Learning Random Walk Models for Inducing Word Dependency Distributions

Kristina Toutanova KRISTINA@CS.STANFORD.EDU
Christopher D. Manning MANNING@CS.STANFORD.EDU
Andrew Y. Ng ANG@CS.STANFORD.EDU
Computer Science Department, Stanford University, Stanford, CA 94305, USA

Abstract

Many NLP tasks rely on accurately estimating word dependency probabilities $\mathrm{P}(w_1|w_2)$, where the words w_1 and w_2 have a particular relationship (such as verb-object). Because of the sparseness of counts of such dependencies, smoothing and the ability to use multiple sources of knowledge are important challenges. For example, if the probability $\mathrm{P}(N|V)$ of noun N being the subject of verb V is high, and V takes similar objects to V', and V' is synonymous to V'', then we want to conclude that $\mathrm{P}(N|V'')$ should also be reasonably high—even when those words did not cooccur in the training data.

To capture these higher order relationships, we propose a Markov chain model, whose stationary distribution is used to give word probability estimates. Unlike the manually defined random walks used in some link analysis algorithms, we show how to automatically learn a rich set of parameters for the Markov chain's transition probabilities. We apply this model to the task of prepositional phrase attachment, obtaining an accuracy of 87.54%.

1. Introduction

Word dependency or co-occurrence probabilities are needed in many natural language tasks. This includes lexicalized parsing, building language models, word sense disambiguation, and information retrieval. However, it is difficult to estimate these probabilities because of the extreme sparseness of data for individual words, and even more so for word pairs, triples, and so on. For instance, Bikel (2003) shows that the parser of Collins (1999) is able to use bi-lexical word dependency probabilities[1] to guide parsing decisions only 1.5% of the time; the rest of the time, it backs off to condition one word on just phrasal and part-of-speech categories. If a system could be built with reasonably accurate knowledge about dependency probabilities between all words, one would expect the performance gains on many tasks to be substantial.

Sophisticated back-off and interpolation methods have been developed for language modeling (Goodman, 2001). Dagan et al. (1999) showed that performance on zero-count events can be greatly improved if the model includes estimates based on distributional similarity. Other kinds of similarity among words have also been used to reduce sparseness. For instance, stemming words is a very traditional way of somewhat lessening sparseness, and resources like WordNet have been used in many natural language models.

All of these ways of using associations and similarities between words to predict the likelihood of unseen events have their advantages. Symbolic knowledge bases, such as WordNet, have the advantage of being based on abundant world knowledge and human intuition, but have the disadvantages of having incomplete coverage and being non-probabilistic. Using stemming or lemmatized words has been helpful for reducing sparseness in some problems, and slightly harmful in others (Hull, 1996).

Here, we propose a method for combining these information sources that induces a distribution over words by learning a Markov chain (random walk) model, where the states correspond to words, such that its stationary distribution is a good model for a specific word-distribution modeling task. The idea of constructing Markov chains whose stationary distributions are informative has been seen in several other applications, such as the Google PageRank algorithm (Brin & Page, 1998), some HITS (Kleinberg, 1998)-like link analysis algorithms (Ng et al., 2001),

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004. Copyright 2004 by the authors.

[1] Bi-lexical probabilities include two words, one in the conditioning context and one in the future, in addition to possibly other variables, for example, $P(salad|eat, V, VP)$.

and for query expansion in IR (Lafferty & Zhai, 2001). Our work is distinguished from these approaches in that rather than using a carefully hand-picked Markov chain, we will automatically *learn* the parameters for the random walk. This allows us to construct Markov chains with many more parameters, that are much richer in structure and of significantly greater complexity than seen in other applications. In doing so, we can also allow our model to learn to exploit diverse knowledge sources such as WordNet, morphology, and various features of words derived from dependency relations; all of these simply become additional "features" made available to the random walk learning algorithm. The proposed techniques are general and can be applied to other problem domains, such as the web, citation, and clickstream data.

In this paper, we choose deciding the attachment site of Prepositional Phrases (PPs) as a touchstone problem, and show how random walk methods can be applied to this problem. PP attachment decisions are a central component problem in parsing and one of the major sources of ambiguity in practice. For example, in the sentence: *He broke the window with a hammer*, the prepositional phrase *with a hammer* could either modify the verb *broke*, and thus mean that the *hammer* was the instrument of the breaking event, or it could modify the noun *window* and thus mean that the *window* perhaps had a stained glass rendition of a *hammer* in it. People immediately recognize the more plausible meaning using their world knowledge, but this knowledge is not readily available to parsers. Previous research has shown that by using statistics of lexical co-occurrences, much higher accuracy can be achieved in comparison to approaches that only look at structure (such as preferring attachment to a verb or the closer word, etc.) (Hindle & Rooth, 1993).

2. Preliminaries

We briefly review Markov chains (MC). For a more detailed treatment, see, e.g., (Brémaud, 1999).

A MC over a set of **states** $\mathcal{S}$ is specified by an **initial distribution** $p_0(S)$ over $\mathcal{S}$, and a set of **state transition probabilities** $p(S_t|S_{t-1})$. A Markov chain defines a distribution over sequences of states, via a generative process in which the initial state S_0 is first sampled from according to p_0, and then states S_t (for $t = 1, 2, \dots$) are sampled in order according to the transition probabilities. The stationary distribution of a MC is given by $\pi(s) = \lim_{t\to\infty} P(S_t = s)$, if the limit exists.

The MCs used in (Brin & Page, 1998; Ng et al., 2001) have the property that on each step, there is a probability $\gamma > 0$ of resetting according to the initial state distribution p_0. Thus, the state transition probabilities can be written

$$p(S_t|S_{t-1}) = \gamma p_0(S_t) + (1-\gamma)p'(S_t|S_{t-1}) \qquad (1)$$

for some appropriate p'. This ensures that the MC has a unique stationary distribution (Brémaud, 1999), and in practice also prevents the chain from getting stuck in small loops (Brin & Page, 1998).

Given a MC as described above, we can construct another MC $S'_0, S'_1, \dots$ with the initial state S'_0 distributed according to p_0, and state transitions given by the p' in Equation (1). It is straightforward to show that

$$\pi(s) = \gamma \sum\nolimits_{t=0}^{\infty} (1-\gamma)^t P(S'_t = s) \qquad (2)$$

where π here is the stationary distribution of the *original* MC $S_0, S_1, \dots$. Equation 2 can be used to efficiently compute π. Also, because terms corresponding to large t have very little weight $(1-\gamma)^t$, when computing π, this sequence may be truncated after the first few (on the order $1/\gamma$) terms without incurring significant error.

Equation (2) gives a useful alternative view of π. Consider a random process in which the state S_0 is initialized according to p_0. On each time step t, with probability γ we "stop" the chain and output the current state S_t; and with probability $1-\gamma$, we till take a state transition step and sample S_{t+1} according to the transition probabilities $p'(S_{t+1}|S_t)$. This process is continued until the chain is stopped and a state is output. Because the number of steps T taken in the chain until it is stopped is distributed according to a geometric distribution with parameter $(1-\gamma)$, we can see using Equation (2) that the random state output by this process will also be distributed according to π.

For the application considered in this paper, it will be useful to consider a generalization of this random process. Specifically, we will construct an MC where, once we have decided to stop the MC (which happens with probability γ on each step), we will allow the state to transition one final time according to a new set of transition probabilities $p''(S_{t+1}|S_t)$ (different from the transition probabilities used in the earlier steps of the walk), and finally output S_{t+1}. Note that if $p''(S_{t+1}|S_t) = 1$ iff $S_{t+1} = S_t$, this reduces to the simpler type of random walk described earlier. In Section 3 we will see how permitting an extra state-transition step at the end allows us to build significantly more expressive models.

3. Random walks for PP attachment

3.1. The PP attachment model

Following most of the literature on Prepositional Phrase (PP) attachment (e.g., Collins & Brooks, 1995;

Table 1. The sparsity of the data: the percent of times tuples in the test set had appeared in the training set.

Factor		**% Non-Zero**
Verbal	$P(p, va)$	99.8
	$P(v\|p, va)$	64.8
	$P(n_1\|p, va, v)$	15.7
	$P(n_2\|p, v, va)$	13.8

Stetina & Nagao, 1997; Harabagiu & Pasca, 1999; Pantel & Lin, 2000; Brill & Resnik, 1994), we focus on the most common configuration that leads to ambiguities: V NP PP. Here, working bottom-up in parsing, the goal is to determine if the PP should be attached to the verb or to the object noun phrase. Previous work has shown the central (but not exclusive) role played by the head words of phrases in resolving such ambiguities, and we follow common practice in representing the problem using only the head words of these constituents and of the NP inside the PP. For example, given the tuple:

(3) v:hang n_1:painting p:with n_2:nail

we would like to determine if the prepositional phrase *with nail* should modify the verb *hang*, or the noun phrase headed by *painting*. Here, clearly, *with (a) nail* modifies the verb *hang*.

We start by building a generative model for the probability of the sequence of four head words and the attachment site $P(V, N_1, P, N_2, Att)$, where V is a verb, P a preposition, and N_1 and N_2 are the two head nouns involved in the attachment problem. The variable Att has as value either va (for verbal attachment) or na (nominal/noun attachment). Using a model for this joint distribution, we can compute the conditional distribution $P(Att|V, N_1, P, N_2)$ and use that to predict the more likely attachment type.

The model makes only two context-specific independence assumptions: that given a verbal attachment, the second noun is independent of the first noun, and that given a nominal attachment, the second noun is independent of the verb. More specifically, the model decomposition is as follows:

$$P(v, n_1, p, n_2, va) = P(p, va)P(v|p, va)P(n_1|p, va, v)P(n_2|p, v, va) \quad (4)$$

$$P(v, n_1, p, n_2, na) = P(p, na)P(v|p, na)P(n_1|p, na, v)P(n_2|p, n_1, na) \quad (5)$$

Each of the factors above, except for $P(p, Att)$, are estimated using random walks.

To illustrate the degree of data sparsity for this problem, Table 1 shows the percentage of test cases for which we had a non-zero relative frequency estimate from the training set for each of the factors needed for Equation 4. As can be seen, for the factors involving two words in addition to the preposition, more than 3/4 of the time we have not seen the tuple in the training set.

3.2. Random walks

We now describe our random walk model for the word dependency distributions needed for equations 4–5. We illustrate with the case of estimating $P(n_2|p, v, va)$. Instantiating the example in (3), this is $P(N_2 = nail|P = with, V = hang, va)$, the probability that, given *hang* is modified by a PP whose head is *with*, *nail* is the head of the noun phrase governed by *with*. This is strictly a tri-lexical dependency, but because prepositions can often be regarded as just a marker of the semantic role of their object noun phrase, we can informally think of this as estimating the probability of a particular sort of semantic dependency; here it is the likelihood of n_2:nail bearing a *with*-type dependency to the word v:hang. Thus, given the preposition, we can view this as estimating a bi-lexical dependency between a verb v and a noun n_2.

We will estimate this probability using a Markov chain. More precisely, we will construct a MC M (whose transition probabilities will depend on p, v, and the fact that $Att = va$) so that its stationary distribution π is a good approximation to $P(n_2|p, v, va)$.

We let the state space $\mathcal{S}$ of our random walk be $\mathcal{W} \times \{0, 1\}$, where $\mathcal{W}$ is the set of all words. Thus, a state is a pair consisting of a word and a single "bit" taking on a value of 0 or 1. As we will shortly see, the extra memory bit allows our walk to "remember" if the word in the current state is a head (0) or a dependent (1), and will permit us to build richer models.[2] For $P(n_2|p, v, va)$, v is a head, and n_2 is a dependent (and the type of the dependency relationship is indicated by p). Below we will write (*nail*, 1) as ${}^d nail$, both for brevity, and to remind us of the extra bit's meaning.

The initial distribution p_0 of our Markov chain puts probability 1 on the state ${}^h v$ (i.e., we always start at the state for the head verb, with the bit-value 0).

Let us first walk through some cases using the "hang painting with nail" example, with the small random walk model shown in figure 1. For the sake of this example, it will be convenient to begin with the case of $T = 1$. We are trying to estimate

$$p(N_2 = nail|V = hang, P = with, Att = va). \quad (6)$$

If, in a training set of disambiguated PP-attachment examples, we have seen the event ($V = hang, P =$

[2]Other examples of Markov chains that can be thought of as random walks with an extra memory bit include (Lafferty & Zhai, 2001; Ng et al., 2001).

with, $Att = va$) before, then clearly one possible estimate for the probability in (6) might be given by its empirical distribution. Specifically, if *nail* was frequently seen in the context of the event ($V = hang, P = with, Att = va$), then we would like to assign a large probability to this event. One way to ensure that the random walk frequently visits *nail* in this setting is therefore to have the probability of transitioning from the initial state to some other state ${}^{d}w$, representing a dependent word, be monotonically increasing in the empirical distribution of $p(N_2 = w|V = hang, P = with, Att = va)$.

Now, suppose that, because of data sparseness problems, we have not seen "v:hang p:with n_2:nail" in our training set, but that we have seen "v:hang p:with n_2:nails" several times. Further, our stemmer indicates that *nail* and *nails* have the same root form. In this setting, we would still like to be able to assign a high probability to $p(nail|hang, with, va)$. I.e., we want π to give ${}^{d}nail$ a large probability. Using the state transitions described above, we already have a large probability of visiting ${}^{d}nails$. If our random walk now gives a large probability of transitioning from ${}^{d}nails$ to ${}^{d}nail$, then we would be done. More broadly, we would like our random walk to be able to make a transition from (w_1, b_1) to (w_2, b_2), if w_1 and w_2 are words with the same root form, and $b_1 = b_2$.

Similarly, if we know that $p(rivet|hang, with, va)$ has a large probability, and if some external knowledge source tells us that *rivet* and *nail* are semantically closely related, then we should infer that $p(nail|hang, with, va)$ should also be fairly large. This can be done by using a thesaurus, or a resource like WordNet, a large collection of words classified into a set of senses (synsets), which are organized in a hierarchy, and permitting transitions between (w_1, b_1) and (w_2, b_2) if an external knowledge source tells us that w_1 and w_2 are related, and $b_1 = b_2$.[3]

More broadly, we have outlined above several different "types" of inferences that can be made about what tuples v, p, n_2 are likely. These types of inferences often exploit external knowledge sources (such as a stemmer, or WordNet), and we have shown several examples of how they can be encoded into a random walk framework, so that the stationary distribution gives a large probability to events that we would like our procedure to conclude are likely. Note in particular that if there are multiple paths to a node, then that "reinforces" a particular conclusion. By combining multiple steps of these inferences together, in figure 1, we should be able to conclude that if (a) *hang with hook*, (b) *fasten with hook* and (c) *fasten with rivet* are likely; that (d) *hooks* and *hook* have the same root, and if (e) *rivet* and *nail* are semantically related, then *fasten with nail* is also likely. Specifically, the sequence of states the random walk might visit based on this information is ${}^{h}hang \xrightarrow{a} {}^{d}hooks \xrightarrow{d} {}^{d}hook \xrightarrow{b} {}^{h}fasten \xrightarrow{c} {}^{d}rivet \xrightarrow{e} {}^{d}nail$. Thus, by considering multiple steps of the random walk, we can combine multiple steps of inference together. But the model by its nature also captures that long multi-step chains do not give much support to their conclusion.

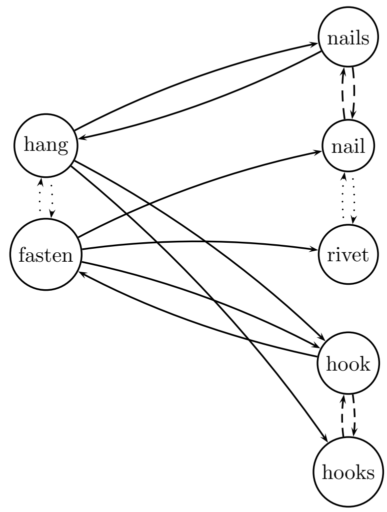

Figure 1. A small words state space for learning the distribution $P_{with}(n_2|v)$.

3.3. Formal model

We now describe our model formally. Our Markov chain's transition probabilities are built up using a set of different **links**, which should be thought of as "basic" transition distributions that correspond to different possible inference steps.

Each link type always leads from states where the memory bit takes on some particular value b_1 to states where the bit takes on a value b_2 (not necessarily different from b_1). The final transition distribution will then be a mixture of the basic transition distributions, where the mixture weights are learned automatically.

Let the **links** $l_1, \ldots, l_k$ be given by transition matrices $T^1, \ldots, T^k$. Each matrix T^i has rows for states with memory bits $startBit(i)$ and its rows are distributions over successor states with memory bit $endBit(i)$.

The probability of transitioning from (w, b) to (w', b')

[3]Of course, some of these could lead to incorrect inferences—even though *hang with nail* may be likely, and WordNet indicates that nail and nail-polish are semantically related, it is incorrect to infer that *hang with nail-polish* is therefore likely. However, we will later describe how a learning algorithm is used to automatically decide the degree to which each of these inferences can be trusted.

in the Markov chain is given by:

$$P(w', b'|w, b) = \sum_{i:startBit(i)=b, endBit(i)=b'} \lambda(w, b, i) T^i(w', b'|w, b)$$

The parameter $\lambda(w, b, i)$ is the weight of link l_i for the state (w, b). It can also be viewed as the probability of taking a link of type l_i given the current state (w, b). The probabilities $\lambda(w, b, i)$ sum to 1 over all links l_i having a starting bit $startBit(i) = b$. Parameters of this form for all states are estimated automatically from data. Since estimating separate parameters for each word would introduce too much sparsity, we define equivalence classes of states for which we tie the parameters. To avoid constrained optimization, we handled the constraints $\sum_i \lambda(w, b, i) = 1, \forall w, b$ and $\lambda(w, b, i) \geq 0$ by representing $\lambda(w, b, i) = e^{\gamma(w,b,i)} / \sum_{i'} e^{\gamma(w,b,i')}$. The new model parameters are the $\gamma(w, b, i)$ and they are not constrained.

As mentioned in Section 2, we also add one more refinement to the model, by further distinguishing between two different kinds of links: ones that can be followed at any time, and ones that can be taken only in a final (T-th) step of the walk. We call the latter type **final** links. The intuition here is that (due to the usual sparseness in NLP data) we do wish to include in our model distributions that back off from conditioning on individual words and that therefore can transition to a highly-smoothed model. But, it would be undesirable to allow transitions to backed-off distributions throughout the random walk. Specifically, allowing such transitions would cause us to lose the intuition of the random walk as exploring close neighbors of a word based on some similarity criterion. An additional advantage of having a special stopping distribution is that we can disable transitions to states that don't have the desired memory bit; for example if the random walk is estimating $P(N_2|v, p, va)$, the last state has to be a dependent. Thus in a final step of the walk, the probability of following a link type leading to a non-dependent state is fixed to zero.

Thus we learn two different transition distributions of the Markov chain — a distribution $P^{nfin}(w', b'|w, b)$, and a distribution $P^{fin}(w', b'|w, b)$. The **final** links participate only in P^{fin}, whereas the other links participate in both P^{fin} and P^{nfin}.

The parameters of the model were fitted to optimize the conditional log-likelihood of the correct attachment sites for a development set of samples, disjoint from the training and test sets, including quadratic regularization. That is, we maximized the objective:

$$\sum_{i=1,\ldots,N} \log P(Att^i|v^i, {n_1}^i, p^i, {n_2}^i) - \lambda \sum_{s=1,\ldots,k} {\gamma_s}^2$$

Here i ranges over the sample set, and s ranges over the model parameters. We performed the optimization using a limited memory quasi-Newton method.

The number of parameters depends on the scheme for defining equivalence classes over states. The parameters correspond to distributions over link types for states and stopping probabilities. The stopping probabilities can also depend on the particular Markov chain. We experimented with binning the parameters based on observed number of times of occurrence of words but the simplest model having a single equivalence class performed on average as well as the more complex models.

3.4. Link types for PP attachment

For modeling $P(N_2|p, v, va)$, we have separate Markov chain transition matrices for each preposition p, with the link types given below. The initial state distribution places probability 1 on the state ${}^hv$. The first eight link types are:

1. **V → N.** Transitions from ${}^hw_1$ to ${}^dw_2$ with probability proportional to the empirical probability of $p(N_2 = w_2|V = w_1, p, Att)$. (**L1**)
2. **Morphology.** Transitions from (w_1, b) to (w_2, b) for all words w_2 that have the same root form as w_1, with probability proportional to the empirical count of w_2 plus a small smoothing parameter α. (**L2Nouns, L2Verbs**)
3. **WordNet Synsets.** Transitions from states (w_1, b) to (w_2, b), for all words w_2 in the same WordNet synonym-set as one of the top three most common senses of w_1, with probability proportional to the empirical count of w_2 plus a small smoothing parameter α. (**L3Nouns, L3Verbs**)
4. **N → V.** Transitions from ${}^dw_1$ to ${}^hw_2$ with probability proportional to the empirical probability of $p(V = w_2|N_2 = w_1, p, Att)$. (**L4**)
5. **External corpus.** Same as link L1, but the empirical probabilities are measured from an additional set of noisy samples, generated automatically by a statistical parser . (**L5**)
6. **V → V.** Transitions from ${}^hw_1$ to ${}^hw_2$ with probability proportional to their distributional similarity with respect to dependents they take. This is defined more precisely in Section 4.
7. **N → N.** Analogously to the previous link type, these are transitions among nouns with probability proportional to their distributional similarity with respect to heads they modify.
8. **V → V.** Transitions from ${}^hw_1$ to ${}^hw_2$ with probability proportional to their distributional similarity over noun objects when modified by p.

We also used the following final links to add at the end over-smoothed back-off distributions. These represent all levels in a linear back-off sequence estimated from the training corpus, and a single level of back-off from the additional corpus of noisy samples. Note that these distributions are the same for every state:

9-12. **Backoff1 through Backoff4** Transitions to ${}^{d}w_2$ with probability proportional to $P(N_2 = w_2|P, Att), P(N_2 = w_2|Att), P(N_2 = w_2|.)$, and uniform respectively. (**L9**,**L10**,**L11**,**L12**)

13. **Backoff5.** Transitions to ${}^{d}w_2$ with probability proportional to $\hat{P}(N_2 = w_2|P, Att)$ estimated from the additional noisy corpus. (**L13**)

Additionally, we add identity links (self-loops), to avoid situations where no link type can be followed.

4. Experiments

We work with the Penn Treebank Wall Street Journal data (Ratnaparkhi et al., 1994), which consists of four-tuples of head words and a specification of the type of attachment. There are 20,801 samples in the training set, 4,039 in the development set, and 3,097 samples in the test set. This same data has been used by several other researchers (Ratnaparkhi et al., 1994; Collins & Brooks, 1995; Stetina & Nagao, 1997). The back-off model of (Collins & Brooks, 1995) is the best-performing previously published algorithm for the task of classifying samples by only using statistics of word occurrences from the training corpus and we use it as one of our baselines. Better results have been reported on this test set by (Stetina & Nagao, 1997) (88.1%) and on other datasets by (Harabagiu & Pasca, 1999), but the C&B algorithm is the clearest established baseline of good performance, since it does not rely on additional processing stages such as word sense disambiguation or use of named entity recognizers. Previous studies of human performance suggest an upper bound on attachment accuracy, given just the four-tuple of head words, of 88.2% (Ratnaparkhi et al., 1994). Therefore it is plausible to accept a Bayes error of about 10% for this task.

Our algorithm uses the training set to estimate empirical distributions and the development set to train the parameters of the random walk. We report accuracy results on the final test set.

In addition to this training data set, we generate additional much noisier training data, using the BLLIP corpus. BLLIP is a corpus of 1,796,386 automatically parsed English sentences (Charniak, 2000). From the parsed sentences, we extracted tuples of four headwords and attachment site for ambiguous verbal or noun PP attachments. This made for a total of 567,582 tuples. We will call this data-set BLLIP-PP. One can expect this data to be rather noisy, since PP attachment is one of the weakest areas for state of the art statistical parsers. We pre-processed the data by lower-casing the verbs and prepositions, and by substituting all digits with the X symbol.

For all models, we ran the Markov chain for at most some d time steps (which may depend on the type of links used); we call d the maximum degree of the Markov chain. (I.e., Equation 2 is truncated after d terms, and renormalized to sum to 1.) Accuracy results on the test set are shown in Table 2. The last column of the table shows significant differences according to McNemar's test.

We first report the accuracy of the simplest walk of maximum degree 1, which is of the same form as the familiar linear mixture models. This walk estimates the probability $P(n_2|p, v, va)$, using link types L1, L9, L10, L11, and L12 as follows:

$$\begin{aligned} P(n_2|p, v, va) &= \lambda_0(p, v, va)\hat{P}(n_2|p, v, va) \\ &+ \lambda_1(p, v, va)\hat{P}(n_2|p, va) \\ &+ \lambda_2(p, v, va)\hat{P}(n_2|va) \\ &+ \lambda_3(p, v, va)\hat{P}(n_2) + \lambda_4(p, v, va)\frac{1}{V} \end{aligned}$$

Similar walks are constructed for all other factors needed for the generative model. Since the maximum degree is 1, only transitions according to a final distribution P^{fin} are taken $P(n_2|p, v, va) = P^{fin}_{p,va}(n_2|v)$.

This is our baseline model and we name it **Baseline**. The accuracy results for the Baseline model are shown in row 4 of table 2. It is worth noting that our simple generative model with linearly interpolated relative frequency estimates (and interpolation parameters fitted discriminatively), performs better than the discriminative back-off C&B algorithm (shown in row 1). The last column shows that it is significantly better at level .005 (p-value .0012).

Next we describe the incremental addition of links to our model, with discussion of the performance achieved. We fix the maximum degree of the walks for estimating the uni-lexical dependencies $P(v|p, Att)$ to $d = 2$, and the maximum degree of all other walks, estimating bi-lexical dependencies, to $d = 3$.[4]

1. **Morphology.** Adding a link between verbs (**L2Verbs**) was helpful. This link was added to the Markov chains for estimating $P(V|p, Att)$, $P(N_1|v, p, Att)$, and $P(N_2|v, p, va)$. The accuracy on the test set was 86.08%, as shown in row 6 of

[4]For computational reasons, we have only explored paths of maximum degree three for models with many features. For smaller models, higher degree walks show an advantage. Thoroughly investigating the contribution of longer walks is left to future research.

Table 2. Summary of results on the final test set of 3,097 samples. In the significance column $>$ means at level .05 and $\gg$ means at level .005.

		Model	Link Types	Degree	Accuracy	Signif
Baselines	1	C&B			84.18%	
	2	C&B + stem verbs			84.50%	not > 1
	3	C&B on BLLIP-PP			85.53%	> 1
	4	Baseline	L1,L8,L9,L10,L11	1,1	85.86%	$\gg 1$
	5	Baseline + stem verbs	L1,L8,L9,L10,L11	1,1	86.02%	$\gg 2$
Random Walks	6	Morph Verbs	+ L2Verbs	2,3	86.08%	not > 4, $\gg 2$
	7	Morph Verbs and Nouns	+ L2Nouns	2,3	86.18%	not > 4, $\gg 2$
	8	Morphology & Synonyms	+L3Verbs,L3Nouns	2,3	86.53%	not > 4, $\gg 2$
	9	Sim_{JS_β}	Baseline +L2Verbs+L6,L7	2,3	86.44%	not > 4, $\gg 2$
	10	Final	see text	2,3	87.54%	> 9,$\gg 3$,$\gg 4$

Table 2. The accuracy of this model is slightly better than the accuracy of **Baseline** ran on a stemmed version of the corpus (shown in row 5). Adding noun morphology as well was also helpful as can be seen in row 7 of the table. The models with morphology are both significantly better than the **C&B** algorithm trained and tested on the data preprocessed by stemming verbs (shown in row 2), but they are not significantly better than **Baseline**.

2. **WordNet Synsets.** We use WordNet in a simple way – for every word, we find its top three most common senses, and make a link from the word to all other words having those senses. We obtained accuracy gains from adding the synonym links, as can be seen in row 8 of the table. However, this improvement over the baseline is not significant at level .05. (It is significant at .1.)

3. **Similarity based on Jensen-Shannon divergence.** We add links between states with the same memory bit with transition probabilities proportional to their distributional similarity. For the sake of concreteness, consider a random walk for estimating $P(N_2|p, v, va)$. Let q_v denote the empirical distribution of dependents of the preposition p modifying verb v: $\hat{P}(N_2|p, v, va)$ estimated from the BLLIP-PP corpus. We define a similarity function between verbs $sim_{JS_\beta}(v_1, v_2) = \exp(-\beta JS(q_{v_1}, q_{v_1}))$. JS stands for *Jensen-Shannon divergence* between two probability distributions (Rao, 1982) and is defined in terms of the KL *divergence* D as:
$JS(q_1, q_2) = \frac{1}{2}\{D(q_1||avg_{q_1,q_2}) + D(q_2||avg_{q_1,q_2})\}$

   The same similarity function was used in (Dagan et al., 1999; Lee, 1999). We add a link from verbs to verbs (link type **L6**) that has transitions from each verb, to its top K closest neighbors in terms of the similarity sim_{JS_β}.[5] The transition probability is the normalized value of the similarity. Similarly we add links between nouns based on their similarity sim_{JS_β} with respect to the empirical distribution $\hat{P}(V|p, n, va)$ in BLLIP-PP (link type **L7**).

   Up until now we have been discussing the $P(N_2|p, v, va)$ dependency distribution. For the other dependency relations distributions – $P(N_2|p, n_1, na)$, and $P(N_1|p, v, Att)$, we similarly add links between the heads based on their sim_{JS_β} with respect to the empirical distribution of their dependents in BLLIP-PP, and between the dependents proportional to their similarity sim_{JS_β} of head distributions. The accuracy of the resulting model, when these links are added is shown in row 9 of Table 2. This model does not include the noun morphology and synonym features as the model in row 8. We found that these links were no longer helpful after the addition of more powerful features.

4. **Final Model.** The major addition to the final model is link **L5**, which is relative frequency estimated from BLLIP-PP, and the backoff link **L13**, also a relative frequency estimate from BLLIP-PP.

   The final model includes the links from the **Baseline**, **L5**, **L13**, morphology for verbs, and the previously discussed sim_{JS_β} links. In addition, one more kind of sim_{JS_β} links was added – **L8**.

   Other algorithms can also make use of additional noisy training data. We ran the **C&B** algorithm on the union of the training data and BLLIP-PP and its accuracy was also improved as shown in row 2 of the table. However, the random walk model is able to make better use of the additional noisy data, as it learns suitable weights for the estimates obtained from it.

   Considering the relatively unsuccessful attempts in the past to use additional unsupervised data to

[5] In our experiments, β was 50, and K was 25.

improve lexical estimates for statistical parsers, this result is very encouraging, and shows the importance of learning proper weights for additional noisy data.[6]

The final model had an accuracy of 87.54%, which is close to the upper bound. According to the statistical significance tests, the final model is significantly better then the baseline, **C&B** trained on the BLLIP-PP data, and model Sim_{JS_β}.

5. Discussion and conclusions

Random walk models provide a general framework for unifying and combining various notions of similarity-based smoothing. A walk of length 1 is just a linear interpolation, with interpolation weights typically set empirically as we do here (with the difference that we train to maximize conditional rather than joint likelihood). A walk of length 3 following exactly one forward link (like **L1**), followed by one backward link (like **L4**), and another forward link gives exactly the same estimate as co-occurrence smoothing (Essen & Steinbiss, 1992; Lee, 1999). A walk of length 2 using only one kind of similarity between head states, and forward links, is similar to distributional similarity smoothing (Lee, 1999).

But the random walks framework that we propose is much more general. A multitude of link types can be defined in it, and they are automatically weighted by the learning algorithm. Paths of shorter and longer lengths can be followed (though the most highly contributing paths are the shorter ones). The generality of this approach to similarity-based smoothing not only gives a high performance prepositional phrase attachment system, but holds the promise of *learning* complex but effective random walk models in other domains.

Acknowledgments. Our thanks to the anonymous reviewers for helpful comments and to Jenny Finkel for the optimization code. This work was supported in part by the ARDA AQUAINT program, and in part by the Department of the Interior/DARPA under contract number NBCHD030010.

References

Bikel, D. M. (2003). *Intricacies of Collins' parsing model* (Technical Report TR No. MS-CIS-03-11). University of Pennsylvania.

Brémaud, P. (1999). *Markov chains: Gibbs fields, monte carlo simulation, and queues*. Springer-Verlag.

Brill, E., & Resnik, P. (1994). A rule-based approach to prepositional phrase attachment disambiguation. *Proceedings of COLING*.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *WWW7/Computer Networks and ISDN Systems*, *30*, 107–117.

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. *Proc. 14th National Conference on Artificial Intelligence* (pp. 598 – 603).

Charniak, E. (2000). A maximum-entropy-inspired parser. *NAACL 1* (pp. 132–139).

Collins, M. (1999). *Head-driven statistical models for natural language parsing*. Doctoral dissertation, University of Pennsylvania.

Collins, M., & Brooks, J. (1995). Prepositional attachment through a backed-off model. *Proceedings of the Third Workshop on Very Large Corpora* (pp. 27–38).

Dagan, I., Lee, L., & Pereira, F. (1999). Similarity-based models of cooccurrence probabilities. *Machine Learning*, *34*, 43–69.

Essen, U., & Steinbiss, V. (1992). Cooccurrence smoothing for stochastic language modeling. *ICASSP* (pp. 161–164).

Goodman, J. T. (2001). A bit of progress in language modeling. *MSR Technical Report MSR-TR-2001-72*.

Harabagiu, S., & Pasca, M. (1999). Integrating symbolic and statistical methods for prepositional phrase attachment. *Proceedings of FLAIRS-99* (pp. 303–307).

Hindle, D., & Rooth, M. (1993). Structural ambiguity and lexical relations. *Computational Linguistics*, *19*, 103–120.

Hull, D. (1996). Stemming algorithms – A case study for detailed evaluation. *Journal of the American Society for Information Science*, *47*, 70–84.

Kleinberg, J. (1998). Authoritative sources in a hyperlinked environment. *9th ACM-SIAM Symposium on Discrete Algorithms*.

Lafferty, J., & Zhai, C. (2001). Document language models, query models, and risk minimization for information retrieval. *SIGIR* (pp. 111–119).

Lee, L. (1999). Measures of distributional similarity. *37th Annual Meeting of the Association for Computational Linguistics* (pp. 25–32).

Ng, A. Y., Zheng, A. X., & Jordan, M. (2001). Link analysis, eigenvectors, and stability. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*.

Pantel, P., & Lin, D. (2000). An unsupervised approach to prepositional phrase attachment using contextually similar words. *ACL:00* (pp. 101–108).

Rao, R. C. (1982). Diversity: Its measurement, decomposition, aportionment and analysis. *The Indian Journal of Statistics*, *44*, 1–22.

Ratnaparkhi, A., Reynar, J., & Roukos, S. (1994). A maximum entropy model for prepositional phrase attachment. *Workshop on Human Language Technology*.

Stetina, J., & Nagao, M. (1997). Corpus based PP attachment ambiguity resolution with a semantic dictionary. *Proc. 5th Workshop on Very Large Corpora* (pp. 66–80).

[6] In Charniak (1997), an experiment where 30 million words of text were parsed automatically and added as additional training data for a parser estimating lexical dependency probabilities, resulted in a rather small accuracy increase – 0.1% precision and recall.