ParaQuery: Making Sense of Paraphrase Collections

Lili Kotlerman
Bar-Ilan University
Israel
lili.dav@gmail.com

Nitin Madnani and Aoife Cahill
Educational Testing Service
Princeton, NJ, USA
{nmadnani,acahill}@ets.org

Abstract

Pivoting on bilingual parallel corpora is a popular approach for paraphrase acquisition. Although such pivoted paraphrase collections have been successfully used to improve the performance of several different NLP applications, it is still difficult to get an intrinsic estimate of the quality and coverage of the paraphrases contained in these collections. We present *ParaQuery*, a tool that helps a user interactively explore and characterize a given pivoted paraphrase collection, analyze its utility for a particular domain, and compare it to other popular lexical similarity resources – all within a single interface.

1 Introduction

Paraphrases are widely used in many Natural Language Processing (NLP) tasks, such as information retrieval, question answering, recognizing textual entailment, text simplification etc. For example, a question answering system facing a question "*Who invented bifocals and lightning rods?*" could retrieve the correct answer from the text "*Benjamin Franklin invented strike termination devices and bifocal reading glasses*" given the information that "*bifocal reading glasses*" is a paraphrase of "*bifocals*" and "*strike termination devices*" is a paraphrase of "*lightning rods*".

There are numerous approaches for automatically extracting paraphrases from text (Madnani and Dorr, 2010). We focus on generating paraphrases by pivoting on bilingual parallel corpora as originally suggested by Bannard and Callison-Burch (2005). This technique operates by attempting to infer semantic equivalence between phrases in the same language by using a second language as a bridge. It builds on one of the initial steps used to train a phrase-based statistical machine translation system. Such systems rely on *phrase tables* – a tabulation of correspondences between phrases in the source language and phrases in the target language. These tables are usually extracted by inducing word alignments between sentence pairs in a parallel training corpus and then incrementally building longer phrasal correspondences from individual words and shorter phrases. Once such a tabulation of bilingual correspondences is available, correspondences between phrases in one language may be inferred simply by using the phrases in the other language as *pivots*, e.g., if both "*man*" and "*person*" correspond to "*personne*" in French, then they can be considered paraphrases. Each paraphrase pair (*rule*) in a pivoted paraphrase collection is defined by a *source* phrase e_1, the *target* phrase e_2 that has been inferred as its paraphrase, and a probability score $p(e_2|e_1)$ obtained from the probability values in the bilingual phrase table.[1]

Pivoted paraphrase collections have been successfully used in different NLP tasks including automated document summarization (Zhou et al., 2006), question answering (Riezler et al., 2007), and machine translation (Madnani, 2010). Yet, it is still difficult to get an estimate of the intrinsic quality and coverage of the paraphrases contained in these collections. To remedy this, we propose *ParaQuery* – a tool that can help explore and analyze pivoted paraphrase collections.

2 ParaQuery

In this section we first briefly describe how to set up *ParaQuery* (§2.1) and then demonstrate its use in detail for interactively exploring and characterizing a paraphrase collection, analyzing its utility for a particular domain, and comparing it with other word-similarity resources (§2.2). Detailed documentation will be included in the tool.

[1] There may be other values associated with each pair, but we ignore them for the purposes of this paper.

2.1 Setting up

ParaQuery operates on pivoted paraphrase collections and can accept collections generated using any set of tools that are preferred by the user, as long as the collection is stored in a pre-defined plain-text format containing the source and target phrases, the probability values, as well as information on pivots (optional but useful for pivot-driven analysis, as shown later). This format is commonly used in the machine translation and paraphrase generation community. In this paper, we adapt the Thrax and Joshua (Ganitkevitch et al., 2012) toolkits to generate a pivoted paraphrase collection using the English-French EuroParl parallel corpus, which we use as our example collection for demonstrating *ParaQuery*. Once a pivoted collection is generated, *ParaQuery* needs to convert it into an SQLite database against which queries can be run. This is done by issuing the *index* command at the *ParaQuery* command-line interface (described in §2.2.1).

2.2 Exploration and Analysis

In order to provide meaningful exploration and analysis, we studied various scenarios in which paraphrase collections are used, and found that the following issues typically interest the developers and users of such collections:

1. Semantic relations between the paraphrases in the collection (e.g. synonymy, hyponymy) and their frequency.
2. The frequency of inaccurate paraphrases, possible ways of de-noising the collection, and the meaningfulness of scores (better paraphrases should be scored higher).
3. The utility of the collection for a specific domain, i.e. whether domain terms of interest are present in the collection.
4. Comparison of different collections based on the above dimensions.

We note that paraphrase collections are used in many tasks with different acceptability thresholds for semantic relations, noisy paraphrases etc. We do not intend to provide an exhaustive judgment of paraphrase quality, but instead allow users to characterize a collection, enabling an analysis of the aforesaid issues and providing information for them to decide whether a given collection is suitable for their specific task and/or domain.

2.2.1 Command line interface

ParaQuery allows interactive exploration and analysis via a simple command line interface, by processing user issued *queries* such as:

show **<*query*>:** display the rules which satisfy the conditions of the given *query*.

show count **<*query*>:** display the number of such rules.

explain **<*query*>:** display information about the pivots which yielded each of these rules.

analyze **<*query*>:** display statistics about these rules and save a report to an output file.

The following information is stored in the SQLite database for each paraphrase rule:[2]

- The source and the target phrases, and the probability score of the rule.
- Are the source and the target identical?
- Do the source and the target have the same part of speech?[3]
- Length of the source and the target, and the difference in their lengths.
- Number of pivots and the list of pivots.
- Are both the source and the target found in WordNet (WN)? If yes, the WN relation between them (synonym, derivation, hypernym, hyponym, co-hyponym, antonym, meronym, holonym, pertainym) or the minimal distance, if they are not connected directly.

Therefore, all of the above can be used, alone or in combination, to constrain the queries and define the rule(s) of interest. Figure 1 presents simple queries processed by the *show* command: the first query displays top-scoring rules with "*man*" as their source phrase, while the second adds restriction on the rules' score. By default, the tool displays the 10 best-scoring rules per query, but this limit can be changed as shown. For each rule, the corresponding score and semantic relation/distance is displayed.

[2] Although some of this information is available in the paraphrase collection that was indexed, the remaining is automatically computed and injected into the database during the indexing process. Indexing the French-pivoted paraphrase collection (containing 3,633,015 paraphrase rules) used in this paper took about 6 hours.

[3] We use the simple parts of speech provided by WordNet (nouns, verbs, adjectives and adverbs).

The queries provide a flexible way to define and work with the rule set of interest, starting from filtering low-scoring rules till extracting specific semantic relations or constraining on the number of pivots. Figure 2 presents additional examples of queries. The tool also enables filtering out target terms with a recurrent lemma, as illustrated in the same figure. Note that *ParaQuery* also contains a batch mode (in addition to the interactive mode illustrated so far) to automatically extract the output for a set of queries contained in a batch script.

```
query> set limit 5
query> show source = "man"

  man => men     [0.02669] [synonym]
  man => people  [0.01404] [meronym]
  man => person  [0.01228] [hypernym]
  man => rights  [0.008373] [WN distance=6]
  man => citizen [0.005146] [WN distance=3]

query> show source = "man" and prob < 0.001

  man => public       [0.0005019] [WN distance=3]
  man => anybody      [0.0004759] [not in WN]
  man => a person     [0.0004182] [not in WN]
  man => troops       [0.0002291] [WN distance=5]
  man => our citizens [0.0002023] [not in WN]
```

Figure 1: Examples of the *show* command and the probability constraint.

2.2.2 Analyzing pivot information

It is well known that pivoted paraphrase collections contain a lot of noisy rules. To understand the origins of such rules, an *explain* query can be used, which displays the pivots that yielded each paraphrase rule, and the probability share of each pivot in the final probability score. Figure 3 shows an example of this command.

We see that noisy rules can originate from stop-word pivots, e.g. "*l*'". It is common to filter rules containing stop-words, yet perhaps it is also important to exclude stop-word pivots, which was never considered in the past. We can use *ParaQuery* to further explore whether discarding stop-word pivots is a good idea. Figure 4 presents a more complex query showing paraphrase rules that were extracted via a single pivot "*l*'". We see that the top 5 such rules are indeed noisy, indicating that perhaps all of the 5,360 rules satisfying the query can be filtered out.

2.2.3 Analysis of rule sets

In order to provide an overall analysis of a rule set or a complete collection, *ParaQuery* includes the

```
query> show source = "man" and relation = "hypernym"

  man => person      [0.01228] [hypernym]
  man => individual  [0.003127] [hypernym]
  man => persons     [8.839e-05] [hypernym]
  man => individuals [6.799e-05] [hypernym]

query> set unique_tgt on
query> show source = "man" and relation = "hypernym"

  man => person     [0.01228] [hypernym]
  man => individual [0.003127] [hypernym]

query> show source = "man" and distance = 4

  man => member      [6.534e-05] [WN distance=4]
  man => businessmen [6.249e-05] [WN distance=4]
  man => union       [5.766e-05] [WN distance=4]
  man => year        [1.685e-05] [WN distance=4]
  man => need        [3.17e-06] [WN distance=4]
```

Figure 2: Restricting the output of the *show* command using WordNet relations and distance, and the unique lemma constraint.

```
query> explain source = "man" and prob < 0.01

  man => rights     [0.008373] [WN distance=6]
    1.  homme : 0.005095152449060402
    2.  l' homme : 0.0032782141291460873

  man => citizen    [0.005146] [WN distance=3]
    1.  citoyen : 0.005146163785838685

  man => male       [0.004446] [synonym]
    1.  homme : 0.003821364336795302
    2.  hommes : 6.249019761606025E-4

  man => s          [0.003421] [WN distance=5]
    1.  l : 0.00341807739914768
    2.  ’ : 3.328130356209791E-6

  man => politician [0.003397] [WN distance=3]
    1.  homme : 0.0033967682993736007

  man => individual [0.003127] [hypernym]
    1.  homme : 0.0021229801871085
    2.  personne : 9.042530177822003E-4
    3.  citoyen : 6.741699282758096E-5
    4.  humain : 3.232734671281396E-5
```

Figure 3: An example of the *explain* command.

analyze command. Figure 5 shows the typical information provided by this command. In addition, a report is generated to a file, including the analysis information for the whole rule set and for its three parts: *top*, *middle* and *bottom*, as defined by the scores of the rules in the set. The output to the file is more detailed and expands on the information presented in Figure 5. For example, it also includes, for each part, rule samples and score distributions for each semantic relation and different WordNet distances.

The information contained in the report can be

```
query> set limit 5
query> show source = "*" and pivots = 1 and pivots include "l"

  evropa => s [0.2724] [not in WN]
     tym => s [0.2043] [not in WN]
      za => s [0.2043] [not in WN]
    gaat => s [0.1135] [not in WN]
     tak => s [0.1135] [not in WN]

query> show count source = "*" and pivots = 1 and pivots include "l"
5360
```

Figure 4: Exploring French stop-word pivots using the *pivots* condition of the *show* command.

```
query> analyze source = "man*" and prob > 0.1
Retrieving rules from the database ... found  155 paraphrase rules.
Analyzing...
10%... 20%... 30%... 40%... 50%... 60%... 70%... 80%... 90%...

Random rule sample:
--------------------
  mandate for negotiation => negotiating mandate  [0.351307189542]
  manifestly => obviously  [0.157373877673]
  manifestations => events  [0.116528555867]

Statistics for the 155 rule(s):
-----------------------------------
  Average number of pivots: 2.10322580645

  Results of WordNet analysis based on 20 rule(s) (12.9% of the 155 rule(s)):
      SYNONYM: 7 rule(s) (35.0%):
      DERIVATION: 2 rule(s) (10.0%):
      HYPERNYM: 2 rule(s) (10.0%):
      CO-HYPONYM: 1 rule(s) (5.0%):
      UNDEFINED RELATION: 6 rule(s) (30.0%):
      HYPONYM: 1 rule(s) (5.0%):
      PERTAINYM: 1 rule(s) (5.0%):

  Average WordNet distance for rules corresponding to UNDEFINED RELATION: 4.66666666667

  The number of unique source sides is: 134

  The average number of target sides per source is: 1.15671641791

   The average number of 'NOT IN WN' targets per source is: 1.00746268657
   The average number of 'SYNONYM' targets per source is: 0.0522388059701
   The average number of 'DERIVATION' targets per source is: 0.0149253731343
   The average number of 'HYPERNYM' targets per source is: 0.0149253731343
```

Figure 5: An example of the *analyze* command (full output not shown for space reasons).

TOP	BOTTOM
finest ⇒ better	approach ⇒ el
outdoors ⇒ external	effect ⇒ parliament
unsettled ⇒ unstable	comment ⇒ speak up
intelligentsia ⇒ intelligence	propose ⇒ allotted
caretaker ⇒ provisional	prevent ⇒ aimed
luckily ⇒ happily	energy ⇒ subject matter

Table 1: A random sample of *undefined relation* rules from our collection's top and bottom parts.

easily used for generating graphs and tables. For example, Figure 6 shows the distribution of semantic relations in the three parts of our example paraphrase collection. The figure characterizes the collection in terms of semantic relations it contains and illustrates the fact that the scores agree with their desired behavior: (1) the collection's top-scoring part contains significantly more synonyms than its middle and bottom parts, (2) similar trends hold for derivations and hypernyms, which are more suitable for paraphrasing than co-hyponyms and other relations not defined in WordNet (we refer to these relations as *undefined relations*), (3) such undefined relations have the highest frequency in the collection's bottom part, and are least frequent in its top part. Among other conclusions, the figure shows, that discarding the lower-scoring middle and bottom parts of the collection would allow retaining almost all the synonyms and derivations, while filtering out most of the co-hyponyms and a considerable number of undefined relations.

Yet from Figure 6 we see that undefined relations constitute the majority of the rules in the collection. To better understand this, random rule samples provided in the analysis output can be used, as shown in Table 1. From this table, we see that the top-part rules are indeed mostly valid for paraphrasing, unlike the noisy bottom-part rules. The score distributions reported as part of the analysis can be used to further explore the collection and set sound thresholds suitable for different tasks and needs.

2.2.4 Analysis of domain utility

One of the frequent questions of interest is whether a given collection is suitable for a specific domain. To answer this question, *ParaQuery* allows the user to run the analysis from §2.2.3 over rules whose source phrases belong to a specific domain, by means of the *analyze* <*query*> *using* <*file*> command. The *file* can hold either a list of domain terms or a representative domain text, from which frequent terms and term collocations will be automatically extracted, presented to the user, and utilized for analysis. The analysis includes the coverage of the domain terms in the paraphrase collection, and can also be restricted to top-K rules per source term, a common practice in many NLP applications. We do not show an example of this command due to space considerations.

2.2.5 Comparison with other collections

The output of the *analyze* command can also be used to compare different collections, either in general or for a given domain. Although *ParaQuery* is designed for pivoted paraphrase collections, it allows comparing them to non-pivoted paraphrase collections as well. Next we present an example of such a comparative study, performed using *ParaQuery* via several *analyze* commands.

Table 2 compares three different collections: the French pivoted paraphrase collection, a distributional similarity resource (Kotlerman et al., 2010) and a Wikipedia-based resource (Shnarch et al., 2009). The table shows the collection sizes, as well as the number of different (unique) source phrases in them and, correspondingly, the average number of target phrases per source. From the table we can see that the distributional similarity resource contains a lot of general language terms found in WordNet, while the Wikipedia resource includes only a small amount of such terms. A sample of rules from the Wikipedia collection explains this behavior, e.g. '*Yamaha SR500* ⇒ *motorcycle*'. The table provides helpful information to decide which collection is (more) suitable for specific tasks, such as paraphrase recognition and generation, query expansion, automatic generation of training data for different supervised tasks, etc.

3 Conclusions and Future Work

We presented *ParaQuery*—a tool for interactive exploration and analysis of pivoted paraphrase collections—and showed that it can be used to estimate the intrinsic quality and coverage of the paraphrases contained in these collections, a task that is still somewhat difficult. *ParaQuery* can also be used to answer the questions that users of such collections are most interested in. We plan to release *ParaQuery* under an open-source license, including our code for generating paraphrase collections that can then be indexed and analyzed by

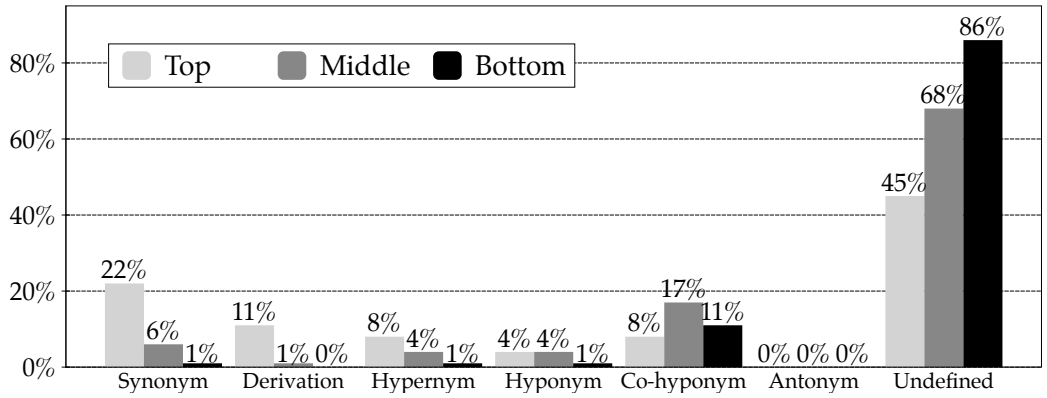

Figure 6: Distribution of semantic relations in the top, middle and bottom parts of the example collection. The parts are defined by binning the scores of the rules in the collection.

Collection	Size (rules)	In WordNet	Unique Src	Avg. Tgts per Src	d_{avg} for UR
Pivoted (FR)	3,633,015	757,994 (21%)	188,898	16.064	2.567
Dist.Sim.	7,298,321	3,252,967 (45%)	113,444	64.334	6.043
Wikipedia	7,880,962	295,161 (4%)	2,727,362	2.890	8.556

Table 2: Comparing the French-pivoted paraphrase collection to distributional-similarity based and Wikipedia-based similarity collections, in terms of total size, percentage of rules in WordNet, number of unique source phrases, average number of target phrases per source phrase, and the average WordNet distance between the two sides of the *undefined relation* (UR) rules.

ParaQuery. We also plan to include pre-generated paraphrase collections in the release so that users of *ParaQuery* can use it immediately.

In the future, we plan to use this tool for analyzing the nature of pivoted paraphrases. The quality and coverage of these paraphrases is known to depend on several factors, including (a) the genre of the bilingual corpus, (b) the word-alignment algorithm used during bilingual training, and (c) the pivot language itself. However, there have been no explicit studies designed to measure such variations. We believe that *ParaQuery* is perfectly suited to conducting such studies and moving the field of automated paraphrase generation forward.

Acknowledgments

This work was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT).

References

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of ACL*, pages 597–604.

Juri Ganitkevitch, Yuan Cao, Jonathan Weese, Matt Post, and Chris Callison-Burch. 2012. Joshua 4.0: Packing, PRO, and Paraphrases. In *Proceedings of WMT*, pages 283–291.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional Distributional Similarity for Lexical Inference. *Natural Language Engineering*, 16(4):359–389.

Nitin Madnani and Bonnie J. Dorr. 2010. Generating Phrasal and Sentential Paraphrases: A Survey of Data-driven Methods. *Computational Linguistics*, 36(3):341–387.

Nitin Madnani. 2010. *The Circle of Meaning: From Translation to Paraphrasing and Back*. Ph.D. thesis, Department of Computer Science, University of Maryland College Park.

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu O. Mittal, and Yi Liu. 2007. Statistical Machine Translation for Query Expansion in Answer Retrieval. In *Proceedings of ACL*, pages 464–471.

Eyal Shnarch, Libby Barak, and Ido Dagan. 2009. Extracting lexical reference rules from Wikipedia. In *Proceedings of ACL-IJCNLP*, pages 450–458.

Liang Zhou, Chin-Yew Lin, Dragos Stefan Muntenau, and Eduard Hovy. 2006. ParaEval: Using Paraphrases to Evaluate Summaries Automatically. In *Proceedings of HLT-NAACL*, pages 447–454.