Hitting the Right Paraphrases in Good Time

Stanley Kok
Department of Computer Science & Engineering
University of Washington
Seattle, WA 98195, USA
koks@cs.washington.edu

Chris Brockett
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
chrisbkt@microsoft.com

Abstract

We present a random-walk-based approach to learning paraphrases from bilingual parallel corpora. The corpora are represented as a graph in which a node corresponds to a phrase, and an edge exists between two nodes if their corresponding phrases are aligned in a phrase table. We sample random walks to compute the average number of steps it takes to reach a ranking of paraphrases with better ones being "closer" to a phrase of interest. This approach allows "feature" nodes that represent domain knowledge to be built into the graph, and incorporates truncation techniques to prevent the graph from growing too large for efficiency. Current approaches, by contrast, implicitly presuppose the graph to be bipartite, are limited to finding paraphrases that are of length two away from a phrase, and do not generally permit easy incorporation of domain knowledge. Manual evaluation of generated output shows that our approach outperforms the state-of-the-art system of Callison-Burch (2008).

1 Introduction

Automatically learning paraphrases, or alternative ways of expressing the same meaning, is an active area of NLP research because of its usefulness in a variety of applications, e.g., question answering (Lin and Pantel, 2001; Ravichandran and Hovy, 2002; Reizler et al., 2007), document summarization (Barzilay et al., 1999; McKeown et al., 2002), natural language generation (Iordanskaja et al., 1991; Lenke, 1994; Stede, 1999), machine translation (Kauchak and Barzilay, 2006; Callison-Burch et al., 2006; Madnani et al., 2007).

Early work on paraphrase acquisition has focused on using monolingual parallel corpora (Barzilay and McKeown, 2001; Barzilay and Lee, 2003; Pang et al., 2003; Quirk et al., 2004). While effective, such methods are hampered by the scarcity of monolingual parallel corpora, an obstacle that limits both the quantity and quality of the paraphrases learned. To address this limitation, Bannard and Callison-Burch (2005) focused their attention on the abundance of bilingual parallel corpora. The crux of this system (referred to below as ”BCB”) is to align phrases in a bilingual parallel corpus and hypothesize English phrases as potential paraphrases if they are aligned to the same phrase in another language (the "pivot"). Callison-Burch (2008) further refines BCB with a system that constrains paraphrases to have the same syntactic structure (Syntactic Bilingual Phrases: SBP).

We take a graphical view of the state-of-the-art BCB and SBP approaches by representing the bilingual parallel corpora as a graph. A node corresponds to a phrase, and an edge exists between two nodes if their corresponding phrases are aligned. This graphical form makes the limitations of the BCB/SBP approaches more evident. The BCB/SBP graph is limited to be bipartite with English nodes on one side and foreign language nodes on the other, and an edge can only exist between nodes on different sides. This neglects information between foreign language nodes that may aid in learning paraphrases. Further, by only considering English nodes that are linked via a foreign language node as potential paraphrases,

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 145–153,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

these approaches will fail to find paraphrases separated by distances greater than length two.

In this paper, we present HTP (Hitting Time Paraphraser), a paraphrase learning approach that is based on random walks (Lovász, 1996) and hitting times (Aldous and Fill, 2001). Hitting time measures the average number of steps one needs to take in a random traversal of a graph before reaching a destination node from a source node. Intuitively, the smaller the hitting time from a phrase E to E' (i.e., the closer E' is to E), the more likely it is that E' is a good paraphrase of E. The advantages of HTP are as follows:

- By traversing paths of lengths greater than two, our approach is able to find more paraphrases of a given phrase.
- We do not require the graph to be bipartite. Edges can exist between nodes of different foreign languages if their corresponding phrases are aligned. This allows information from foreign phrase alignments to be used in finding English paraphrases.
- We permit domain knowledge to be easily incorporated as nodes in the graph. This allows domain knowledge to favor good paraphrases over bad ones, thereby improving performance.

In this paper, we focus on learning English paraphrases. However, our system can be applied to learning paraphrases in any language.

We begin by reviewing random walks and hitting times in the next section. Then we describe our paraphrase learning algorithm (Section 3), and report our experiments (Section 4). We discuss related work in Section 5. Finally, we conclude with future work (Section 6).

2 Background

A directed graph consists of a set of nodes V, and a set of edges E. A directed edge is a pair (i, j) where $i, j \in V$. Associated with the graph is a $|V| \times |V|$ adjacency matrix W. Each entry W_{ij} in the matrix is the weight of edge (i, j), or zero if the edge does not exist.

In a *random walk* (Lovász, 1996), we traverse from node to node via the edges. Suppose at time step t, we are at node i. In the next step, we move to its neighbor j with probability proportional to the weight of the edge (i, j), i.e., with probability $W_{ij} / \sum_j W_{ij}$. This probability is known as the *transition probability* from i to j. Note that the transition probabilities from a node to its neighbors sum to 1.

The *hitting time* h_{ij} (Aldous and Fill, 2001) from node i to j is defined as the average number of steps one takes in a random walk starting from i to visit j for the first time. Hitting time has the property of being robust to noise. This is a desirable property for our system which works on bilingual parallel corpora containing numerous spurious alignments between phrases (i.e., edges between nodes). However, as observed by Liben-Nowell and Kleinberg (2003), *hitting time* has the drawback of being sensitive to portions of the graph that are far from the start node because it considers paths of length up to ∞.

To circumvent this problem, Sarkar and Moore (2007) introduced the notion of *truncated* hitting time where random walks are limited to have at most T steps. The truncated hitting time h_{ij}^T from node i to j is defined as the average number of steps one takes to reach j for the first time starting from i in a random walk that is limited to at most T steps. h_{ij}^T is defined to be 0 if $i = j$ or $T = 0$, and to be T if j is not reach in T steps. As $T \to \infty$, $h_{ij}^T \to h_{ij}$.

In a recent work, Sarkar et al. (2008) showed that truncated hitting time can be approximated accurately with high probability by sampling. They run M independent length-T random walks from node i. In m of these runs, node j is visited for the first time at time steps $t_j^1, \ldots, t_j^m$. The *estimated* truncated hitting time is given by

$$\hat{h}_{ij}^T = \frac{\sum_{k=1}^m t_j^k}{M} + (1 - \frac{m}{M})T \qquad (1)$$

They also showed that the number of samples of random walks M has to be at least $\frac{1}{2\epsilon^2} \log \frac{2n}{d}$ in order for the estimated truncated hitting time to be a good estimate of the actual truncated hitting time with high probability, i.e., for $P(|\hat{h}_{ij}^T - h_{ij}^T| \le \epsilon T) \ge 1 - \delta$, where n is the number of nodes in the graph, ϵ and δ are user-specified parameters, and $0 \le \epsilon, \delta \le 1$.

3 Hitting Time Paraphraser (HTP)

HTP takes a *query* phrase as input, and outputs a list of paraphrases, with better paraphrases at the top of

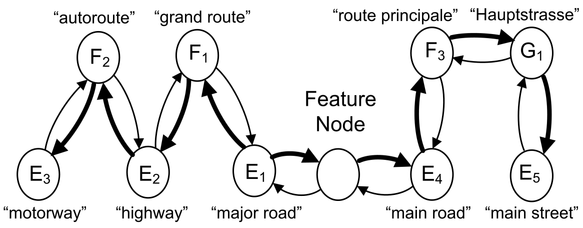

Figure 1: Graph created from English-French (E-F), English-German (E-G), and French-German (F-G) bilingual parallel corpora. Bold edges have large positive weights (high transition probabilities).

the list. HTP also requires as input a set of bilingual parallel corpora that have been processed into phrase tables of the kind used in statistical machine translation.

A bilingual parallel corpus is made up of sentences in two languages. Two sentences that are translations of one another are paired together, and a phrase in one sentence is aligned with a phrase in the other with the same meaning. From such alignments, we can count for a phrase E both the number of times it occurs ($Count_E$), and the number of times it is aligned with a phrase F in the other language ($Count_{E,F}$). With these counts we can estimate the probability of F given E as $P(F|E) = \frac{Count_{E,F}}{Count_E}$.

HTP represents the aligned phrases as a graph. A node corresponds to a phrase, and a directed edge exists from node i to j if their corresponding phrases are aligned. The weight of edge (i, j) is given by $P(j|i)$ which is computed as described in the previous paragraph.

Table 1: The HTP algorithm.

```
function $HTP(E, C, d, n, m, T, \delta, l)$
  input: $E$, query phrase
         $C$, tables of aligned phrases
         $d$, maximum distance of nodes from $E$
         $n$, maximum number of nodes in graph
         $m$, number of samples of random walks
         $T$, maximum number of steps taken by a
           random walk
         $\delta$, probability that estimated truncated hitting
           time deviates from actual value by a large
           margin (see Equation 1)
         $l$, number of top outgoing edges to select at
           each node in a random walk
  output:$(E'_1, \ldots, E'_k)$, paraphrases of $E$ ranked in
         order of increasing hitting times
  calls: $CreateGraph(E, C, d, n)$ creates graph $G$
           from $C$ containing at most $n$ nodes that are
           at most $d$ steps from $E$
         $EstimateHitTimes(E, G, m, T, \delta)$, estimates
           the truncated hitting times of each node in $G$
           by running $m$ random walks
         $PruneNodes((E_1, \ldots, E_k), G)$, removes nodes
           from $G$ if their hitting times is equal to $T$.
         $AddFeatureNodes(G)$, adds nodes
           representing domain knowledge to $G$

$G \leftarrow CreateGraph(E, C, d, n)$
$(E_1, \ldots, E_k) \leftarrow EstimateHitTimes(E, G, m, T, \delta)$
$G' \leftarrow PruneNodes((E_1, \ldots, E_k), G)$
$G'' \leftarrow AddFeatureNodes(G')$
$(E'_1, \ldots, E'_k) \leftarrow EstimateHitTimes(E, G'', m, T, \delta)$
return $(E'_1, \ldots, E'_k)$
```

Figure 1 gives an example of a graph created from English-French, English-German, and French-German parallel corpora. We use this figure to illustrate the strengths of HTP. First, by using moderately long random walks, HTP is able to find paraphrases that are separated by long paths. For example, there is a high probability path of length 4 $(E_1, F_1, E_2, F_2, E_3)$ from E_1 to E_3. Because of the path's high probability, it will appear in many of the random walks starting from E_1 that are sampled on the graph, and thus E_3 will be visited in many of the samples. This causes the truncated hitting time $h^T_{E_1 E_3}$ to be small, allowing HTP to find E_3 as a plausible paraphrase of E_1. Second, by allowing edges between nodes of different foreign languages (i.e., by not requiring the graph to be bipartite), HTP allows strong correlation between foreign language nodes to aid in finding paraphrases. In the figure, even though E_4 and E_5 are not linked via a common foreign language node, there is a high probability path linking them (E_4, F_3, G_1, E_5). This allows HTP to find E_5 as a reasonable paraphrase of E_4. Third, HTP enables domain knowledge to be incorporated as nodes in the graph. For example, we could incorporate the domain knowledge that phrases with lots of unigrams in common are likely to be paraphrases. In Figure 1, the "feature" node represents such knowledge, linking E_4 and E_1 as possible paraphrases even though they have no foreign language nodes in common. Note that such domain knowledge nodes can be linked to arbitrary nodes, not just English ones.

The HTP algorithm is shown in Table 1. It takes as input a query phrase and a set of bilingual phrase tables. The algorithm begins by creating a graph from the phrase tables. Then it estimates the truncated hitting times of each node from the query node by sampling random walks of length T. Next it prunes nodes (and their associated edges) if their truncated hitting times are equal to T. To the resulting graph, it then adds nodes representing domain knowledge and estimates the truncated hitting times of the nodes by sampling random walks as before. Finally, it returns the nodes in the same language as the query phrase in order of increasing hitting times.

3.1 Graph Creation

An obvious approach to creating a graph from bilingual parallel corpora is to create a node for every phrase in the corpora, and two directed edges (i, j) and (j, i) for every aligned phrase pair i and j. Let H refer to the graph that is created in this manner. Such an approach is only tractable for small bilingual parallel corpora that would result in a small H, but not for large corpora containing millions of sentences, such as those described in Section 4.1. Therefore we approximate H with a graph H' that only contains nodes "near" to the node representing the query phrase. Specifically, we perform breadth-first search starting from the query node up to a depth d, or until the number of nodes visited in the search has reached a maximum of n nodes. Some nodes at the periphery of H' have edges to nodes that are not in H' but are in H. For a periphery node j that has edges to nodes $j_1, \ldots, j_k$ outside H', we create a "dummy" node a, and replace edges $(j, j_1), \ldots, (j, j_k)$ with a single edge (j, a) with weight $\sum_{x=1}^{k} W_{j,j_x}$. We also add edges (a, j) and (a, a) (each with a heuristic weight of 0.5). The dummy nodes and their edges approximate the transition probabilities at H''s periphery. Our empirical results show that this approximation works well in practice.

3.2 Graph Pruning

After H' is created, we run M independent length-T random walks on it starting from the query node to estimate the truncated hitting times of all nodes.

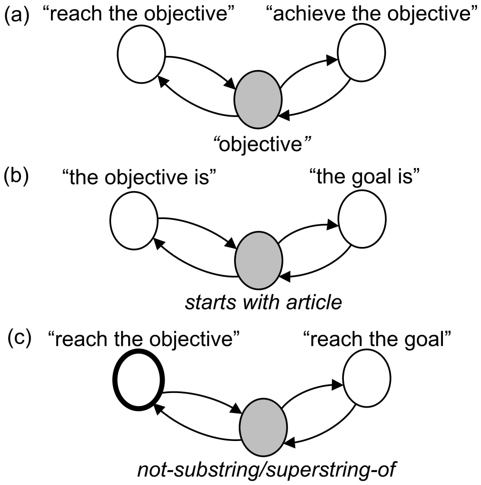

Figure 2: Feature nodes representing domain knowledge. Feature nodes are shaded. The bold node represents a query phrase. (a) n-gram nodes (b) "syntax" nodes (c) "not-substring/superstring-of" nodes.

A node in H' may have many outgoing edges, most of which may be due to spurious phrase alignments. For efficiency, and to reduce the noise due to spurious edges, we select among a node's top l outgoing edges with the highest transition probabilities, when deciding which node to visit next at each step of a random walk

For each random walk k, we record the first time that a node j is visited t_j^k. Using Equation 1, we estimate the truncated hitting time of each node. Then we remove nodes (and their associated edges) that are far from the query node, i.e., with times equal to T. Such nodes either are not visited in any of the random walks, or are always visited for the first time at step T.

3.3 Adding Domain Knowledge

Next we add nodes representing domain knowledge to the pruned graph. In this version of HTP, we implemented three types of *feature nodes*.

First, we have *n-gram nodes*. These nodes capture the domain knowledge that phrases containing many words in common are likely to be paraphrases. For each 1 to 4-gram that appears in English phrases, we create an n-gram node a. We add directed edges (a, j) and (j, a) if node j represents an English phrase containing n-gram a. For example, in Figure 2(a), "reach the objective" is connected to "objective" because it contains that unigram. Note that such nodes create short paths between nodes with many n-grams in common, thereby reducing the hitting times between them.

Second, we have *"syntax" nodes*, which represent syntactic classes of the start and end words of English phrases. We created classes such as *interrogatives* ("whose", "what", "where", etc.), *articles* ("the", "a", "an"), etc. For each class c, we create syntax nodes a_c and a'_c to respectively represent the conditions that a phrase begins and ends with a word in class c. Directed edges (a_c, j) and (j, a_c) are added if node j starts with a word in class c (similarly we add (a'_c, j) and (j, a'_c) if it ends with a word in class c). For example, in Figure 2(b), "the objective is" is linked to "starts with *article*" because it begins with "the". These syntax nodes allow HTP to capture broad commonalities about structural distribution, without requiring syntactic equivalence as in Callison-Burch 2008 (or the use of a parser).

Third, we have *"not-substring/superstring-of" nodes*. We observed that many English phrases (e.g., "reach the objective" and "reach the") that are superstrings or substrings of each other tend to be aligned to several shared non-English phrases in the bilingual parallel corpora used in our experiments. Most such English phrase pairs are not paraphrases, but they are linked by many short paths via their common aligned foreign phrase, and thus have small hitting times. To counteract this, we create a "not-substring/superstring-of" node a. The query node i is always connected to a via edges (i, a) and (a, i). We add edges (a, j) and (j, a) if English phrase j is not a substring or superstring of the query phrase (see Figure 2(c)).

With the addition of the above, each node representing an English phrase can have four kinds of outgoing edges: edges to foreign phrase nodes, and edges to the three kinds of feature nodes. Let $f_{phrase}, f_{ngram}, f_{syntax}, f_{substring}$ denote the distribution of transition probabilities among the four kinds of outgoing edges. Note that $f_{phrase} + f_{ngram} + f_{syntax} + f_{substring} = 1.0$. These values are user-specified or can be set with tuning data. An outgoing edge from English phrase node i that originally had weight (transition probability) W_{ij} will now have weight $W_{ij} \times f_{phrase}$. All k edges from i to n-gram nodes will have weight $\frac{f_{ngram}}{k}$. Likewise for edges to the other two kinds of feature nodes. Each of the k outgoing edges from a feature node is simply set to have a weight of $\frac{1}{k}$.

After adding the feature nodes, we again run M independent length-T random walks to estimate the truncated hitting times of the nodes, and return the English phrase nodes in order of increasing hitting times.

4 Experiments

We conducted experiments to investigate how HTP compares with the state of the art, and to evaluate the contributions of its components.

4.1 Dataset

We used the Europarl dataset (Koehn, 2005) for our experiments. This dataset contains English transcripts of the proceedings of the European Parliament, and their translations into 10 other European languages. In the dataset, there are about a million sentences per language, and English sentences are aligned with sentences in the other languages. Callison-Burch (2008) aligned English phrases with phrases in each of the other languages using Giza++ (Och and Ney, 2004). We used his English-foreign phrasal alignments which are publicly available on the web at http://ironman.jhu.edu/emnlp08.tar. In addition, we paired sentences of different non-English languages that correspond to the same English sentence, and aligned the phrases using 5 iterations of IBM model 1 in each direction, followed by 5 iterations of HMM alignment with paired training using the algorithm described in Liang et al. (2006). We further used the technique of Chen et al. (2009) to remove a phrase alignment F-G (where F and G are phrases in different foreign languages) if it was always aligned to different phrases in a third "bridge" foreign language. As observed by Chen et al., this helped to remove spurious alignments. We used Finnish as the bridge language; when either F or G is Finnish, we used Spanish as the bridge language; when F and G were Finnish and Spanish, we used English as the bridge language. In our experiments, we used phrases of length 1 to 4 of the following six languages: English, Danish, German, Spanish, Finnish, and Dutch. All the phrasal alignments between each pair of languages (15 in total) were used as input to HTP and its comparison systems. A small subset of the remaining phrase alignments were used for tuning parameters.

4.2 Systems

We compared HTP to the state-of-the-art SBP system (Callison-Burch, 2008). We also investigated the contribution of the feature nodes by running HTP without them. In addition, we ran HTP on a bipartite graph, i.e., one created from English-foreign phrase alignments only without any phrase alignments between foreign languages.

We used Callison-Burch (2008)'s implementation of SBP that is publicly available at http://ironman.jhu.edu/emnlp08.tar. SBP is based on BCB (Bannard and Callison- Burch, 2005) which computes the probability that English phrase E' is a paraphrase of E using the following formula:

$$P(E'|E) \approx \sum_{C \in \mathcal{C}} \sum_{F \in C} P(E'|F)P(F|E) \tag{2}$$

where $\mathcal{C}$ is set of bilingual parallel corpora, and F is a foreign language phrase. Representing phrases as nodes, and viewing $P(E'|F)$ and $P(F|E)$ as transition probabilities of edges (F, E') and (E, F), we see that BCB is summing over the transition probabilities of all length-two paths between E and E'. All E' paraphrases of E can then be ranked in order of decreasing probability as given by Equation 2. The SBP system modifies Equation 2 to incorporate syntactic information, thus:

$$P(E'|E) \approx \frac{1}{|\mathcal{C}|} \sum_{C \in \mathcal{C}} \sum_{F \in C} P(E'|F, syn_E))P(F|E, syn_E) \tag{3}$$

where syn_E is the syntax of phrase E, and $P(E'|F, syn_E)) = 0$ if E' is not of the same syntactic category. From Equation 3, we can see that SBP constrains E' to have the same syntactic structure as E. To obtain the syntactic structure of each English phrase, each English sentence in every parallel corpus has to be parsed to obtain its parse tree. An English phrase can have several syntactic structures because different parse trees can have the phrase as their leaves, and in each of these, SBP associates the phrase with all subtrees that have the phrase as their leaves. SBP thus offers several ways of choosing which syntactic structure a phrase should be associated with. In our experiments, we used the best performing method of averaging Equation 3 over all syntactic structures that E is associated with.

Table 2: Scoring Standards.

0	Clearly wrong; grammatically incorrect, or does not preserve meaning
1	Minor grammatical errors (e.g., subject-verb disagreement or wrong tense), or meaning is largely preserved but not completely
2	Totally correct; grammatically correct and meaning is preserved

4.3 Methodology

To evaluate performance, we used 33,216 English translations from the Linguistic Data Consortium's Multiple Translation Chinese (MTC) corpora (Huang et al., 2002). We randomly selected 100 1- to 4-grams that appeared in both Europarl and MTC sentences (excluding stop words, numbers, and phrases containing periods and commas). For each of those 100 phrases, we randomly selected a MTC sentence containing that phrase. We then replaced the phrase in the sentence with each paraphrase output by the systems, and evaluated the correctness of the paraphrase in the context of the sentence. We had two volunteers manually score the paraphrases on a 3-point scale (Table 2), using a simplified version of the scoring system used by Callison-Burch (2008). We deemed a paraphrase to be correct if it was scored 1 or 2, and wrong if it was scored 0. Evaluation was blind, and the paraphrases were presented randomly to the volunteers. The Kappa measure of inter-annotator agreement was 0.62, which indicates substantial agreement between the evaluators. We took the average score for each paraphrase.

The parameters used for HTP were as follows (see Table 1 for parameter descriptions): $d = 6, n = 50,000, m = 1,000,000, T = 10, \delta = 0.05, l = 20$, $f_{phrase} = 0.1, f_{ngram} = 0.1, f_{syntax} = 0.4, f_{substring} = 0.4$. ($\epsilon \leq 0.03$ with these values of n, m, T, and δ.)

Table 3: HTP vs. SBP.

	HTP	SBP
Correct top-1 paraphrases	71%	53%
Correct top-k paraphrases	54%	39%
Count of correct paraphrases	420	145
Correct paraphrases	43%	39%

Table 4: HTP vs. HTP without feature nodes.

	HTP	HTP-NoFeatNodes
Correct top-1 paraphrases	61%	41%
Correct top-k paraphrases	43%	29%
Count of correct paraphrases	420	283
Correct paraphrases	43%	29%

4.4 Results

HTP versus SBP. Comparison between HTP and SBP is complicated by the fact that the two systems did not output the same number of paraphrases for the 100 query phrases. HTP output paraphrases for all the query phrases, but SBP only did so for 49 query phrases. Of those 49 query phrases, HTP returned at least as many paraphrases as SBP, and for many it returned more.

To provide a fair comparison, we present results both for these 49 query phrases, and for all paraphrases returned by each of the systems. The upper half of Table 3 shows results for the 49 query phrases. The first row of Table 3 reports the percentage of top-1 paraphrases from this set that are correct, while the second row reports the percentage of correct top-k paraphrases from this set, where k is the number of queries returned by SBP, and is limited to at most 10. The value of k may differ for each query: if SBP and HTP return 3 and 20 paraphrases respectively, we only consider the top 3. On the third and fourth rows, we present the number of correct paraphrases and the percentage of correct paraphrases among the top 10 paraphrases returned by HTP for all 100 queries and the corresponding figures for the 49 queries for SBP. (When a system returned fewer than 10 paraphrases for a query, we consider all the paraphrases for that query.) It is evident from Table 3 that HTP consistently outperforms SBP: not only does it return more correct paraphrases overall (420 versus 145), it also has higher precision (43% versus 39%)

Table 5: HTP vs. HTP with bipartite graph.

	HTP	HTP-Bipartite
Correct top-1 paraphrases	62%	58%
Correct top-k paraphrases	46%	41%
Count of correct paraphrases	420	361
Correct paraphrases	43%	41%

HTP and SBP respectively took 48 and 468 seconds per query on a 3 GHz machine. The times are not directly comparable because the systems are implemented in different languages (HTP in C# and SBP in Java), and use different data structures.

HTP without Feature Nodes. Both HTP and HTP minus feature nodes output paraphrases for each of the 100 query phrases. Table 4 compares performance in the same manner as in Table 3, except that the "top-1" and "top-k" results are over all 100 query phrases. We see that feature nodes boost HTP's performance, allowing HTP to return more correct paraphrases (420 versus 283), and at higher precision (43% versus 29%).

HTP with Bipartite Graph. Lastly, we investigate the contribution of alignments between foreign phrases to HTP's performance. HTP-Bipartite refers to HTP that is given a set consisting only of English-foreign phrase alignment as input. HTP-Bipartite does not return paraphrases for 5 query phrases. Thus, in Table 5, the "top-1" and "top-k" results are for the 95 query phrases for which both systems return paraphrases. From the better performance of HTP, we see that the foreign phrase alignments help in finding English paraphrases.

5 Related Work

Random walks and hitting times have been successfully applied to a variety of applications. Brand (2005) has used hitting times for collaborative filtering, in which product recommendations to users are made based on purchase history. In computer vision, hitting times have been used to determine object shape from silhouettes (Gorelick et al., 2004), and for image segmentation (Grady and Schwartz, 2006). In social network analysis, Liben-Nowell and Kleinberg (2003) have investigated the use of hitting times for predicting relationships between entities. Recently, Mei et al. (2008) have used the hitting times of nodes in a bipartite graph created from search engine query logs to find related queries. They used an iterative algorithm to compute the hitting time, which converges slowly on large graphs. In HTP, we have sought to obviate this issue by using *truncated* hitting time that can be computed efficiently by sampling random walks.

Several approaches have been proposed to learn paraphrases. Barzilay and Mckeown (2001) acquire paraphrases from a monolingual parallel corpus using a co-training algorithm. Their co-trained classifier determines whether two phrases are paraphrases of one another using their surrounding contexts. Lin and Pantel (2001) learn paraphrases using the distributional similarity of paths in dependency trees. Ibrahim et al. (2003) generalize syntactic paths in aligned monolingual sentence pairs in order to generate paraphrases. Pang et al. (2003) merge parse trees of monolingual sentence pairs, and then compress the merged tree into a word lattice that can subsequently be used to generate paraphrases. Recently, Zhao et al. (2008) used dependency parses to learn paraphrase patterns that include part-of-speech slots. In other recent work, Das and Smith (2009) use a generative model for paraphrase detection. Rather than outputing paraphrases of an input phrase, their system detects whether two input sentences are paraphrases of one another.

6 Conclusion and Future Work

We have introduced HTP, a novel approach based on random walks and hitting times for the learning of paraphrases from bilingual parallel corpora. HTP works by converting aligned phrases into a graph, and finding paraphrases that are "close" to a phrase of interest. Compared to previous approaches, HTP is able to find more paraphrases by traversing paths of lengths longer than 2; utilizes information in the edges between foreign phrase nodes; and allows domain knowledge to be easily incorporated. Empirical results show its effectiveness in learning new paraphrases.

As future work, we plan to learn the distribution of weights on edges to phrase nodes and feature nodes automatically from data, rather than tuning them manually, and to develop a probabilistic model supporting HTP. We intend also to apply HTP to learning paraphrases in languages other than English and investigate the impact of the learned paraphrases on resource-sparse machine translation systems.

Acknowledgments

This work was done while the first author was an intern at Microsoft Research. We would like to thank Xiaodong He, Jianfeng Gao, Chris Quirk, Kristina Toutanova, Bob Moore, and other members of the MSR NLP group, along with Dengyong Zhou (TMSN) for their insightful comments and assistance in the course of this project.

References

David Aldous and Jim Fill. 2001. *Reversible Markov Chains and Random Walks on Graphs*. http://www.stat.berkeley.edu/~aldous/RWG/book.html.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 597–604.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of HLT/NAACL*, pages 16–23.

Regina Barzilay and Kathleen McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the ACL*, pages 50–57.

Regina Barzilay, Kathleen McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 550–557.

Matthew Brand. 2005. A random walks perspective on maximizing satisfaction and profit. In *Proceedings of the 8th SIAM Conference on Optimization*.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of HLT/NAACL*, pages 17–24.

Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, pages 196–205.

Yu Chen, Martin Kay, and Andreas Eisele. 2009. Intersecting multilingual data for faster and better statistical translations. In *Proceedings of HLT/NAACL*.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference* *of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*.

Lena Gorelick, Meirav Galun, Eitan Sharon, Ronen Basri, and Achi Brandt. 2004. Shape representation and classification using the Poisson equation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*.

Leo Grady and Eric L. Schwartz. 2006. Isoperimetric graph partitioning for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:469–475.

Shudong Huang, David Graff, and George Doddington. 2002. Multiple-translation Chinese corpus. Linguistic Data Consortium, Philadelphia.

Ali Ibrahim, Boris Katz, and Jimmy Lin. 2003. Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the 2nd International Workshop on Paraphrasing*, pages 57–64.

Lidija Iordanskaja, Richard Kittredge, and Alain Polguère. 1991. Lexical selection and paraphrase in a meaning-text generation model. In Cécile L. Paris, William R. Swartout, and William C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*. Kluwer Academic.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of HLT/NAACL*, pages 455–462.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*.

Nils Lenke. 1994. Anticipating the reader's problems and the automatic generation of paraphrases. In *Proceedings of the 15th Conference on Computational Linguistics*, pages 319–323.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of HLT/NAACL*, pages 104–111.

David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the 12th International Conference on Information and Knowledge*, pages 556–559.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 323–328.

László Lovász. 1996. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szőnyi, editors, *Combinatorics, Paul Erdős is Eighty, Vol. 2*, pages 353–398.

Nitin Madnani, Necip Fazil Ayan, Philip Resnik, and Bonnie J. Dorr. 2007. Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*, pages 120–127.

Kathleen R. McKeown, Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. In *Proceedings of the 2nd International Conference on HLT Research*, pages 280–285.

Qiaozhu Mei, Dengyong Zhou, and Kenneth Church. 2008. Query suggestion using hitting time. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management*, pages 469–478.

Franz J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417–449.

Bo Pang, Kevin Knight, and Daniel Marcu. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of HLT/NAACL*, pages 102–109.

Chris Quirk, Chris Brockett, and William B. Dolan. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of EMNLP*, pages 142–149.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 41–47.

Stefan Reizler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of the 45th Annual Meeting of the ACL*.

Purnamrita Sarkar and Andrew W. Moore. 2007. A tractable approach to finding closest truncated-commute-time neighbors in large graphs. In *Proceedings of the 23th Conference on Uncertainty in Artificial Intelligence*.

Purnamrita Sarkar, Andrew W. Moore, and Amit Prakash. 2008. Fast incremental proximity search in large graphs. In *Proceedings of the 25th International Conference on Machine Learning*.

Manfred Stede. 1999. *Lexical Semantics and Knowledge Representation in Multilingual Text Generation*. Kluwer Academic Publishers.

Shiqi Zhao, Haifeng Wang, Ting Liu, and Sheng Li. 2008. Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of ACL*.