosen) training entities consisted of 15 entities each from the Sports, Politics, and Actors categories, 4 from the Pop, and 6 from the Composers category. Wherever initial labeled data for training was needed, it was obtained from this set.

- *Test*, a test set (107 entities). All testing was done using this group of entities.

This is not a large set of classes or instances, so results should be taken with a grain of salt. Our intent though is to show that entity models have some value for classification and this data set is sufficient to support that belief.

5.3 Experiments

For each approach we ran 10 trials, varying the training data for each trial. Although the *Train* dataset includes 55 entities, we randomly selected 40 for each trial, the only requirement being that each of the trials had the same number of training instances per category. We then classified the 107 entities in *Test* into one of the five categories and calculated accuracy. The final accuracy for a classifier was the average over the 10 trials.

We calculated results for the two baselines, for 5NN with the L_1 distance measure, and for the class models approach using both the L_1 and KLD distance measures. In all cases, we set the snippet window to include 25 words ($n = 12$). We set m, the number of snippets incorporated into the model, to either 300 or ∞. Figure 5 shows the results.

To get a sense of how the algorithms confused the classes, we ran another experiment where each algorithm was trained using all 55 entities (rather than only 40) in *Train*, setting $m = 300$ and $n = 12$. Figure 6 shows the confusion matrix for the class models approach with the L_1 distance measure. The figure suggests that it is easy to mistake several types of entities as pop stars: all pop stars were classified correctly, but 4 additional entities were assigned to that class. The general trends were similar across all classifiers, though the specific classes that were troublesome shifted: naive

Algorithm	Similarity	m,n	Sports	Actors	Politics	Comp.	Pop	Average
5NN	L_1	∞,12	95 ± 3	81 ± 5	88 ± 1	**100 ± 0**	97 ± 4	90 ± 2
		300,12	96 ± 2	95 ± 0	81 ± 2	78 ± 5	**100 ± 0**	90 + 1
Class Models	KLD	∞,12	75 ± 3	40 ± 6	77 ± 1	57 ± 0	30 ± 5	65 ± 2 *
		300,12	82 ± 1	61 ± 5	87 ± 1	63 ± 0	38 ± 0	76 ± 2 *
	L_1	∞,12	91 ± 2	90 ± 4	89 ± 2	**100 ± 0**	**100 ± 0**	91 ± 1
		300,12	94 ± 0	**99 ± 2**	81 ± 1	86 ± 0	**100 ± 0**	91 ± 0
Naive Bayes	NA	∞,12	94 ± 3	82 ± 5	**92 ± 1**	86 ± 0	79 ± 6	89 ± 1
		300,12	**97 ± 3**	87 ± 5	92 ± 3	86 ± 0	86 ± 7	**92 ± 2**
Max Ent	NA	∞,12	84 ± 3	97 ± 2	88 ± 2	77 ± 7	56 ± 10	86 ± 2 *
		300,12	86 ± 3	96 ± 1	91 ± 1	81 ± 5	56 ± 8	87 ± 2

Figure 5: Percent accuracy for each of the classifiers. Boldface indicates the highest in the category. A two-tailed t-test was performed to compare the average accuracy of each classifier (last column) to Naive Bayes with $m = 300$,$n = 12$. Statistically significant differences (at $P < 0.05$) are indicated with an asterisk.

	Actual class					
	Spo.	Pol.	Act.	Comp.	Pop	(FA%)
Spo.	34	4	0	0	0	10.5
Pol.	1	28	0	1	0	6.6
Act.	0	0	21	0	0	0.0
Comp.	0	0	0	6	0	0.0
Pop	2	1	1	0	8	36.4
Accuracy(%)	91.8	84.8	95.4	85.7	100	

Figure 6: Confusion matrix for the class models based approach using 55 training and 107 test entities. $m = 300$, $n = 12$, and the distance metric was L_1. Average accuracy is 90.65%. Columns represent the actual class and rows represent assigned class (e.g., there were 33 politician entities that were assigned into three different classes).

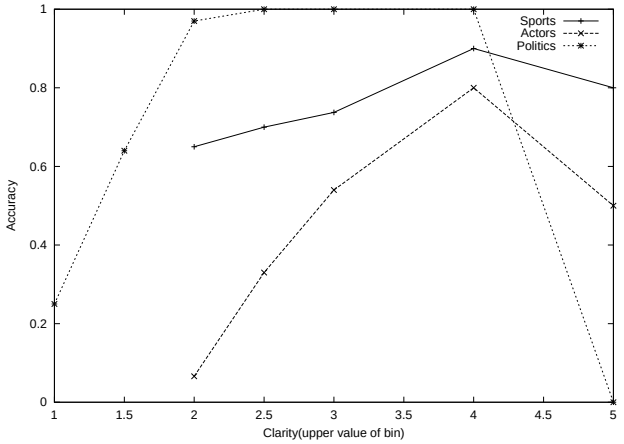

Figure 7: Relation of classification accuracy to clarity scores for the class models based approach.

Bayes and maximum entropy tended to incorrectly place entities into the actor class.

In Section 3 we showed a relationship between the clarity score of an entity and m, the number of snippets used to build its model. We posited that high clarity would lead to better accuracy in tasks using entity models. To explore that issue, we used the classification task to see how entity clarity relates to entity classification accuracy.

For each of the 107 entities in the test set, we calculated their accuracy (averaged over 10 trials) when used in the class model approach with KL distance. We then grouped the entities into bins based on their clarity scores. Figure 7 shows the relationship between clarity (on the X axis) and accuracy of entities in three categories with that clarity (Y axis). The other two categories (Composers and Pop-stars) showed similar trends but the number of test entities in these cases were too few to establish conclusive results and therefore have not been shown.

Classification accuracy peaks in the 3.5-4 range, indicating that high clarity yields better performance (on average). Interestingly, entities that are *too* clear do as poorly as entities that are unfocused. This situation arises when there are very few mentions of an entity, so the model is built from few snippets and is quite sparse. The sparseness means that it is quite different from a broad background model (so clarity is high), but also means that it is not sufficiently general to classify well (low accuracy). Low clarity scores mean that the entity's model is so broad (similar to the background) that it could encompass anything, so it does not lend itself to accurate classification.

5.4 Discussion

A column-wise look at Figure 5 indicates that on average the class models based approach using KL divergence performs the worst. On average the highest accuracy is attained by the Naive Bayes classifier using $n = 12$ and $m = 300$. The L_1 measure on both the 5NN and class models approaches performs almost equally well. On a per-class basis one of 5NN or the class models approach usually performs better.

The class models approach using KLD, and maximum entropy with $n = 12$, $m = \infty$, performed significantly worse than naive Bayes (marked by * in the table). All other approaches are statistically comparable to the naive Bayes classifier.

For a fixed number of training instances, the data used for training causes some variation in accuracy. Figure 5 shows that Politics and Sports exhibit low variance, whereas Actors and Pop-stars exhibit, in general, lower accuracy, higher variance and a higher false alarm rate. This effect is because of the vocabulary of the snippets used to build the entity models. Politicians and sports stars appear in the news for typical reasons, whereas news articles about actors and pop music stars cover a multitude of subjects ranging from gossip to movie reviews. Hence the entity models for a single actor may contain a surprisingly rich vocabulary. That point explains their lower classification accuracy. The accuracy of the actors class model also depends on the training data. For example, if an actor

like *Elizabeth Taylor* is used to train the Actors model, the accuracy may drop because Taylor appears in the news quite often for reasons having little to do with acting.

6. ASSOCIATION DISCOVERY

In this section we explore the possibility of finding relationships between entities and descriptions of those relationships. For example, finding that *Navratilova* and *Graf* are connected because of their *tennis* activities.

We propose a method that does not rely on co-occurrence of names to extract relationships, but uses similarities in the way that they are described. We also do not restrict ourselves to finding associations within certain classes or groups. So although a tennis star is more likely to be associated with another tennis star, we hope to discover relations across groups like tennis stars and movie stars.

Our method can be extended to build associations of the form *Tennis* $\rightarrow$ *Graf*, that is, where the links are not just between entities but also include classes of entities (as in the previous section). Once this is obtained it can used to build a taxonomy of people, places and, organizations. This can then be used as a topic or concept hierarchy [20] or like the Yahoo hierarchies. This is thus a start to using language modeling to generate hierarchies and association graphs.

The problem of finding relationships between items is a typical text mining challenge [7, 10, 13], though co-occurrence techniques and/or careful natural language processing are more likely to be used in that arena. The work by Conrad and Utt [8] found connections between names by both co-occurrence (related names occurring the pseudo-document of a name) and by similarity (querying the pseudo-documents).

6.1 Relation discovery

Our goal is to discover relations between entities and to obtain a description of the relationship.

Rather than using the distance measure L_1 directly (shown in Section 5 to perform well for categorization), we define a symmetric similarity measure *overlap* between two entities as $overlap(p, q) = 1 - L_1(p, q)/2$. (This value is the extent to which the two probability distributions intersect.) The overlap measure ranges from 0 for no overlap to 1 for perfect overlap.

Does the overlap measure correspond to actual relationships? To find out, we compared seven entities to 107 others and ranked the 749 pairs in order of decreasing *overlap*. The list was given to each of two evaluators. They were instructed to indicate the strength of the association with a value from the set (0,0.25,0.5,0.75,1) with zero indicating *totally unrelated* and one indicating *completely related*. Associations in this case were not restricted to select categories like Sports or Movie stars, but could be much broader. The scores assigned by both evaluators for each relationship were averaged. This average was binned by overlap score and the average in each bin was computed. Figure 8 shows the very strong relationship between our calculated overlap score and the average relatedness score in each bin.

The overlap measure can also be used to build a network of related entities as shown in Figure 9. A set of 25 entities were compared against each other; any overlap value over 0.20 caused a link between the two entities. In the graph, the connected components are clearly related to each other.

6.2 Relation description

A relationship between two entities can be thought of as a distribution of words that might be used to describe the relationship. We want to obtain a language model of the relationship between

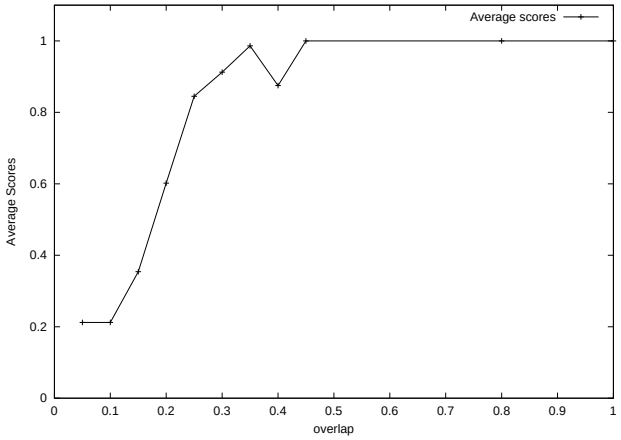

Figure 8: Relationship of evaluator assigned scores to a relationship with $overlap$.

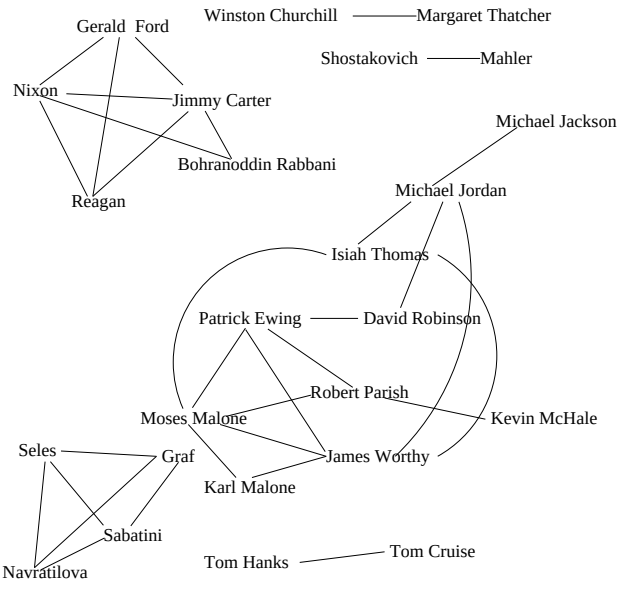

Figure 9: Network of entities, built with an overlap threshold of 0.20. The drawing was laid out by hand.

two entities, each of which is represented by its own model. Using that relationship model, we can find high probability terms that we hypothesize will describe the relationship meaningfully.

Using the overlap measure, a model for the distribution of the relation between entity E_1 and E_2 is obtained as follows:

$$P^R_{ml}(w) = \frac{\min(P^1_{ml}(w), P^2_{ml}(w))}{\sum_{w' \in V} min(P^1_{ml}(w'), P^2_{ml}(w'))} \quad (6)$$

$$P^R(w) = \lambda P^R_{ml}(w) + (1-\lambda) P_{coll}(w) \quad (7)$$

where $P^1_{ml}(w)$ and $P^2_{ml}(w)$ are the maximum likelihood estimates of the word w in the models for entities E_1 and E_2, respectively, and we set $\lambda = \textit{overlap}(P_1, P_2)$. Equation 6 computes a new distribution which captures the intersection of the maximum likelihood distributions of the models for E_1 and E_2, and then normalizes its size to 1.0. Equation 7 models the relationship using a mixture model of the distribution obtained in Equation 6 and the collection. The motivation behind setting λ equal to $\textit{overlap}(P_1, P_2)$ is that in the case where there is less overlap, more weight is given to the collection model. When $E_1 = E_2$ the relationship model is the maximum likelihood model of the entity.

Stop words were removed in the construction of the entity models for this task. Our goal is to find descriptive words, so we felt it was simpler to include only content words in the modeling.

Figure 10 illustrates some of the high probability words derived from three sample pairs of entities. Only terms with a probability over 0.01 are listed.

6.3 Evaluating Relation Descriptions

Generally it is difficult to evaluate the quality of automatically generated descriptions. We chose to determine the quality of our results by asking human annotators to judge what was done.

We generated a set of entity pairs by randomly selecting 25 entities and then comparing each of those to another 106 entities. We discarded any of the 2,650 pairs that had an overlap score below 0.20, resulting in only 69 pairs considered strongly enough related to be evaluated. For each such relationship, the top 10 terms with a value of $P^R(w) > 0.01$ were extracted. We called this set of terms a "relation description" for that pair.

We hired two undergraduates to evaluate the relationship descriptions. They were asked to evaluate each term in the relation description and indicate whether or not it was descriptive of the connection between the two. The evaluators made their judgments independently and then worked together to adjudicate the results. The evaluators were asked to use their own *a priori* knowledge of how two entities were related, not to research the issue. One evaluator thought 63.8% terms were relevant and the other thought 61.8% were. When they worked together they found 61.7% terms relevant.

To score a relationship, we averaged the scores of all of the terms in the description. We have three scores for each pair: one from each judge and the result of adjudication. Figure 11 shows a distribution of the relationship scores. It is clear that most of the relationships had high scores (0.5-0.8), suggesting again that the descriptive words are generally on target.

6.4 Discussion

Our goals in this section were two-fold: (1) to determine *if* two people are related, and if so, (2) to try to find the terms that *describe* how they are related.

Figure 9 gives an idea of the kind of relations our system extracted. It is easy to see that entities of a given category cluster together. For example, as illustrated in Figure 10, *Pete Sampras* shows most similarity to *Steffi Graf* (another tennis player), less

E_1	E_2	overlap score	Actual relation	description of relation	
				Term	$P(w)$
Pete Sampras	Stefi Graf	0.39	Tennis players	champion	0.026
				wimbledon	0.023
				match	0.020
				tennis	0.019
				open	0.013
Pete Sampras	Michael Jordan	0.17	Sports players	player	0.015
Pete Sampras	Gerry Rawlings	0.01	None	*no terms extracted*	

Figure 10: Example of the top few terms in some sample relation descriptions.

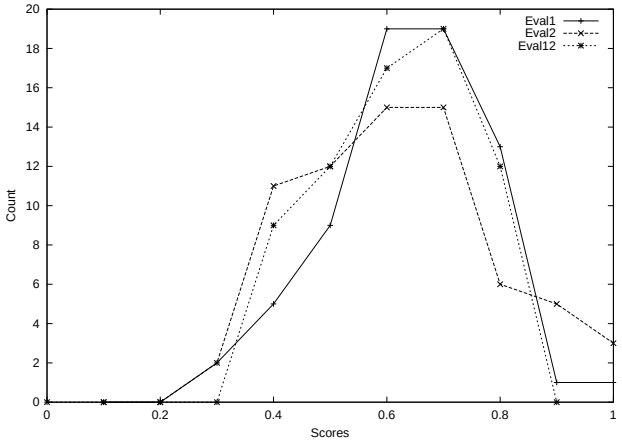

Figure 11: Score-wise distribution of *relation scores*. *Eval1* and *Eval2* denote the two evaluators; *Eval12* denotes the adjudicate result.

affinity to *Michael Jordan* (another sports person) and the least affinity to *Gerry Rawlings* (a politician).

We also tried to quantify the quality of the relationship model using human judgments. The evaluation of the relation scores gives an insight into the quality of terms generated for the relationship model. Figure 10 gives a peek into some of high probability terms of these relationship models.

An evaluator's idea of whether terms describe a relationship depended largely on his or her prior knowledge of vocabulary in a domain. For example, one relation where the evaluators had a significant difference of opinion (before adjudication) was that between *Andre Agassi* and *Steffi Graf*, both of whom are tennis players. One evaluator was aware of the details of tennis and marked domain specific terms like *seed* relevant. The other knew little of tennis, so felt that *seed* was a non-relevant term. There were also some differences of opinions such as one marking *dollars* as a relevant term to describe the relation between *Arnold Schwarzenegger* and *Madonna*, two successful Hollywood personalities.

We have shown a new method for modeling and describing relations. Evaluation is as subjective as anything else in information retrieval, but the trends seem clear.

7. CONCLUSION AND FUTURE WORK

We have defined entity language models, a probabilistic representation of how entities are described. The model depends only on the ability of finding references to a name and collection snippets of unstructured text around those references. No sophisticated natural language processing or understanding or knowledge databases beyond that are used. An entity is represented entirely by how it is described.

We have shown that entity models are rich enough that they can be used to find answers to fact-based questions that expect an entity as an answer. We compared our results on that type of questions to more elaborate systems used in the TREC-8 question answering evaluation and showed that this approach would have been competitive. An obvious next step is to try our technique on later TREC evaluation tasks. It may also prove interesting to model other entities such as dates, numbers, and so on. Although it is difficult to imagine a model for *53*, models of *1776* or *September 11, 2001* seem promising.

We also showed that entity models can be used to classify entities into already defined groups. Our experiments in this task used a small set of classes and so should be viewed cautiously. However, our results were competitive with existing classification approaches that are known to perform well. We have begun building a substantially larger set of entities and putting them into classes.

Finally, we showed that it is possible to used entity models to determine whether or not two entities are related and, if so, to describe their relationship with reasonable accuracy. We believe that we may be able to find better descriptions for at least the cases where the names are mentioned together somewhere in the text. For entities that are strongly related but that are not described together, we may not be able to do better than use a list of words or phrases.

We are pleased that entity models have worked so well on these preliminary tasks. We are exploring additional ways that entity models can be used, including connecting them into data mining systems, incorporating them into news tracking systems, and leveraging them for summarization of the personalities involved in a story.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, NSF grant #IIS-9907018, the Advanced Research and Development Activity under contract number MDA904-01-C-0984 and in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

8. REFERENCES

[1] S. Abney, M. Collins, and A. Singhal. Answer extraction. In *Proc. of the 6th ANLP Conference*, pages 296–301, 2000.

[2] E. Amitay. Using common hypertext links to identify the best phrasal description of target web documents. In *Proc. of the SIGIR'98 Post-Conference Workshop on Hypertext Information Retrieval for the Web*, 1998.

[3] D. M. Bikel, R. L. Schwartz, and R. M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231, 1999.

[4] R. Byrd and Y. Ravin. Identifying and extracting relations from text. In *NLDB*, 1999.

[5] K. Church, W.Gale, P.Hanks, and D.Hindle. Using statistics in lexical analysis. In *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pages 115–164, 1991.

[6] C. L. A. Clarke, G. V. Cormack, and T. R. Lynam. Exploiting redundancy in question answering. In *Research and Development in Information Retrieval*, pages 358–365, 2001.

[7] C. Clifton and R. Steinheiser. Data mining on text. In *Proc. of The 22nd Annual International Computer Software and Applications Conference*, 1998.

[8] J. G. Conrad and M. H. Utt. A system for discovering relationships by feature extraction from text databases. In *Proc. of the 17th ACM-SIGIR Conference*, pages 260–270, 1994.

[9] S. Cronen-Townsend and W. B. Croft. Quantifying query ambiguity. In *Proc. of HLT*, pages 94–98, 2002.

[10] J. Dorre, P. Gerstl, and R. Seiffert. Text mining: Finding nuggets in mountains of textual data. In *Proc. of the 5th ACM SIGKDD*, pages 398–401, 1999.

[11] S. Haas and R. Losee Jr. Looking in text windows: Their size and composition. *Information Processing and Management*, 50:619–629, 1994.

[12] E.-H. Han and G. Karypis. Centroid-based document classification: Analysis and experimental results. In *Principles of Data Mining and Knowledge Discovery*, pages 424–431, 2000.

[13] M. Hearst. Untangling text data mining. In *Proc. of ACL*, 1999.

[14] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Research and Development in Information Retrieval*, pages 120–127, 2001.

[15] D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98*, pages 4–15, 1998.

[16] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[17] U. Y. Nahm and R. J. Mooney. Mining soft-matching rules from textual data. In *IJCAI*, pages 979–986, 2001.

[18] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99*, pages 61–67, 1999.

[19] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-2. In *The Second Text retrieval Conference*, pages 21–34, 1994.

[20] M. Sanderson and W. B. Croft. Deriving concept hierarchies from text. In *Research and Development in Information Retrieval*, pages 206–213, 1999.

[21] C. Stanley, F. Douglas, and B. Ronald. Evaluation metrics for language models. In *DARPA Broadcast News Transcription and Understanding Workshop.*, 1998.

[22] In *The Eighth Text REtrieval Conference (TREC 8)*. NIST, 1999. NIST Special Publication 500-246.

[23] F. Xu, D. Kurz, J. Piskorski, and S. Schmeier. Term extraction and mining of term relations from unrestricted texts in the financial domain. In *Proc. of BIS*, 2002.

[24] J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, 18(1):79–112, 2000.