Journal of Artificial Intelligence Research 38 (2010) 135-187 Submitted 12/09; published 05/10

A Survey of Paraphrasing and Textual Entailment Methods

Ion Androutsopoulos ION@AUEB.GR
Prodromos Malakasiotis RULLLER@AUEB.GR
Department of Informatics
Athens University of Economics and Business
Patission 76, GR-104 34 Athens, Greece

Abstract

Paraphrasing methods recognize, generate, or extract phrases, sentences, or longer natural language expressions that convey almost the same information. Textual entailment methods, on the other hand, recognize, generate, or extract pairs of natural language expressions, such that a human who reads (and trusts) the first element of a pair would most likely infer that the other element is also true. Paraphrasing can be seen as bidirectional textual entailment and methods from the two areas are often similar. Both kinds of methods are useful, at least in principle, in a wide range of natural language processing applications, including question answering, summarization, text generation, and machine translation. We summarize key ideas from the two areas by considering in turn recognition, generation, and extraction methods, also pointing to prominent articles and resources.

1. Introduction

This article is a survey of computational methods for paraphrasing and textual entailment. Paraphrasing methods recognize, generate, or extract (e.g., from corpora) paraphrases, meaning phrases, sentences, or longer texts that convey the same, or almost the same information. For example, (1) and (2) are paraphrases. Most people would also accept (3) as a paraphrase of (1) and (2), though it could be argued that in (3) the construction of the bridge has not necessarily been completed, unlike (1) and (2).[1] Such fine distinctions, however, are usually ignored in paraphrasing and textual entailment work, which is why we say that paraphrases may convey *almost* the same information.

(1) Wonderworks Ltd. constructed the new bridge.
(2) The new bridge was constructed by Wonderworks Ltd.
(3) Wonderworks Ltd. is the constructor of the new bridge.

Paraphrasing methods may also operate on *templates* of natural language expressions, like (4)–(6); here the slots X and Y can be filled in with arbitrary noun phrases. Templates specified at the syntactic or semantic level may also be used, where the slot fillers may be required to have particular syntactic relations (e.g., verb-object) to other words or constituents, or to satisfy semantic constraints (e.g., requiring Y to denote a book).

(4) X wrote Y.
(5) Y was written by X.
(6) X is the writer of Y.

[1] Readers familiar with tense and aspect theories will have recognized that (1)–(3) involve an "accomplishment" of Vendler's (1967) taxonomy. The accomplishment's completion point is not necessarily reached in (3), unlike (1)–(2).

Textual entailment methods, on the other hand, recognize, generate, or extract pairs $\langle T,H \rangle$ of natural language expressions, such that a human who reads (and trusts) T would infer that H is most likely also true (Dagan, Glickman, & Magnini, 2006). For example, (7) textually entails (8), but (9) does not textually entail (10).[2]

(7) The drugs that slow down Alzheimer's disease work best the earlier you administer them.
(8) Alzheimer's disease can be slowed down using drugs.
(9) Drew Walker, Tayside's public health director, said: "It is important to stress that this is not a confirmed case of rabies."
(10) A case of rabies was confirmed.

As in paraphrasing, textual entailment methods may operate on templates. For example, in a discourse about painters, composers, and their work, (11) textually entails (12), for any noun phrases X and Y. However, (12) does not textually entail (11), when Y denotes a symphony composed by X. If we require textual entailment between templates to hold for all possible slot fillers, then (11) textually entails (12) in our example's discourse, but the reverse does not hold.

(11) X painted Y.
(12) Y is the work of X.

In general, we cannot judge if two natural language expressions are paraphrases or a correct textual entailment pair without selecting particular *readings* of the expressions, among those that may be possible due to multiple word senses, syntactic ambiguities etc. For example, (13) textually entails (14) with the financial sense of "bank", but not when (13) refers to the bank of a river.

(13) A bomb exploded near the French bank.
(14) A bomb exploded near a building.

One possibility, then, is to examine the language expressions (or templates) only in particular contexts that make their intended readings clear. Alternatively, we may want to treat as correct any textual entailment pair $\langle T,H \rangle$ for which there are *possible* readings of T and H, such that a human who reads T would infer that H is most likely also true; then, if a system reports that (13) textually entails (14), its response is to be counted as correct, regardless of the intended sense of "bank". Similarly, paraphrases would have *possible* readings conveying almost the same information.

The lexical substitution task of SEMEVAL (McCarthy & Navigli, 2009), where systems are required to find an appropriate substitute for a particular word in the context of a given sentence, can be seen as a special case of paraphrasing or textual entailment, restricted to pairs of words. SEMEVAL's task, however, includes the requirement that it must be possible to use the two words (original and replacement) in exactly the same context. In a similar manner, one could adopt a stricter definition of paraphrases, which would require them not only to have the same (or almost the same) meaning, but also to be expressions that can be used interchangeably in grammatical sentences. In that case, although (15) and (16) are paraphrases, their underlined parts are not, because they cannot be swapped in the two sentences; the resulting sentences would be ungrammatical.

(15) <u>Edison invented the light bulb in 1879</u>, providing a long lasting source of light.
(16) <u>Edison's invention of the light bulb in 1879</u> provided a long lasting source of light.

A similar stricter definition of textual entailment would impose the additional requirement that H and T can replace each other in grammatical sentences.

[2] Simplified examples from RTE-2 (Bar-Haim, Dagan, Dolan, Ferro, Giampiccolo, Magnini, & Szpektor, 2006).

1.1 Possible Applications of Paraphrasing and Textual Entailment Methods

The natural language expressions that paraphrasing and textual entailment methods consider are not always statements. In fact, many of these methods were developed having question answering (QA) systems in mind. In QA systems for document collections (Voorhees, 2001; Pasca, 2003; Harabagiu & Moldovan, 2003; Mollá & Vicedo, 2007), a question may be phrased differently than in a document that contains the answer, and taking such variations into account can improve system performance significantly (Harabagiu, Maiorano, & Pasca, 2003; Duboue & Chu-Carroll, 2006; Harabagiu & Hickl, 2006; Riezler, Vasserman, Tsochantaridis, Mittal, & Liu, 2007). For example, a QA system may retrieve relevant documents or passages, using the input question as a query to an information retrieval or Web search engine (Baeza-Yates & Ribeiro-Neto, 1999; Manning, 2008), and then check if any of the retrieved texts textually entails a candidate answer (Moldovan & Rus, 2001; Duclaye, Yvon, & Collin, 2003).[3] If the input question is (17) and the search engine returns passage (18), the system may check if (18) textually entails any of the candidate answers of (19), where we have replaced the interrogative "who" of (17) with all the expressions of (18) that a named entity recognizer (Bikel, Schwartz, & Weischedel, 1999; Sekine & Ranchhod, 2009) would ideally have recognized as person names.[4]

(17) Who sculpted the Doryphoros?

(18) The Doryphoros is one of the best known Greek sculptures of the classical era in Western Art. The Greek sculptor *Polykleitos* designed this work as an example of the "canon" or "rule", showing the perfectly harmonious and balanced proportions of the human body in the sculpted form. The sculpture was known through the Roman marble replica found in Herculaneum and conserved in the Naples National Archaeological Museum, but, according to *Francis Haskell* and *Nicholas Penny*, early connoisseurs passed it by in the royal Bourbon collection at Naples without notable comment.

(19) Polykleitos/Francis Haskell/Nicholas Penny sculpted the Doryphoros.

The input question may also be paraphrased, to allow more, potentially relevant passages to be obtained. Question paraphrasing is also useful when mapping user questions to lists of frequently asked questions (FAQs) that are accompanied by their answers (Tomuro, 2003); and natural language interfaces to databases often generate question paraphrases to allow users to understand if their queries have been understood (McKeown, 1983; Androutsopoulos, Ritchie, & Thanisch, 1995).

Paraphrasing and textual entailment methods are also useful in several other natural language processing applications. In text summarization (Mani, 2001; Hovy, 2003), for example, an important processing stage is typically sentence extraction, which identifies the most important sentences of the texts to be summarized. During that stage, especially when generating a single summary from several documents (Barzilay & McKeown, 2005), it is important to avoid selecting sentences (e.g., from different news articles about the same event) that convey the same information (paraphrases) as other sentences that have already been selected, or sentences whose information follows from other already selected sentences (textual entailment).

Sentence compression (Knight & Marcu, 2002; McDonald, 2006; Cohn & Lapata, 2008; Clarke & Lapata, 2008; Cohn & Lapata, 2009; Galanis & Androutsopoulos, 2010), often also a processing stage of text summarization, can be seen as a special case of sentence paraphrasing, as suggested by

[3] Culicover (1968) discussed different types of paraphrasing and entailment, and proposed the earliest computational treatment of paraphrasing and textual entailment that we are aware of, with the goal of retrieving passages of texts that answer natural language queries. We thank one of the anonymous reviewers for pointing us to Culicover's work.

[4] Passage (18) is based on Wikipedia's page for Doryphoros.

Zhao et al. (2009), with the additional constraint that the resulting sentence must be shorter than the original one and still grammatical; for example, a sentence matching (5) or (6) could be shortened by converting it to a paraphrase of the form of (4). Most sentence compression work, however, allows less important information of the original sentence to be discarded. Hence, the resulting sentence is entailed by, it is not necessarily a paraphrase of the original one. In the following example, (21) is a compressed form of (20) produced by a human.[5]

(20) Mother Catherine, 82, the mother superior, will attend the hearing on Friday, he said.
(21) Mother Catherine, 82, the mother superior, will attend.

When the compressed sentence is not necessarily a paraphrase of the original one, we may first produce (grammatical) candidate compressions that are textually entailed by the original sentence; hence, a mechanism to generate textually entailed sentences is useful. Additional mechanisms are needed, however, to rank the candidates depending on the space they save and the degree to which they maintain important information; we do not discuss additional mechanisms of this kind.

Information extraction systems (Grishman, 2003; Moens, 2006) often rely on manually or automatically crafted patterns (Muslea, 1999) to locate text snippets that report particular types of events and to identify the entities involved; for example, patterns like (22)–(24), or similar patterns operating on syntax trees, possibly with additional semantic constraints, might be used to locate snippets referring to bombing incidents and identify their targets. Paraphrasing or textual entailment methods can be used to generate additional semantically equivalent extraction patterns (in the case of paraphrasing) or patterns that textually entail the original ones (Shinyama & Sekine, 2003).

(22) *X* was bombed
(23) bomb exploded near *X*
(24) explosion destroyed *X*

In machine translation (Koehn, 2009), ideas from paraphrasing and textual entailment research have been embedded in measures and processes that automatically evaluate machine-generated translations against human-authored ones that may use different phrasings (Lepage & Denoual, 2005; Zhou, Lin, & Hovy, 2006a; Kauchak & Barzilay, 2006; Padó, Galley, Jurafsky, & Manning, 2009); we return to this issue in following sections. Paraphrasing methods have also been used to automatically generate additional reference translations from human-authored ones when training machine translation systems (Madnani, Ayan, Resnik, & Dorr, 2007). Finally, paraphrasing and textual entailment methods have been employed to allow machine translation systems to cope with source language words and longer phrases that have not been encountered in training corpora (Zhang & Yamamoto, 2005; Callison-Burch, Koehn, & Osborne, 2006a; Marton, Callison-Burch, & Resnik, 2009; Mirkin, Specia, Cancedda, Dagan, Dymetman, & Szpektor, 2009b). To use an example of Mirkin et al. (2009b), a phrase-based machine translation system that has never encountered the expression "file a lawsuit" during its training, but which knows that pattern (25) textually entails (26), may be able to produce a more acceptable translation by converting (27) to (28), and then translating (28). Some information would be lost in the translation, because (28) is not a paraphrase of (27), but the translation may still be preferable to the outcome of translating directly (27).

(25) *X* filed a lawsuit against *Y* for *Z*.

[5] Example from Clarke et al.s paper, "Written News Compression Corpus (Clarke & Lapata, 2008); see Appendix A.

(26) X accused Y of Z.
(27) Cisco filed a lawsuit against Apple for patent violation.
(28) Cisco accused Apple of patent violation.

In natural language generation (Reiter & Dale, 2000; Bateman & Zock, 2003), for example when producing texts describing the entities of a formal ontology (O'Donnell, Mellish, Oberlander, & Knott, 2001; Androutsopoulos, Oberlander, & Karkaletsis, 2007), paraphrasing can be used to avoid repeating the same phrasings (e.g., when expressing properties of similar entities), or to produce alternative expressions that improve text coherence, adhere to writing style (e.g., avoid passives), or satisfy other constraints (Power & Scott, 2005). Among other possible applications, paraphrasing and textual entailment methods can be employed to simplify texts, for example by replacing specialized (e.g., medical) terms with expressions non-experts can understand (Elhadad & Sutaria, 2007; Deléger & Zweigenbaum, 2009), and to automatically score student answers against reference answers (Nielsen, Ward, & Martin, 2009).

1.2 The Relation of Paraphrasing and Textual Entailment to Logical Entailment

If we represent the meanings of natural language expressions by logical formulae, for example in first-order predicate logic, we may think of textual entailment and paraphrasing in terms of logical entailment ($\models$). If the logical meaning representations of T and H are ϕ_T and ϕ_H, then $\langle T, H \rangle$ is a correct textual entailment pair if and only if $(\phi_T \wedge B) \models \phi_H$; B is a knowledge base, for simplicity assumed here to have the form of a single conjunctive formula, which contains meaning postulates (Carnap, 1952) and other knowledge assumed to be shared by all language users.[6] Let us consider the example below, where logical terms starting with capital letters are constants; we assume that different word senses would give rise to different predicate symbols. Let us also assume that B contains only ψ. Then $(\phi_T \wedge \psi) \models \phi_H$ holds, i.e., ϕ_H is true for any interpretation (e.g., model-theoretic) of constants, predicate names and other domain-dependent atomic symbols, for which ϕ_T and ψ both hold. A sound and complete automated reasoner (e.g., based on resolution, in the case of first-order predicate logic) could be used to confirm that the logical entailment holds. Hence, T textually entails H, assuming again that the meaning postulate ψ is available. The reverse, however, does not hold, i.e., $(\phi_H \wedge \psi) \not\models \phi_T$; the implication ($\Rightarrow$) of ψ would have to be made bidirectional ($\Leftrightarrow$) for the reverse to hold.

T : Leonardo da Vinci painted the Mona Lisa.
ϕ_T : $isPainterOf(DaVinci, MonaLisa)$
H : Mona Lisa is the work of Leonardo da Vinci.
ϕ_H : $isWorkOf(MonaLisa, DaVinci)$
ψ : $\forall x \forall y \; isPainterOf(x,y) \Rightarrow isWorkOf(y,x)$

Similarly, if the logical meaning representations of T_1 and T_2 are ϕ_1 and ϕ_2, then T_1 is a paraphrase of T_2 iff $(\phi_1 \wedge B) \models \phi_2$ and $(\phi_2 \wedge B) \models \phi_1$, where again B contains meaning postulates and common sense knowledge. Ideally, sentences like (1)–(3) would be represented by the same formula, making it clear that they are paraphrases, regardless of the contents of B. Otherwise, it may

[6] Zaenen et al. (2005) provide examples showing that linguistic and world knowledge cannot often be separated.

sometimes be unclear if T_1 and T_2 should be considered paraphrases, because it may be unclear if some knowledge should be considered part of B.

Since natural language expressions are often ambiguous, especially out of context, we may again want to adopt looser definitions, so that T textually entails H iff there are *possible* readings of T and H, represented by ϕ_H and ϕ_T, such that $(\phi_T \wedge B) \models \phi_H$, and similarly for paraphrases. Thinking of textual entailment and paraphrasing in terms of logical entailment allows us to borrow notions and methods from logic. Indeed, some paraphrasing and textual entailment recognition methods map natural language expressions to logical formulae, and then examine if logical entailments hold. This is not, however, the only possible approach. Many other, if not most, methods currently operate on surface strings or syntactic representations, without mapping natural language expressions to formal meaning representations. Note, also, that in methods that map natural language to logical formulae, it is important to work with a form of logic that provides adequate support for logical entailment checks; full first-order predicate logic may be inappropriate, as it is semi-decidable.

To apply our logic-based definition of textual entailment, which was formulated for statements, to questions, let us use identical fresh constants (in effect, Skolem constants) across questions to represent the unknown entities the questions ask for; we mark such constants with question marks as subscripts, but in logical entailment checks they can be treated as ordinary constants. In the following example, the user asks H, and the system generates T. Assuming that the meaning postulate ψ is available in B, $(\phi_T \wedge B) \models \phi_H$, i.e., for any interpretation of the predicate symbols and constants, if $(\phi_T \wedge B)$ is true, then ϕ_H is necessarily also true. Hence, T textually entails H. In practice, this means that if the system manages to find an answer to T, perhaps because T's phrasing is closer to a sentence in a document collection, the same answer can be used to respond to H.

$$
\begin{array}{rcl}
T(generated) & : & \text{Who painted the Mona Lisa?} \\
\phi_T & : & \mathit{isAgent}(W_?) \wedge \mathit{isPainterOf}(W_?, \mathit{MonaLisa}) \\
H(asked) & : & \text{Whose work is the Mona Lisa?} \\
\phi_H & : & \mathit{isAgent}(W_?) \wedge \mathit{isWorkOf}(\mathit{MonaLisa}, W_?) \\
\psi & : & \forall x \forall y \; \mathit{isPainterOf}(x,y) \Rightarrow \mathit{isWorkOf}(y,x)
\end{array}
$$

A logic-based definition of question paraphrases can be formulated in a similar manner, as bidirectional logical entailment. Note also that logic-based paraphrasing and textual entailment methods may actually represent interrogatives as free variables, instead of fresh constants, and they may rely on unification to obtain their values (Moldovan & Rus, 2001; Rinaldi, Dowdall, Kaljurand, Hess, & Molla, 2003).

1.3 A Classification of Paraphrasing and Textual Entailment Methods

There have been six workshops on paraphrasing and/or textual entailment (Sato & Nakagawa, 2001; Inui & Hermjakob, 2003; Dolan & Dagan, 2005; Drass & Yamamoto, 2005; Sekine, Inui, Dagan, Dolan, Giampiccolo, & Magnini, 2007; Callison-Burch, Dagan, Manning, Pennacchiotti, & Zanzotto, 2009) in recent years.[7] The Recognizing Textual Entailment (RTE) challenges (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo, Magnini, Dagan, & Dolan, 2007; Giampiccolo, Dang,

[7]The proceedings of the five more recent workshops are available in the ACL Anthology.

Magnini, Dagan, & Dolan, 2008), currently in their fifth year, provide additional significant thrust.[8] Consequently, there is a large number of published articles, proposed methods, and resources related to paraphrasing and textual entailment.[9] A special issue on textual entailment was also recently published, and its editorial provides a brief overview of textual entailment methods (Dagan, Dolan, Magnini, & Roth, 2009).[10] To the best of our knowledge, however, the present article is the first extensive survey of paraphrasing and textual entailment.

To provide a clearer view of the different goals and assumptions of the methods that have been proposed, we classify them along two dimensions: whether they are *paraphrasing* or *textual entailment* methods; and whether they perform *recognition*, *generation*, or *extraction* of paraphrases or textual entailment pairs. These distinctions are not always clear in the literature, especially the distinctions along the second dimension, which we explain below. It is also possible to classify methods along other dimensions, for example depending on whether they operate on language expressions or templates; or whether they operate on phrases, sentences or longer texts.

The main input to a paraphrase or textual entailment *recognizer* is a pair of language expressions (or templates), possibly in particular contexts. The output is a judgement, possibly probabilistic, indicating whether or not the members of the input pair are paraphrases or a correct textual entailment pair; the judgements must agree as much as possible with those of humans. On the other hand, the main input to a paraphrase or textual entailment *generator* is a single language expression (or template) at a time, possibly in a particular context. The output is a set of paraphrases of the input, or a set of language expressions that entail or are entailed by the input; the output set must be as large as possible, but including as few errors as possible. In contrast, no particular language expressions or templates are provided to a paraphrase or textual entailment *extractor*. The main input in this case is a corpus, for example a monolingual corpus of parallel or comparable texts, such as different English translations of the same French novel, or clusters of multiple monolingual news articles, with the articles in each cluster reporting the same event. The system outputs pairs of paraphrases (possibly templates), or pairs of language expressions (or templates) that constitute correct textual entailment pairs, based on the evidence of the corpus; the goal is again to produce as many output pairs as possible, with as few errors as possible. Note that the boundaries between recognizers, generators, and extractors may not always be clear. For example, a paraphrase generator may invoke a paraphrase recognizer to filter out erroneous candidate paraphrases; and a recognizer or a generator may consult a collection of template pairs produced by an extractor.

We note that articles reporting actual applications of paraphrasing and textual entailment methods to larger systems (e.g., for QA, information extraction, machine translation, as discussed in Section 1.1) are currently relatively few, compared to the number of articles that propose new paraphrasing and textual entailment methods or that test them in vitro, despite the fact that articles of the second kind very often point to possible applications of the methods they propose. The relatively small number of application articles may be an indicator that paraphrasing and textual entailment methods are not used extensively in larger systems yet. We believe that this may be due to at least two reasons. First, the efficiency of the methods needs to be improved, which may require combining recognition, generation, and extraction methods, for example to iteratively produce more training data; we return to this point in following sections. Second, the literature on paraphrasing

[8] The RTE challenges were initially organized by the European PASCAL Network of Excellence, and subsequently as part of NIST's Text Analysis Conference.

[9] A textual entailment portal has been established, as part of ACL's wiki, to help organize all relevant material.

[10] The slides of Dagan, Roth, and Zazotto's ACL 2007 tutorial on textual entailment are also publicly available.

and textual entailment is vast, which makes it difficult for researchers working on larger systems to assimilate its key concepts and identify suitable methods. We hope that this article will help address the second problem, while also acting as an introduction that may help new researchers improve paraphrasing and textual entailment methods further.

In Sections 2, 3, and 4 below we consider in turn recognition, generation, and extraction methods for both paraphrasing and textual entailment. In each of the three sections, we attempt to identify and explain prominent ideas, pointing also to relevant articles and resources. In Section 5, we conclude and discuss some possible directions for future research. The URLs of all publicly available resources that we mention are listed in appendix A.

2. Paraphrase and Textual Entailment Recognition

Paraphrase and textual entailment recognizers judge whether or not two given language expressions (or templates) constitute paraphrases or a correct textual entailment pair. Different methods may operate at different levels of representation of the input expressions; for example, they may treat the input expressions simply as surface strings, they may operate on syntactic or semantic representations of the input expressions, or on representations combining information from different levels.

2.1 Logic-based Approaches to Recognition

One possibility is to map the language expressions to logical meaning representations, and then rely on logical entailment checks, possibly by invoking theorem provers (Rinaldi et al., 2003; Bos & Markert, 2005; Tatu & Moldovan, 2005, 2007). In the case of textual entailment, this involves generating pairs of formulae $\langle \phi_T, \phi_H \rangle$ for T and H (or their possible readings), and then checking if $(\phi_T \wedge B) \models \phi_H$, where B contains meaning postulates and common sense knowledge, as already discussed. In practice, however, it may be very difficult to formulate a reasonably complete B. A partial solution to this problem is to obtain common sense knowledge from resources like WordNet (Fellbaum, 1998) or Extended WordNet (Moldovan & Rus, 2001). The latter also includes logical meaning representations extracted from WordNet's glosses. For example, since "assassinate" is a hyponym (more specific sense) of "kill" in WordNet, an axiom like the following can be added to B (Moldovan & Rus, 2001; Bos & Markert, 2005; Tatu & Moldovan, 2007).

$$\forall x \forall y \; assassinate(x,y) \Rightarrow kill(x,y)$$

Additional axioms can be obtained from FrameNet's frames (Baker, Fillmore, & Lowe, 1998; Lonneker-Rodman & Baker, 2009), as discussed for example by Tatu et al. (2005), or similar resources. Roughly speaking, a frame is the representation of a prototypical situation (e.g., a purchase), which also identifies the situation's main roles (e.g., the buyer, the entity bought), the types of entities (e.g., person) that can play these roles, and possibly relations (e.g., causation, inheritance) to other prototypical situations (other frames). VerbNet (Schuler, 2005) also specifies, among other information, semantic frames for English verbs. On-line encyclopedias have also been used to obtain background knowledge by extracting particular types of information (e.g., is-a relationships) from their articles (Iftene & Balahur-Dobrescu, 2007).

Another approach is to use no particular B (meaning postulates and common sense knowledge), and measure how difficult it is to satisfy both ϕ_T and ϕ_H, in the case of textual entailment recognition, compared to satisfying ϕ_T on its own. A possible measure is the difference of the size of the minimum model that satisfies both ϕ_T and ϕ_H, compared to the size of the minimum model that satisfies ϕ_T on its own (Bos & Markert, 2005); intuitively, a model is an assignment of entities, relations etc. to terms, predicate names, and other domain-dependent atomic symbols. The greater this difference the more knowledge is required in B for $(\phi_T \wedge B) \models \phi_H$ to hold, and the more difficult it becomes for speakers to accept that T textually entails H. Similar bidirectional logical entailment checks can be used to recognize paraphrases (Rinaldi et al., 2003).

2.2 Recognition Approaches that Use Vector Space Models of Semantics

An alternative to using logical meaning representations is to start by mapping each word of the input language expressions to a vector that shows how strongly the word cooccurs with particular other words in corpora (Lin, 1998b), possibly also taking into account syntactic information, for example requiring that the cooccurring words participate in particular syntactic dependencies (Padó & Lapata, 2007). A compositional vector-based meaning representation theory can then be used to combine the vectors of single words, eventually mapping each one of the two input expressions to a single vector that attempts to capture its meaning; in the simplest case, the vector of each expression could be the sum or product of the vectors of its words, but more elaborate approaches have also been proposed (Mitchell & Lapata, 2008; Erk & Padó, 2009; Clarke, 2009). Paraphrases can then be detected by measuring the distance of the vectors of the two input expressions, for example by computing their cosine similarity. See also the work of Turney and Pantel (2010) for a survey of vector space models of semantics.

Recognition approaches based on vector space models of semantics appear to have been explored much less than other approaches discussed in this article, and mostly in paraphrase recognition (Erk & Padó, 2009). They could also be used in textual entailment recognition, however, by checking if the vector of H is particularly close to that of a part (e.g., phrase or sentence) of T. Intuitively, this would check if what H says is included in what T says, though we must be careful with negations and other expressions that do not preserve truth values (Zaenen et al., 2005; MacCartney & Manning, 2009), as in (29)–(30). We return to the idea of matching H to a part of T below.

(29) T: He *denied that* BigCo bought SmallCo.

(30) H: BigCo bought SmallCo.

2.3 Recognition Approaches Based on Surface String Similarity

Several paraphrase recognition methods operate directly on the input surface strings, possibly after applying some pre-processing, such as part-of-speech (POS) tagging or named-entity recognition, but without computing more elaborate syntactic or semantic representations. For example, they may compute the string edit distance (Levenshtein, 1966) of the two input strings, the number of their common words, or combinations of several string similarity measures (Malakasiotis & Androutsopoulos, 2007), including measures originating from machine translation evaluation (Finch, Hwang, & Sumita, 2005; Perez & Alfonseca, 2005; Zhang & Patrick, 2005; Wan, Dras, Dale, & Paris, 2006). The latter have been developed to automatically compare machine-generated translations against human-authored reference translations. A well known measure is BLEU (Papineni, Roukos, Ward, & Zhu, 2002; Zhou et al., 2006a), which roughly speaking examines the percentage of word n-grams (sequences of consecutive words) of the machine-generated translations that also occur in the reference translations, and takes the geometric average of the percentages obtained for different values of n. Although such n-gram based measures have been criticised in machine translation evaluation (Callison-Burch, Osborne, & Koehn, 2006b), for example because they are unaware of synonyms and longer paraphrases, they can be combined with other measures to build paraphrase (and textual entailment) recognizers (Zhou et al., 2006a; Kauchak & Barzilay, 2006; Padó et al., 2009), which may help address the problems of automated machine translation evaluation.

In textual entailment recognition, one of the input language expressions (T) is often much longer than the other one (H). If a part of T's surface string is very similar to H's, this is an indication that H may be entailed by T. This is illustrated in (31)–(32), where H is included verbatim in T.[11] Note, however, that the surface string similarity (e.g., measured by string edit distance) between H and the entire T of this example is low, because of the different lengths of T and H.

(31) T: Charles de Gaulle died in 1970 at the age of eighty. He was thus fifty years old when, as an unknown officer recently promoted to the rank of brigadier general, he made his famous broadcast from London rejecting the capitulation of France to the Nazis after the debacle of May-June 1940.

(32) H: Charles de Gaulle died in 1970.

Comparing H to a sliding window of T's surface string of the same size as H (in our example, six consecutive words of T) and keeping the largest similarity score between the sliding window and H may provide a better indication of whether T entails H or not (Malakasiotis, 2009). In many correct textual entailment pairs, however, using a single sliding window of a fixed length may still be inadequate, because H may correspond to several non-continuous parts of T; in (33)–(34), for example, H corresponds to the three underlined parts of T.[12]

(33) T: The Gaspe, also known as la Gaspesie in French, is a North American peninsula on the south shore of the Saint Lawrence River, in Quebec.

(34) H: The Gaspe is a peninsula in Quebec.

One possible solution is to attempt to align the words (or phrases) of H to those of T, and consider T–H a correct textual entailment pair if a sufficiently good alignment is found, in the simplest case if a large percentage of T's words are aligned to words of H. Another approach would be to use a window of variable length; the window could be, for example, the shortest span of T that contains all of T's words that are aligned to words of H (Burchardt, Pennacchiotti, Thater, & Pinkal, 2009). In any case, we need to be careful with negations and other expressions that do not preserve truth values, as already mentioned. Note, also, that although effective word alignment methods have been developed in statistical machine translation (Brown, Della Pietra, Della Pietra, & Mercer, 1993; Vogel, Ney, & Tillmann, 1996; Och & Ney, 2003), they often perform poorly on textual entailment pairs, because T and H are often of very different lengths, they do not necessarily convey the same information, and textual entailment training datasets are much smaller than those used in machine translation; see MacCartney et al.'s (2008) work for further related discussion and a word alignment method developed especially for textual entailment pairs.[13]

2.4 Recognition Approaches Based on Syntactic Similarity

Another common approach is to work at the syntax level. Dependency grammar parsers (Melcuk, 1987; Kubler, McDonald, & Nivre, 2009) are popular in paraphrasing and textual entailment re-

[11] Example from the dataset of RTE-3 (Giampiccolo et al., 2007).

[12] Modified example from the dataset of the RTE-3 (Giampiccolo et al., 2007).

[13] Cohn et al. (2008) discuss how a publicly available corpus with manually word-aligned paraphrases was constructed. Other word-aligned paraphrasing or textual entailment datasets can be found at the ACL Textual Entailment Portal.

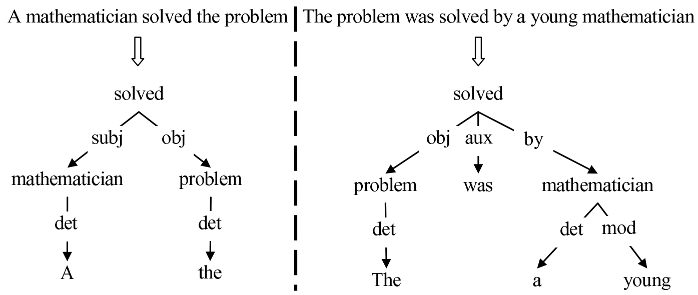

Figure 1: Two sentences that are very similar when viewed at the level of dependency trees.

search, as in other natural language processing areas recently. Instead of showing hierarchically the syntactic constituents (e.g., noun phrases, verb phrases) of a sentence, the output of a dependency grammar parser is a graph (usually a tree) whose nodes are the words of the sentence and whose (labeled) edges correspond to syntactic dependencies between words, for example the dependency between a verb and the head noun of its subject noun phrase, or the dependency between a noun and an adjective that modifies it. Figure 1 shows the dependency trees of two sentences. The exact form of the trees and the edge labels would differ, depending on the parser; for simplicity, we show prepositions as edges. If we ignore word order and the auxiliary "was" of the passive (right) sentence, and if we take into account that the `by` edge of the passive sentence corresponds to the `subj` edge of the active (left) one, the only difference is the extra adjective of the passive sentence. Hence, it is easy to figure out from the dependency trees that the two sentences have very similar meanings, despite their differences in word order. Strictly speaking, the right sentence textually entails the left one, not the reverse, because of the word "young" in the right sentence.

Some paraphrase recognizers simply count the common edges of the dependency trees of the input expressions (Wan et al., 2006; Malakasiotis, 2009) or use other tree similarity measures. A large similarity score (e.g., above a threshold) indicates that the input expressions may be paraphrases. Tree edit distance (Selkow, 1977; Tai, 1979; Zhang & Shasha, 1989) is another example of a similarity measure that can be applied to dependency or other parse trees; it computes the sequence of operator applications (e.g., add, replace, or remove a node or edge) with the minimum cost that turns one tree into the other.[14] To obtain more accurate predictions, it is important to devise an appropriate inventory of operators and assign appropriate costs to the operators during a training stage (Kouylekov & Magnini, 2005; Mehdad, 2009; Harmeling, 2009). For example, replacing a noun with one of its synonyms should be less costly than replacing it with an unrelated word; and removing a dependency between a verb and an adverb should perhaps be less costly than removing a dependency between a verb and the head noun of its subject or object.

In textual entailment recognition, one may compare H's parse tree against subtrees of T's parse tree (Iftene & Balahur-Dobrescu, 2007; Zanzotto, Pennacchiotti, & Moschitti, 2009). It may be possible to match H's tree against a single subtree of T, in effect a single syntactic window on T, as illustrated in Figure 2, which shows the dependency trees of (33)–(34); recall that (34) does not match a single window of (33) at the surface string level.[15] This is also a further example of how operating at a higher level than surface strings may reveal similarities that may be less clear at lower

[14] EDITS, a suite to recognize textual entailment by computing edit distances, is publicly available.

[15] Figure 2 is based on the output of Stanford's parser. One might argue that "North" should modify "American".

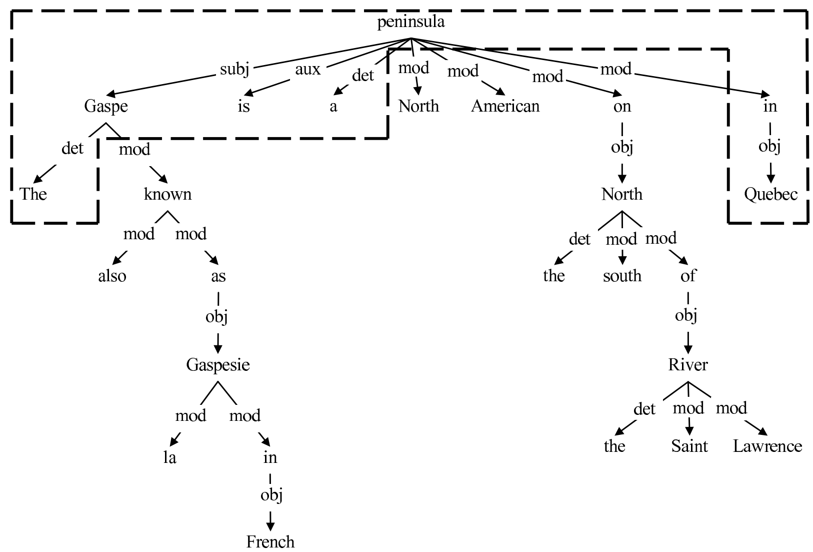

Figure 2: An example of how dependency trees may make it easier to match a short sentence (subtree inside the dashed line) to a part of a longer one.

levels. Another example is (35)–(36); although (35) includes verbatim (36), it does not textually entail (36).[16] This is clear when one compares the syntactic representations of the two sentences: Israel is the subject of "was established" in (36), but not in (35). The difference, however, is not evident at the surface string level, and a sliding window of (35) would match exactly (36), wrongly suggesting a textual entailment.

(35) T: The National Institute for Psychobiology in Israel was established in 1979.
(36) H: Israel was established in 1979.

Similar arguments can be made in favour of computing similarities at the semantic level (Qiu, Kan, & Chua, 2006); for example, both the active and passive forms of a sentence may be mapped to the same logical formula, making their similarity clearer than at the surface or syntax level. The syntactic or semantic representations of the input expressions, however, cannot always be computed accurately (e.g., due to parser errors), which may introduce noise; and, possibly because of the noise, methods that operate at the syntactic or semantic level do not necessarily outperform in practice methods that operate on surface strings (Wan et al., 2006; Burchardt, Reiter, Thater, & Frank, 2007; Burchardt et al., 2009).

2.5 Recognition via Similarity Measures Operating on Symbolic Meaning Representations

Paraphrases may also be recognized by computing similarity measures on graphs whose edges do not correspond to syntactic dependencies, but reflect semantic relations mentioned in the input expressions (Haghighi, 2005), for example the relation between a buyer and the entity bought. Relations of this kind may be identified by applying semantic role labeling methods (Màrquez, Carreras,

[16] Modified example from Haghighi et al.'s (2005) work.

Litkowski, & Stevenson, 2008) to the input language expressions. It is also possible to compute similarities between meaning representations that are based on FrameNet's frames (Burchardt et al., 2007). The latter approach has the advantage that semantically related expressions may invoke the same frame (as with "announcement", "announce", "acknowledge") or interconnected frames (e.g., FrameNet links the frame invoked by "arrest" to the frame invoked by "trial" via a path of temporal precedence relations), making similarities and implications easier to capture (Burchardt et al., 2009).[17] The prototypical semantic roles that PropBank (Palmer, Gildea, & Kingsbury, 2005) associates with each verb may also be used in a similar manner, instead of FrameNet's frames. Similarly, in the case of textual entailment recognition, one may compare H's semantic representation (e.g., semantic graph or frame) to parts of T's representation.

WordNet (Fellbaum, 1998), automatically constructed collections of near synonyms (Lin, 1998a; Moore, 2001; Brockett & Dolan, 2005), or resources like NOMLEX (Meyers, Macleod, Yangarber, Grishman, Barrett, & Reeves, 1998) and CATVAR (Habash & Dorr, 2003) that provide nominalizations of verbs and other derivationally related words across different POS categories (e.g., "to invent" and "invention"), can be used to match synonyms, hypernyms–hyponyms, or, more generally, semantically related words across the two input expressions. According to WordNet, in (37)–(38) "shares" is a direct hyponym (more specific meaning) of "stock", "slumped" is a direct hyponym of "dropped", and "company" is an indirect hyponym (two levels down) of "organization".[18] By treating semantically similar words (e.g., synonyms, or hypernyms-hyponyms up to a small hierarchical distance) as identical (Rinaldi et al., 2003; Finch et al., 2005; Tatu, Iles, Slavick, Novischi, & Moldovan, 2006; Iftene & Balahur-Dobrescu, 2007; Malakasiotis, 2009; Harmeling, 2009), or by considering (e.g., counting) semantically similar words across the two input language expressions (Brockett & Dolan, 2005; Bos & Markert, 2005), paraphrase recognizers may be able to cope with paraphrases that have very similar meanings, but very few or no common words.

(37) The shares of the company dropped.
(38) The organization's stock slumped.

In textual entailment recognition, it may be desirable to allow the words of T to be more distant hyponyms of the words of H, compared to paraphrase recognition. For example, "X is a computer" textually entails "X is an artifact", and "computer" is a hyponym of "artifact" four levels down.

Measures that exploit WordNet (or similar resources) and compute the semantic similarity between two words or, more generally, two texts have also been proposed (Leacock, Miller, & Chodorow, 1998; Lin, 1998c; Resnik, 1999; Budanitsky & Hirst, 2006; Tsatsaronis, Varlamis, & Vazirgiannis, 2010).[19] Some of them are directional, making them more suitable to textual entailment recognition (Corley & Mihalcea, 2005). Roughly speaking, measures of this kind consider (e.g., sum the lengths of) the paths in WordNet's hierarchies (or similar resources) that connect the senses of corresponding (e.g., most similar) words across the two texts. They may also take into account information such as the frequencies of the words in the two texts and how rarely they are encountered in documents of a large collection (inverse document frequency). The rationale is that frequent words of the input texts that are rarely used in a general corpus are more important, as in

[17] Consult, for example, the work of Erk and Padó (2006) for a description of a system that can annotate texts with FrameNet frames. The FATE corpus (Burchardt & Pennacchiotti, 2008), a version of the RTE 2 test set (Bar-Haim et al., 2006) with FrameNet annotations, is publicly available.

[18] Modified example from the work of Tsatsaronis (2009)

[19] Pedersen's WordNet::Similarity package implements many of these measures.

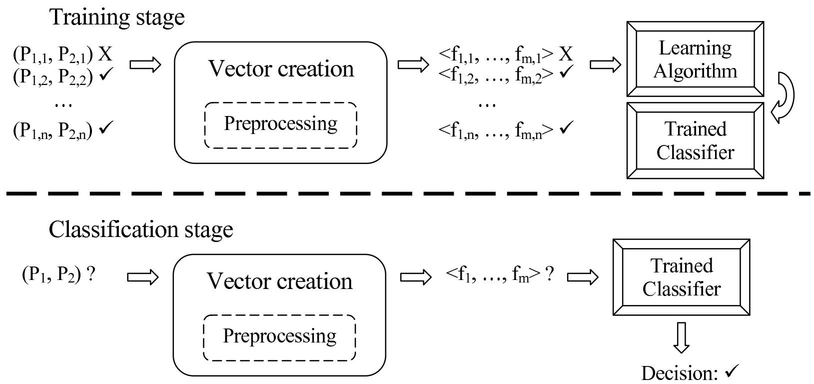

Figure 3: Paraphrase and textual entailment recognition via supervised machine learning.

information retrieval; hence, the paths that connect them should be assigned greater weights. Since they often consider paths between word *senses*, many of these measures would ideally be combined with word sense disambiguation (Yarowski, 2000; Stevenson & Wilks, 2003; Kohomban & Lee, 2005; Navigli, 2008), which is not, however, always accurate enough for practical purposes.

2.6 Recognition Approaches that Employ Machine Learning

Multiple similarity measures, possibly computed at different levels (surface strings, syntactic or semantic representations) may be combined by using machine learning (Mitchell, 1997; Alpaydin, 2004), as illustrated in Figure 3.[20] Each pair of input language expressions $\langle P_1, P_2 \rangle$, i.e., each pair of expressions we wish to check if they are paraphrases or a correct textual entailment pair, is represented by a feature vector $\langle f_1, \ldots, f_m \rangle$. The vector contains the scores of multiple similarity measures applied to the pair, and possibly other features. For example, many systems also include features that check for polarity differences across the two input expressions, as in "this is *not* a confirmed case of rabies" vs. "a case of rabies *was* confirmed", or modality differences, as in "a case *may* have been confirmed" vs. "a case *has* been confirmed" (Haghighi, 2005; Iftene & Balahur-Dobrescu, 2007; Tatu & Moldovan, 2007). Bos and Markert (2005) also include features indicating if a theorem prover has managed to prove that the logical representation of one of the input expressions entails the other or contradicts it. A supervised machine learning algorithm trains a classifier on manually classified (as correct or incorrect) vectors corresponding to training input pairs. Once trained, the classifier can classify unseen pairs as correct or incorrect paraphrases or textual entailment pairs by examining their features (Bos & Markert, 2005; Brockett & Dolan, 2005; Zhang & Patrick, 2005; Finch et al., 2005; Wan et al., 2006; Burchardt et al., 2007; Hickl, 2008; Malakasiotis, 2009; Nielsen et al., 2009).

A preprocessing stage is commonly applied to each input pair of language expressions, before converting it to a feature vector (Zhang & Patrick, 2005). Part of the preprocessing may provide

[20] WEKA (Witten & Frank, 2005) provides implementations of several well known machine learning algorithms, including C4.5 (Quinlan, 1993), Naive Bayes (Mitchell, 1997), SVMs (Vapnik, 1998; Cristianini & Shawe-Taylor, 2000; Joachims, 2002), and AdaBoost (Freund & Schapire, 1995; Friedman, Hastie, & Tibshirani, 2000). More efficient implementations of SVMs, such as LIBSVM and SVM-LIGHT, are also available. Maximum Entropy classifiers are also very effective; see chapter 6 of the book "Speech and Language Processing" (Jurafsky & Martin, 2008) for an introduction; Stanford's implementation is frequently used.

information that is required to compute the features; for example, this is when a POS tagger or a parser would be applied.[21] The preprocessing may also normalize the input pairs; for example, a stemmer may be applied; dates may be converted to a consistent format; names of persons, organizations, locations etc. may be tagged by their semantic categories using a named entity recognizer; pronouns or, more generally, referring expressions, may be replaced by the expressions they refer to (Hobbs, 1986; Lappin & Leass, 1994; Mitkov, 2003; Mollá, Schwitter, Rinaldi, Dowdall, & Hess, 2003; Yang, Su, & Tan, 2008); and morphosyntactic variations may be normalized (e.g., passive sentences may be converted to active ones).[22]

Instead of mapping each $\langle P_1, P_2 \rangle$ pair to a feature vector that contains mostly scores measuring the similarity between P_1 and P_2, it is possible to use vectors that encode directly parts of P_1 and P_2, or parts of their syntactic or semantic representations. Zanzotto et al. (2009) project each $\langle P_1, P_2 \rangle$ pair to a vector that, roughly speaking, contains as features all the fragments of P_1 and P_2's parse trees. Leaf nodes corresponding to identical or very similar words (according to a WordNet-based similarity measure) across P_1 and P_2 are replaced by co-indexed slots, to allow the features to be more general. Zanzotto et al. define a measure (actually, different versions of it) that, in effect, computes the similarity of two pairs $\langle P_1, P_2 \rangle$ and $\langle P_1', P_2' \rangle$ by counting the parse tree fragments (features) that are shared by P_1 and P_1', and those shared by P_2 and P_2'. The measure is used as a kernel in an Support Vector Machine (SVM) that learns to separate positive textual entailment pairs $\langle P_1, P_2 \rangle = \langle T, H \rangle$ from negative ones. A (valid) kernel can be thought of as a similarity measure that projects two objects to a highly dimensional vector space, where it computes the inner product of the projected objects; efficient kernels compute the inner product directly from the original objects, without computing their projections to the highly dimensional vector space (Vapnik, 1998; Cristianini & Shawe-Taylor, 2000; Joachims, 2002). In Zanzotto et al.'s work, each object is a $\langle T, H \rangle$ pair, and its projection is the vector that contains all the parse tree fragments of T and H as features. Consult, for example, the work of Zanzotto and Dell' Arciprete (2009) and Moschitti (2009) for further discussion of kernels that can be used in paraphrase and textual entailment recognition.

2.7 Recognition Approaches Based on Decoding

Pairs of paraphrasing or textual entailment expressions (or templates) like (39), often called *rules*, that may have been produced by extraction mechanisms (to be discussed in Section 4) can be used by recognizers much as, and often in addition to synonyms and hypernyms-hyponyms.

(39) "X is fond of Y" $\approx$ "X likes Y"

Given the paraphrasing rule of (39) and the information that "child" is a synonym of "kid" and "candy" a hyponym of "sweet", a recognizer could figure out that (40) textually entails (43) by gradually transforming (40) to (43) as shown below.[23]

(40) Children are fond of sweets.

[21] Brill's (1992) POS tagger is well-known and publicly available. Stanford's tagger (Toutanova, Klein, Manning, & Singer, 2003) is another example of a publicly available POS tagger. Commonly used parsers include Charniak's (2000), Collin's (2003), the Link Grammar Parser (Sleator & Temperley, 1993), MINIPAR, a principle-based parser (Berwick, 1991) very similar to PRINCIPAR (Lin, 1994), MaltParser (Nivre, Hall, Nilsson, Chanev, Eryigit, Kuebler, Marinov, & Marsi, 2007), and Stanford's parser (Klein & Manning, 2003).

[22] Porter's stemmer (1997) is well-known. An example of a publicly available named-entity recognizer is Stanford's.

[23] Modified example from Bar-Haim et al.'s (2009) work.

(41) Kids are fond of sweets.
(42) Kids like sweets.
(43) Kids like candies.

Another recognition approach, then, is to search for a sequence of rule applications or other transformations (e.g., replacing synonyms, or hypernyms-hyponyms) that turns one of the input expressions (or its syntactic or semantic representation) to the other. We call this search *decoding*, because it is similar to the decoding stage of machine translation (to be discussed in Section 3), where a sequence of transformations that turns a source-language expression into a target-language expression is sought. In our case, if a sequence is found, the two input expressions constitute a positive paraphrasing or textual entailment pair, depending on the rules used; otherwise, the pair is negative. If each rule is associated with a confidence score (possibly learnt from a training dataset) that reflects the degree to which the rule preserves the original meaning in paraphrase recognition, or the degree to which we are confident that it produces an entailed expression, we may search for the sequence of transformations with the maximum score (or minimum cost), much as in approaches that compute the minimum (string or tree) edit distance between the two input expressions. The pair of input expressions can then be classified as positive if the maximum-score sequence exceeds a confidence threshold (Harmeling, 2009). One would also have to consider the contexts where rules are applied, because a rule may not be valid in all contexts, for instance because of the different possible senses of the words it involves. A possible solution is to associate each rule with a vector that represents the contexts where it can be used (e.g., a vector of frequently occurring words in training contexts where the rule applies), and use a rule only in contexts that are similar to its associated context vector; with slotted rules, one can also model the types of slot values (e.g., types of named entities) the rule can be used with the work of Pantel, Bhagat, Coppola, Chklovski, and Hovy (2007), and Szpektor, Dagan, Bar-Haim, and Goldberger (2008).

Resouces like WordNet and extraction methods, however, provide thousands or millions of rules, giving rise to an exponentially large number of transformation sequences to consider.[24] When operating at the level of semantic representations, the sequence sought is in effect a proof that the two input expressions are paraphrases or a valid textual entailment pair, and it may be obtained by exploiting theorem provers, as discussed earlier. Bar-Haim et al. (2007) discuss how to search for sequences of transformations, seen as proofs at the syntactic level, when the input language expressions and their reformulations are represented by dependency trees. In subsequent work (Bar-Haim et al., 2009), they introduce compact forests, a data structure that allows the dependency trees of multiple intermediate reformulations to be represented by a single graph, to make the search more efficient. They also combine their approach with an SVM-based recognizer; sequences of transformations are used to bring T closer to H, and the SVM recognizer is then employed to judge if the transformed T and H consitute a positive textual entailment pair or not.

2.8 Evaluating Recognition Methods

Experimenting with paraphase and textual entailment recognizers requires datasets containing both positive and negative input pairs. When using discriminative classifiers (e.g., SVMs), the negative training pairs must ideally be near misses, otherwise they may be of little use (Schohn & Cohn, 2000; Tong & Koller, 2002). Near misses can also make the test data more challenging.

[24] Collections of transformation rules and resources that can be used to obtain such rules are listed at the ACL Textual Entailment Portal. Mirkin et al. (2009a) discuss how to evaluate collections of textual entailment rules.

method	*accuracy (%)*	*precision (%)*	*recall (%)*	*F-measure (%)*
Corley & Mihalcea (2005)	71.5	72.3	92.5	81.2
Das & Smith (2009)	76.1	79.6	86.1	82.9
Finch et al. (2005)	75.0	76.6	89.8	82.7
Malakasiotis (2009)	76.2	79.4	86.8	82.9
Qiu et al. (2006)	72.0	72.5	93.4	81.6
Wan et al. (2006)	75.6	77.0	90.0	83.0
Zhang & Patrick (2005)	71.9	74.3	88.2	80.7
BASE_1	66.5	66.5	100.0	79.9
BASE_2	69.0	72.4	86.3	78.8

Table 1: Paraphrase recognition results on the MSR corpus.

The most widely used benchmark dataset for paraphrase recognition is the Microsoft Research (MSR) Paraphrase Corpus. It contains 5,801 pairs of sentences obtained from clusters of online news articles referring to the same events (Dolan, Quirk, & Brockett, 2004; Dolan & Brockett, 2005). The pairs were initially filtered by heuristics, which require, for example, the word edit distance of the two sentences in each pair to be neither too small (to avoid nearly identical sentences) nor too large (to avoid too many negative pairs); and both sentences to be among the first three of articles from the same cluster (articles referring to the same event), the rationale being that initial sentences often summarize the events. The candidate paraphrase pairs were then filtered by an SVM-based paraphrase recognizer (Brockett & Dolan, 2005), trained on separate manually classified pairs obtained in a similar manner, which was biased to overidentify paraphrases. Finally, human judges annotated the remaining sentence pairs as paraphrases or not. After resolving disagreements, approximately 67% of the 5,801 pairs were judged to be paraphrases. The dataset is divided in two non-overlapping parts, for training (30% of all pairs) and testing (70%). Zhang and Patrick (2005) and others have pointed out that the heuristics that were used to construct the corpus may have biased it towards particular types of paraphrases, excluding for example paraphrases that do not share any common words.

Table 1 lists all the published results of paraphrase recognition experiments on the MSR corpus we are aware of. We include two baselines we used: BASE_1 classifies all pairs as paraphrases; BASE_2 classifies two sentences as paraphrases when their surface word edit distance is below a threshold, tuned on the training part of the corpus. Four commonly used evaluation measures are used: accuracy, precision, recall, and F-measure with equal weight on precision and recall. These measures are defined below. *TP* (true positives) and *FP* (false positives) are the numbers of pairs that have been correctly or incorrectly, respectively, classified as positive (paraphrases). *TN* (true negatives) and *FN* (false negatives) are the numbers of pairs that have been correctly or incorrectly, respectively, classified as negative (not paraphrases).

$$precision = \frac{TP}{TP+FP}, \qquad recall = \frac{TP}{TP+FN},$$
$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}, \quad F\text{-}measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

All the systems of Table 1 have better recall than precision, which implies they tend to over-classify pairs as paraphrases, possibly because the sentences of each pair have at least some common words and refer to the same event. Systems with higher recall tend to have lower precision, and vice versa, as one would expect. The high F-measure of BASE_1 is largely due to its perfect recall; its precision

method	*accuracy (%)*	*precision (%)*	*recall (%)*	*F-measure (%)*
Bensley & Hickl (2008)	74.6	–	–	–
Iftene (2008)	72.1	65.5	93.2	76.9
Siblini & Kosseim (2008)	68.8	–	–	–
Wang & Neumann (2008)	70.6	–	–	–
BASE$_1$	50.0	50.0	100.0	66.7
BASE$_2$	54.9	53.6	73.6	62.0

Table 2: Textual entailment recognition results (for two classes) on the RTE-4 corpus.

is significantly lower, compared to the other systems. BASE$_2$, which uses only string edit distance, is a competitive baseline for this corpus. Space does not permit listing published evaluation results of all the paraphrase recognition methods that we have discussed. Furthermore, comparing results obtained on different datasets is not always meaningful.

For textual entailment recognition, the most widely used benchmarks are those of the RTE challenges. As an example, the RTE-3 corpus contains 1,600 $\langle T, H \rangle$ pairs (positive or negative). Four application scenarios where textual entailment recognition might be useful were considered: information extraction, information retrieval, question answering, and summarization. There are 200 training and 200 testing pairs for each scenario; Dagan et al. (2009) explain how they were constructed. The RTE-4 corpus was constructed in a similar way, but it contains only test pairs, 250 for each of the four scenarios. A further difference is that in RTE-4 the judges classified the pairs in three classes: true entailment pairs, false entailment pairs where H contradicts T (Harabagiu, Hickl, & Lacatusu, 2006; de Marneffe, Rafferty, & Manning, 2008), and false pairs where reading T does not lead to any conclusion about H; a similar pilot task was included in RTE-3 (Voorhees, 2008). The pairs of the latter two classes can be merged, if only two classes (true and false) are desirable. We also note that RTE-3 included a pilot task requiring systems to justify their answers. Many of the participants, however, used technical or mathematical terminology in their explanations, which was not always appreciated by the human judges; also, the entailments were often obvious to the judges, to the extent that no justification was considered necessary (Voorhees, 2008). Table 2 lists the best accuracy results of RTE-4 participants (for two classes only), along with results of the two baselines described previously; precision, recall, and F-measure scores are also shown, when available. All four measures are defined as in paraphrase recognition, but positives and negatives are now textual entailment pairs.[25] Again, space does not permit listing published evaluation results of all the textual entailment recognition methods that we have discussed, and comparing results obtained on different datasets is not always meaningful.

It is also possible to evaluate recognition methods indirectly, by measuring their impact on the performance of larger natural language processing systems (Section 1.1). For instance, one could measure the difference in the performance of a QA system, or the degree to which the redundancy of a generated summary is reduced when using paraphrase and/or textual entailment recognizers.

[25] *Average precision*, borrowed from information retrieval evaluation, has also been used in the RTE challenges. Bergmair (2009), however, argues against using it in RTE challenges and proposes alternative measures.

3. Paraphrase and Textual Entailment Generation

Unlike recognizers, paraphrase or textual entailment *generators* are given a single language expression (or template) as input, and they are required to produce as many output language expressions (or templates) as possible, such that the output expressions are paraphrases or they constitute, along with the input, correct textual entailment pairs. Most generators assume that the input is a single sentence (or sentence template), and we adopt this assumption in the remainder of this section.

3.1 Generation Methods Inspired by Statistical Machine Translation

Many generation methods borrow ideas from statistical machine translation (SMT).[26] Let us first introduce some central ideas from SMT, for the benefit of readers unfamiliar with them. SMT methods rely on very large bilingual or multilingual parallel corpora, for example the proceedings of the European parliament, without constructing meaning representations and often, at least until recently, without even constructing syntactic representations.[27] Let us assume that we wish to translate a sentence F, whose words are $f_1, f_2, \ldots, f_{|F|}$ in that order, from a foreign language to our native language. Let us also denote by N any candidate translation, whose words are $a_1, a_2, \ldots, a_{|N|}$. The best translation, denoted N^*, is the N with the maximum probability of being a translation of F, i.e:

$$N^* = \arg\max_N P(N|F) = \arg\max_N \frac{P(N)P(F|N)}{P(F)} = \arg\max_N P(N)P(F|N) \tag{44}$$

Since F is fixed, the denominator $P(F)$ above is constant and can be ignored when searching for N^*. $P(N)$ is called the *language model* and $P(F|N)$ the *translation model*.

For modeling purposes, it is common to assume that F was in fact originally written in our native language and it was transmitted to us via a noisy channel, which introduced various deformations. The possible deformations may include, for example, replacing a native word with one or more foreign ones, removing or inserting words, moving words to the left or right etc. The commonly used IBM models 1 to 5 (Brown et al., 1993) provide an increasingly richer inventory of word deformations; more recent phrase-based SMT systems (Koehn, Och, & Marcu, 2003) also allow directly replacing entire phrases with other phrases. The foreign sentence F can thus be seen as the result of applying a sequence of transformations $D = \langle d_1, d_2, \ldots, d_{|D|} \rangle$ to N, and it is common to search for the N^* that maximizes (45); this search is called *decoding*.

$$N^* = \arg\max_N P(N) \max_D P(F,D|N) \tag{45}$$

An exhaustive search is usually intractable. Hence, heuristic search algorithms (e.g., based on beam search) are usually employed (Germann, Jahr, Knight, Marcu, & Yamada, 2001; Koehn, 2004).[28]

Assuming for simplicity that the individual deformations $d_i(\cdot)$ of D are mutually independent, $P(F,D|N)$ can be computed as the product of the probabilities of D's individual deformations. Given a bilingual parallel corpus with words aligned across languages, we can estimate the probabilities

[26] For an introduction to SMT, see chapter 25 of the book "Speech and Language Processing" (Jurafsky & Martin, 2008), and chapter 13 of the book "Foundations of Statistical Natural Language Processing" (Manning & Schuetze, 1999). For a more extensive discussion, consult the work of Koehn (2009).

[27] See Koehn's Statistical Machine Translation site for commonly used SMT corpora and tools.

[28] A frequently used SMT system that includes decoding facilities is Moses.

of all possible deformations $d_i(\cdot)$. In practice, however, parallel corpora do not indicate word alignment. Hence, it is common to find the most probable word alignment of the corpus given initial estimates of individual deformation probabilities, then re-estimate the deformation probabilities given the resulting alignment, and iterate (Brown et al., 1993; Och & Ney, 2003).[29]

The translation model $P(F,D|N)$ estimates the probability of obtaining F from N via D; we are interested in Ns with high probabilities of leading to F. We also want, however, N to be grammatical, and we use the language model $P(N)$ to check for grammaticality. $P(N)$ is the probability of encountering N in our native language; it is estimated from a large monolingual corpus of our language, typically assuming that the probability of encountering word a_i depends only on the preceding $n-1$ words. For $n=3$, $P(N)$ becomes:

$$P(N) = P(a_1) \cdot P(a_2|a_1) \cdot P(a_3|a_1,a_2) \cdot P(a_4|a_2,a_3) \cdots P(a_{|N|}|a_{|N|-2},a_{|N|-1}) \tag{46}$$

A language model typically also includes smoothening mechanisms, to cope with n-grams that are very rare or not present in the monolingual corpus, which would lead to $P(N) = 0$.[30]

In principle, an SMT system could be used to generate paraphrases, if it could be trained on a sufficiently large monolingual corpus of parallel texts. Both N and F are now sentences of the same language, but N has to be different from the given F, and it has to convey the same (or almost the same) information. The main problem is that there are no readily available monolingual parallel corpora of the sizes that are used in SMT, to train the language model on them. One possibility is to use multiple translations of the same source texts; for example, different English translations of novels originally written in other languages (Barzilay & McKeown, 2001), or multiple English translations of Chinese news articles, as in the Multiple-Translation Chinese Corpus. Corpora of this kind, however, are still orders of magnitude smaller than those used in SMT.

To bypass the lack of large monolingual parallel corpora, Quirk et al. (2004) use clusters of news articles referring to the same event. The articles of each cluster do not always report the same information and, hence, they are not parallel texts. Since they talk about the same event, however, they often contain phrases, sentences, or even longer fragments with very similar meanings; corpora of this kind are often called *comparable*. From each cluster, Quirk et al. select pairs of similar sentences (e.g., with small word edit distance, but not identical sentences) using methods like those employed to create the MSR corpus (Section 2.8).[31] The sentence pairs are then word aligned as in machine translation, and the resulting alignments are used to create a table of phrase pairs as in phrase-based SMT systems (Koehn et al., 2003). A phrase pair $\langle P_1, P_2 \rangle$ consists of contiguous words (taken to be a phrase, though not necessarily a syntactic constituent) P_1 of one sentence that are aligned to different contiguous words P_2 of another sentence. Quirk et al. provide the following examples of discovered pairs.

[29] GIZA++ is often used to train IBM models and align words.

[30] See chapter 4 of the book "Speech and Language Processing" (Jurafsky & Martin, 2008) and chapter 6 of the book "Foundations of Statistical Natural Language Processing" (Manning & Schuetze, 1999) for an introduction to language models. SRILM (Stolcke, 2002) is a commonly used tool to create language models.

[31] Wubben et al. (2009) discuss similar methods to pair news titles. Barzilay & Elhadad (2003) and Nelken & Shieber (2006) discuss more general methods to align sentences of monolingual comparable corpora. Sentence alignment methods for bilingual parallel or comparable corpora are discussed, for example, by Gale and Church (1993), Melamed (1999), Fung and Cheung (2004), Munteanu and Marcu (2006); see also the work of Wu (2000). Sentence alignment methods for parallel corpora may perform poorly on comparable corpora (Nelken & Shieber, 2006).

P_1	P_2
injured	wounded
Bush administration	White House
margin of error	error margin
...	...

Phrase pairs that occur frequently in the aligned sentences may be assigned higher probabilities; Quirk et al. use probabilities returned by IBM model 1. Their decoder first constructs a lattice that represents all the possible paraphrases of the input sentence that can be produced by replacing phrases by their counterparts in the phrase table; i.e., the possible deformations $d_i(\cdot)$ are the phrase replacements licensed by the phrase table.[32] Unlike machine translation, not all of the words or phrases need to be replaced, which is why Quirk et al. also allow a degenerate identity deformation $d_{id}(\xi) = \xi$; assigning a high probability to the identity deformation leads to more conservative paraphrases, with fewer phrase replacements. The decoder uses the probabilities of $d_i(\cdot)$ to compute $P(F,D|N)$ in equation (45), and the language model to compute $P(N)$. The best scored N^* is returned as a paraphrase of F; the n most highly scored Ns could also be returned. More generally, the table of phrase pairs may also include synonyms obtained from WordNet or similar resources, or pairs of paraphrases (or templates) discovered by paraphrase extraction methods; in effect, Quirk et al.'s construction of a monolingual phrase table is a paraphrase extraction method. A language model may also be applied locally to the replacement words of a deformation and their context to assess whether or not the new words fit the original context (Mirkin et al., 2009b).

Zhao et al. (2008, 2009) demonstrated that combining phrase tables derived from multiple resources improves paraphrase generation. They also proposed scoring the candidate paraphrases by using an additional, application-dependent model, called the *usability model*; for example, in sentence compression (Section 1.1) the usability model rewards Ns that have fewer words than F. Equation (45) then becomes (47), where $U(F,N)$ is the usability model and λ_i are weights assigned to the three models; similar weights can be used in (45).

$$N^* = \arg\max_N U(F,N)^{\lambda_1} P(N)^{\lambda_2} \max_D P(F,D|N)^{\lambda_3} \tag{47}$$

Zhao et al. actually use a log-linear formulation of (47); and they select the weights λ_i that maximize an objective function that rewards many and correct (as judged by human evaluators) phrasal replacements.[33] One may replace the translation model by a paraphrase recognizer (Section 2) that returns a confidence score; in its log-linear formulation, (47) then becomes (48), where $R(F,N)$ is the confidence score of the recognizer.

$$N^* = \arg\max_N [\lambda_1 \log U(F,N) + \lambda_2 \log P(N) + \lambda_3 log R(F,N)] \tag{48}$$

Including hyponyms-hypernyms or textual entailment rules (Section 2.7) in the phrase table would generate sentences N that textually entail or are entailed (depending on the direction of the

[32] Chevelu et al. (2009) discuss how other decoders could be developed especially for paraphrase generation.

[33] In a "reluctant paraphrasing" setting (Dras, 1998), for example when revising a document to satisfy length requirements, readability measures, or other externally imposed constraints, it may be desirable to use an objective function that rewards making as *few* changes as possible, provided that the constraints are satisfied. Dras (1998) discusses a formulation of this problem in terms of integer programming.

rules and whether we replace hyponyms by hypernyms or the reverse) by F. SMT-inspired methods, however, have been used mostly in paraphrase generation, not in textual entailment generation.

Paraphrases can also be generated by using pairs of machine translation systems to translate the input expression to a new language, often called a *pivot* language, and then back to the original language. The resulting expression is often different from the input one, especially when the two translation systems employ different methods. By using different pairs of machine translation systems or different pivot languages, multiple paraphrases may be obtained. Duboue and Chu-Carroll (2006) demonstrated the benefit of using this approach to paraphrase questions, with an additional machine learning classifier to filter the generated paraphrases; their classifier uses features such as the cosine similarity between a candidate generated paraphrase and the original question, the lengths of the candidate paraphrase and the original question, features showing whether or not both questions are of the same type (e.g., both asking for a person name), etc. An advantage of this approach is that the machine translation systems can be treated as black boxes, and they can be trained on readily available parallel corpora of different languages. A disadvantage is that translation errors from both directions may lead to poor paraphrases. We return to pivot languages in Section 4.

In principle, the output of a generator may be produced by mapping the input to a representation of its meaning, a process that usually presupposes parsing, and by passing on the meaning representation, or new meaning representations that are logically entailed by the original one, to a natural language generation system (Reiter & Dale, 2000; Bateman & Zock, 2003) to produce paraphrases or entailed language expressions. This approach would be similar to using language-independent meaning representations (an "interlingua") in machine translation, but here the meaning representations would not need to be language-independent, since only one language is involved. An approach similar to syntactic transfer in machine translation may also be adopted (McKeown, 1983). In that case, the input language expression (assumed to be a sentence) is first parsed. The resulting syntactic representation is then modified in ways that preserve, or affect only slightly, the original meaning (e.g., turning a sentence from active to passive), or in ways that produce syntactic representations of entailed language expressions (e.g., pruning certain modifiers or subordinate clauses). New language expressions are then generated from the new syntactic representations, possibly by invoking the surface realization components of a natural language generation system. Parsing, however, the input expression may introduce errors, and producing a correct meaning representation of the input, when this is required, may be far from trivial. Furthermore, the natural language generator may be capable of producing language expressions of only a limited variety, missing possible paraphrases or entailed language expressions. This is perhaps why meaning representation and syntactic transfer do not seem to be currently popular in paraphrase and textual entailment generation.

3.2 Generation Methods that Use Bootstrapping

When the input and output expressions are slotted templates, it is possible to apply bootstrapping to a large monolingual corpus (e.g., the entire Web), instead of using machine translation methods. Let us assume, for example, that we wish to generate paraphrases of (49), and that we are given a few pairs of seed values of X and Y, as in (50) and (51).

(49) X is the author of Y.

(50) $\langle X =$ "Jack Kerouac", $Y =$ "On the Road"$\rangle$

(51) $\langle X =$ "Jules Verne", $Y =$ "The Mysterious Island"$\rangle$

We can retrieve from the corpus sentences that contain any of the seed pairs:

(52) Jack Kerouac wrote “On the Road”.
(53) “The Mysterious Island” was written by Jules Verne.
(54) Jack Kerouac is most known for his novel “On the Road”.

By replacing the known seeds with the corresponding slot names, we obtain new templates:

(55) X wrote Y.
(56) Y was written by X.
(57) X is most known for his novel Y.

In our example, (55) and (56) are paraphrases of (49); however, (57) textually entails (49), but is not a paraphrase of (49). If we want to generate paraphrases, we must keep (55) and (56) only; if we want to generate templates that entail (49), we must keep (57) too. Some of the generated candidate templates may neither be paraphrases of, nor entail (or be entailed by) the original template. A good paraphrase or textual entailment recognizer (Section 2) or a human in the loop would be able to filter out bad candidate templates; see also Duclaye et al.’s (2003) work, where Expectation Maximization (Mitchell, 1997) is used to filter the candidate templates. Simpler filtering techniques may also be used. For example, Ravichandran et al. (2002, 2003) assign to each candidate template a pseudo-precision score; roughly speaking, the score is computed as the number of retrieved sentences that match the candidate template with X and Y having the values of any seed pair, divided by the number of retrieved sentences that match the template when X has a seed value and Y any value, not necessarily the corresponding seed value.

Having obtained new templates, we can search the corpus for new sentences that match them; for example, sentence (58) matches the generated template (56). From the new sentences, more seed values can be obtained, if the slot values correspond to types of expressions (e.g., person names) that can be recognized reasonably well, for example by using a named entity recognizer or a gazetteer (e.g., a large list of book titles); from (58) we would obtain the new seed pair (59). More iterations may be used to generate more templates and more seeds, until no more templates and seeds can be discovered or a maximum number of iterations is reached.

(58) Frankenstein was written by Mary Shelley.
(59) $\langle X = \text{“Mary Shelley”}, Y = \text{“Frankenstein”}\rangle$

Figure 4 illustrates how a bootstrapping paraphrase generator works. Templates that textually entail or that are textually entailed by an initial template, for which seed slot values are provided, can be generated similarly, if the paraphrase recognizer is replaced by a textual entailment recognizer.

If slot values can be recognized reliably, we can also obtain the initial seed slot values automatically by retrieving directly sentences that match the original templates and by identifying the slot values in the retrieved sentences.[34] If we are also given a mechanism to identify sentences of interest in the corpus (e.g., sentences involving particular terms, such as names of known diseases and medicines), we can also obtain the initial templates automatically, by identifying sentences of interest, identifying slot values (e.g., named entities of particular categories) in the sentences, and using the contexts of the slot values as initial templates. In effect, the generation task then becomes an extraction one, since we are given a corpus, but neither initial templates nor seed slot values. TEASE (Szpektor, Tanev, Dagan, & Coppola, 2004) is a well-known bootstrapping method of this

[34] Seed slot values per semantic relation can also be obtained from databases (Mintz, Bills, Snow, & Jurafsky, 2009).

kind, which produces textual entailment pairs, for example pairs like (60)–(61), given only a monolingual (non-parallel) corpus and a dictionary of terms. (60) textually implies (61), for example in contexts like those of (62)–(63), but not the reverse.[35]

(60) *X* prevents *Y*
(61) *X* reduces *Y* risk
(62) Aspirin prevents heart attack.
(63) Aspirin reduces heart attack risk.

TEASE does not specify the directionality of the produced template pairs, for example whether (60) textually entails (61) or vice versa, but additional mechanisms have been proposed that attempt to guess the directionality; we discuss one such mechanism, LEDIR (Bhagat, Pantel, & Hovy, 2007), in Section 4.1 below. Although TEASE can also be used as a generator, if particular input templates are provided, we discuss it further in Section 4.2, along with other bootstrapping *extraction* methods, since in its full form it requires no initial templates (nor seed slot values). The reader is reminded that the boundaries between recognizers, generators, and extractors are not always clear.

Similar bootstrapping methods have been used to generate information extraction patterns (Riloff & Jones, 1999; Xu, Uszkoreit, & Li, 2007). Some of these methods, however, require corpora annotated with instances of particular types of events to be extracted (Huffman, 1995; Riloff, 1996b; Soderland, Fisher, Aseltine, & Lehnert, 1995; Soderland, 1999; Muslea, 1999; Califf & Mooney, 2003), or texts that mention the target events and near-miss texts that do not (Riloff, 1996a).

Marton et al. (2009) used a similar approach, but without iterations, to generate paraphrases of unknown source language phrases in a phrase-based SMT system (Section 1.1). For each unknown phrase, they collected contexts where the phrase occurred in a monolingual corpus of the source language, and they searched for other phrases (candidate paraphrases) in the corpus that occurred in the same contexts. They subsequently produced feature vectors for both the unknown phrase and its candidate paraphrases, with each vector showing how often the corresponding phrase cooccurred with other words. The candidate paraphrases were then ranked by the similarity of their vectors to the vector of the unknown phrase. The unknown phrases were in effect replaced by their best paraphrases that the SMT system knew how to map to target language phrases, and this improved the SMT system's performance.

3.3 Evaluating Generation Methods

In most generation applications, for example when rephrasing queries to a QA system (Section 1.1), it is desirable not only to produce correct outputs (correct paraphrases, or expressions that constitute correct textual entailment pairs along with the input), but also to produce as many correct outputs as possible. The two goals correspond to high precision and recall, respectively. For a particular input s_i, the precision p_i and recall r_i of a generator can now be defined as follows (cf. Section 2.8). TP_i is the number of correct outputs for input s_i, FP_i is the number of wrong outputs for s_i, and FN_i is the number of outputs for s_i that have incorrectly not been generated (missed).

$$p_i = \frac{TP_i}{TP_i + FP_i}, \quad r_i = \frac{TP_i}{TP_i + FN_i}$$

The precision and recall scores of a method over a set of inputs $\{s_i\}$ can then be defined using micro-averaging or macro-averaging:

[35] Example from the work of Szpektor et al. (2004).

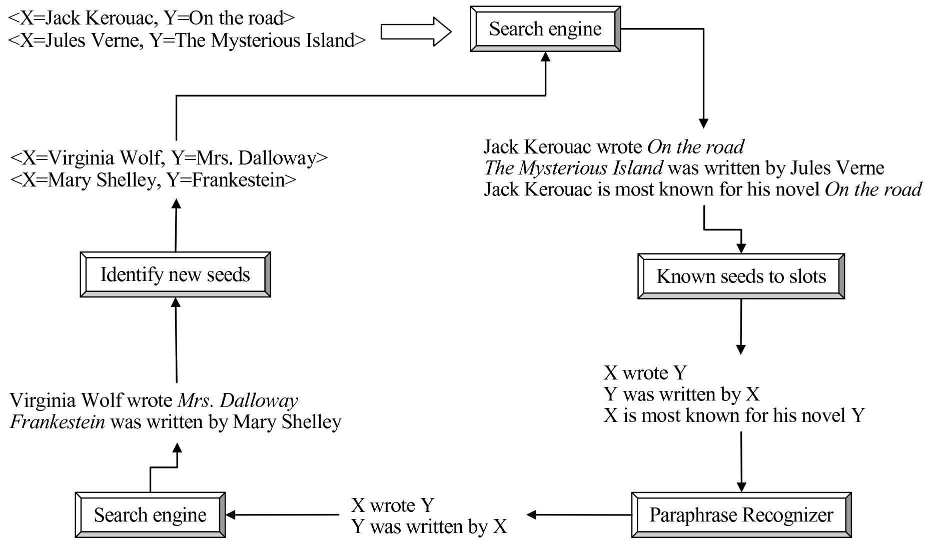

Figure 4: Generating paraphrases of "*X* wrote *Y*" by bootstrapping.

$$macro\text{-}precision = \sum_i p_i, \quad macro\text{-}recall = \sum_i r_i$$

$$micro\text{-}precision = \frac{\sum_i TP_i}{\sum_i (TP_i + FP_i)}, \quad micro\text{-}recall = \frac{\sum_i TP_i}{\sum_i (TP_i + FN_i)}$$

In any case, however, recall cannot be computed in generation, because FN_i is unknown; there are numerous correct paraphrases of an input s_i that may have been missed, and there are even more (if not infinite) language expressions that entail or are entailed by s_i.[36]

Instead of reporting recall, it is common to report (along with precision) the average number of outputs, sometimes called *yield*, defined below, where we assume that there are n test inputs. A better option is to report the yield at different precision levels, since there is usually a tradeoff between the two figures, which is controlled by parameter tuning (e.g., selecting different values of thresholds involved in the methods).

$$yield = \frac{1}{n} \sum_{i=1}^{n} (TP_i + FP_i)$$

Note that if we use a fixed set of test inputs $\{s_i\}$, if we store the sets O_i^{ref} of all the correct outputs that a reference generation method produces for each s_i, and if we treat each O_i^{ref} as the set of all *possible* correct outputs that may be generated for s_i, then both precision and recall can be computed, and without further human effort when a new generation method, say M, is evaluated. FN_i is then the number of outputs in O_i^{ref} that have not been produced for s_i by M; FP_i is the number of M's outputs for s_i that are not in O_i^{ref}; and TP_i is the number of M's outputs for s_i that are included in O_i^{ref}. Callison-Burch et al. (2008) propose an evaluation approach of this kind for what we call paraphrase generation. They use phrase alignment heuristics (Och & Ney, 2003; Cohn et al., 2008)

[36] Accuracy (Section 2.8) is also impossible to compute in this case; apart from not knowing FN_i, the number of outputs that have correctly not been generated (TN_i) is infinite.

to obtain aligned phrases (e.g., "resign", "tender his resignation", "leave office voluntarily") from manually word-aligned sentences with the same meanings (from the Multiple-Translation Chinese Corpus). Roughly speaking, they use as $\{s_i\}$ phrases for which alignments have been found; and for each s_i, O_i^{ref} contains the phrases s_i was aligned to. Since O_i^{ref}, however, contains much fewer phrases than the possible correct paraphrases of s_i, the resulting precision score is a (possibly very pessimistic) lower bound, and the resulting recall scores only measure to what extent M managed to discover the (relatively few) paraphrases in O_i^{ref}, as pointed out by Callison-Burch et al.

To the best of our knowledge, there are no widely adopted benchmark datasets for paraphrase and textual entailment generation, unlike recognition, and comparing results obtained on different datasets is not always meaningful. The lack of generation benchmarks is probably due to the fact that although it is possible to assemble a large collection of input language expressions, it is practically impossible to specify in advance all the numerous (if not infinite) correct outputs a generator may produce, as already discussed. In principle, one could use a paraphrase or textual entailment recognizer to automatically judge if the output of a generator is a paraphrase of, or forms a correct entailment pair with the corresponding input expression. Current recognizers, however, are not yet accurate enough, and automatic evaluation measures from machine translation (e.g., BLEU, Section 2.3) cannot be employed, exactly because their weakness is that they cannot detect paraphrases and textual entailment. An alternative, more costly solution is to use human judges, which also allows evaluating other aspects of the outputs, such as their fluency (Zhao et al., 2009), as in machine translation. One can also evaluate the performance of a generator indirectly, by measuring its impact on the performance of larger natural language processing systems (Section 1.1).

4. Paraphrase and Textual Entailment Extraction

Unlike recognition and generation methods, extraction methods are not given particular input language expressions. They typically process large corpora to extract pairs of language expressions (or templates) that constitute paraphrases or textual entailment pairs. The generated pairs are stored to be used subsequently by recognizers and generators or other applications (e.g., as additional entries of phrase tables in SMT systems). Most extraction methods produce pairs of sentences (or sentence templates) or pairs of shorter expressions. Methods to discover synonyms, hypernym-hyponym pairs or, more generally, entailment relations between words (Lin, 1998a; Hearst, 1998; Moore, 2001; Glickman & Dagan, 2004; Brockett & Dolan, 2005; Hashimoto, Torisawa, Kuroda, De Saeger, Murata, & Kazama, 2009; Herbelot, 2009) can be seen as performing paraphrase or textual entailment extraction restricted to pairs of single words.

4.1 Extraction Methods Based on the Distributional Hypothesis

A possible paraphrase extraction approach is to store all the word n-grams that occur in a large monolingual corpus (e.g., for $n \leq 5$), along with their left and right contexts, and consider as paraphrases n-grams that occur frequently in similar contexts. For example, each n-gram can be represented by a vector showing the words that typically precede or follow the n-gram, with the values in the vector indicating how strongly each word co-occurs with the n-gram; for example, pointwise mutual information values (Manning & Schuetze, 1999) may be used. Vector similarity measures, for example cosine similarity or Lin's measure (1998a), can then be employed to identify n-grams that occur in similar contexts by comparing their vectors.[37] This approach has been shown to be viable with very large monolingual corpora; Pasca and Dienes (2005) used a Web snapshot of approximately a billion Web pages; Bhagat and Ravichandran (2008) used 150 GB of news articles and reported that results deteriorate rapidly with smaller corpora. Even if only lightweight linguistic processing (e.g., POS tagging, without parsing) is performed, processing such large datasets requires very significant processing power, although linear computational complexity is possible with appropriate hashing of the context vectors (Bhagat & Ravichandran, 2008). Paraphrasing approaches of this kind are based on Harris's Distributional Hypothesis (1964), which states that words in similar contexts tend to have similar meanings. The bootstrapping methods of Section 3.2 are based on a similar hypothesis that phrases (or templates) occurring in similar contexts (or with similar slot values) tend to have similar meanings, a hypothesis that can be seen as an extension of Harris's.

Lin and Pantel's (2001) well-known extraction method, called DIRT, is also based on the extended Distributional Hypothesis, but it operates at the syntax level. DIRT first applies a dependency grammar parser to a monolingual corpus. Parsing the corpus is generally time-consuming and, hence, smaller corpora have to be used, compared to methods that do not require parsing; Lin and Panel used 1 GB of news texts in their experiments. Dependency paths are then extracted from the dependency trees of the corpus. Let us consider, for example, sentences (64) and (67). Their dependency trees are shown in Figure 5; the similarity between the two sentences is less obvious than in Figure 1, because of the different verbs that are now involved. Two of the dependency paths that can be extracted from the trees of Figure 5 are shown in (65) and (68). The labels of the edges are augmented by the POS-tags of the words they connect (e.g., N:`subj`:V instead of simply `subj`).[38] The first and last words of the extracted paths are replaced by slots, shown as boxed and numbered POS-tags. Roughly speaking, the paths of (65) and (68) correspond to the surface templates of (66) and (69), respectively, but the paths are actually templates specified at the syntactic level.

(64) A mathematician found a solution to the problem.
(65) $\boxed{N_1}$:`subj`:$V \leftarrow$ found $\rightarrow V$:`obj`:$N \rightarrow$ solution $\rightarrow N$:`to`:$\boxed{N_2}$
(66) N_1 found [a] solution to N_2
(67) The problem was solved by a young mathematician.
(68) $\boxed{N_3}$:`obj`:$V \leftarrow$ solved $\rightarrow V$:`by`:$\boxed{N_4}$
(69) N_3 was solved by N_4.

DIRT imposes restrictions on the paths that can be extracted from the dependency trees; for example, they have to start and end with noun slots. Once the paths have been extracted, it looks for pairs of paths that occur frequently with the same slot fillers. If (65) and (68) occur frequently with the same fillers (e.g., $N_1 = N_4 =$ "mathematician", $N_2 = N3 =$ "problem"), they will be included as a pair in DIRT's output (with $N_1 = N_4$ and $N_2 = N3$). A measure based on mutual information (Manning & Schuetze, 1999; Lin & Pantel, 2001) is used to detect paths with common fillers.

Lin and Pantel call the pairs of templates that DIRT produces "inference rules", but there is no directionality between the templates of each pair; the intention seems to be to produce pairs of near paraphrases. The resulting pairs are actually often textual entailment pairs, not paraphrases,

[37] Zhitomirsky-Geffet and Dagan (2009) discuss a bootstrapping approach, whereby the vector similarity scores (initially computed using pointwise mutual information values in the vectors) are used to improve the values in the vectors; the vector similarity scores are then re-computed.

[38] For consistency with previous examples, we show slightly different labels than those used by Lin and Pantel.

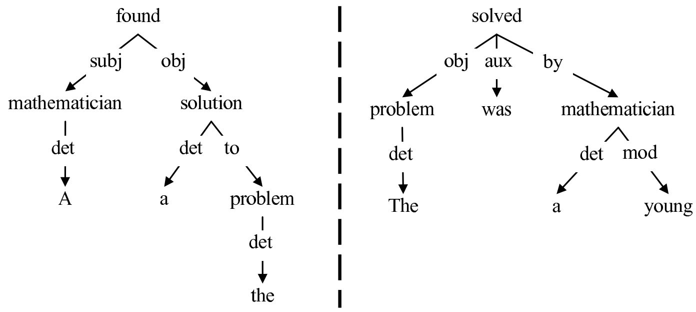

Figure 5: Dependency trees of sentences (64) and (67).

and the directionality of the entailment is unspecified.[39] Bhagat et al. (2007) developed a method, called LEDIR, to classify the template pairs $\langle P_1, P_2 \rangle$ that DIRT and similar methods produce into three classes: (i) paraphrases, (ii) P_1 textually entails P_2 and not the reverse, or (iii) P_2 textually entails P_1 and not the reverse; with the addition of LEDIR, DIRT becomes a method that extracts separately pairs of paraphrase templates and pairs of directional textual entailment templates. Roughly speaking, LEDIR examines the semantic categories (e.g., person, location etc.) of the words that fill P_1 and P_2's slots in the corpus; the categories can be obtained by following WordNet's hypernym-hyponym hierarchies from the filler words up to a certain level, or by applying clustering to the words of the corpus and using the clusters of the filler words as their categories.[40] If P_1 occurs with fillers from a substantially larger number of categories than P_2, then LEDIR assumes P_1 has a more general meaning than P_2 and, hence, P_2 textually entails P_1; similarly for the reverse direction. If there is no substantial difference in the number of categories, P_1 and P_2 are taken to be paraphrases. Szpektor and Dagan (2008) describe a method similar to DIRT that produces textual entailment pairs of *unary* (single slot) templates (e.g., "X takes a nap" $\Rightarrow$ "X sleeps") using a directional similarity measure for unary templates.

Extraction methods based on the (extended) Distributional Hypothesis often produce pairs of templates that are not correct paraphrasing or textual entailment pairs, although they share many common fillers. In fact, pairs involving antonyms are frequent; according to Lin and Pantel (2001), DIRT finds "X solves Y" to be very similar to "X worsens Y"; and the same problem has been reported in experiments with LEDIR (Bhagat et al., 2007) and distributional approaches that operate at the surface level (Bhagat & Ravichandran, 2008).

Ibrahim et al.'s (2003) method is similar to DIRT, but it assumes that a monolingual parallel corpus is available (e.g., multiple English translations of novels), whereas DIRT does not require parallel corpora. Ibrahim et al.'s method extracts pairs of dependency paths only from aligned sentences that share matching *anchors*. Anchors are allowed to be only nouns or pronouns, and they match if they are identical, if they are a noun and a compatible pronoun, if they are of the same semantic category etc. In (70)–(71), square brackets and subscripts indicate matching anchors.[41] The pair of templates of (72)–(73) would be extracted from (70)–(71); for simplicity, we show sentences and templates as surface strings, although the method operates on dependency trees and

[39] Template pairs produced by DIRT are available on-line.

[40] For an introduction to clustering methods, consult chapter 14 of "Foundations of Statistical Natural Language Processing" (Manning & Schuetze, 1999).

[41] Simplified example from Ibrahim et al.'s work (2003).

paths. Matching anchors become matched slots. Heuristic functions are used to score the anchor matches (e.g., identical anchors are preferred to matching nouns and pronouns) and the resulting template pairs; roughly speaking frequently rediscovered template pairs are rewarded, especially when they occur with many different anchors.

(70) The $[\text{clerk}]_1$ liked $[\text{Bovary}]_2$.
(71) $[\text{He}]_1$ was fond of $[\text{Bovary}]_2$.
(72) X liked Y.
(73) X was fond of Y.

By operating on aligned sentences of monolingual parallel corpora, Ibrahim et al.'s method may avoid, to some extent, producing pairs of unrelated templates that simply happen to share common slot fillers; the resulting pairs of templates are also more likely to be paraphrases, rather than simply textual entailment pairs, since they are obtained from aligned sentences of a monolingual parallel corpus. Large monolingual parallel corpora, however, are more difficult to obtain than non-parallel corpora, as already discussed. An alternative is to identify anchors in related sentences from comparable corpora (Section 3.1), which are easier to obtain. Shinyama and Sekine (2003) find pairs of sentences that share the same anchors within clusters of news articles reporting the same event. In their method, anchors are named entities (e.g., person names) identified using a named entity recognizer, or pronouns and noun phrases that refer to named entities; heuristics are employed to identify likely referents. Dependency trees are then constructed from each pair of sentences, and pairs of dependency paths are extracted from the trees by treating anchors as slots.

4.2 Extraction Methods that Use Bootstrapping

Bootstrapping approaches can also be used in extraction, as in generation (Section 3.2), but with the additional complication that there is no particular input template nor seed values of its slots to start from. To address this complication, TEASE (Szpektor et al., 2004) starts with a lexicon of terms of a knowledge domain, for example names of diseases, symptoms etc. in the case of a medical domain; to some extent, such lexicons can be constructed automatically from a domain-specific corpus (e.g., medical articles) via term acquisition techniques (Jacquemin & Bourigault, 2003). TEASE then extracts from a (non-parallel) monolingual corpus pairs of textual entailment templates that can be used with the lexicon's terms as slot fillers. We have already shown a resulting pair of templates, (60)–(61), in Section 3.2; we repeat it as (74)–(75) below. Recall that TEASE does not indicate the directionality of the resulting template pairs, for example whether (74) textually entails (75) or vice versa, but mechanisms like LEDIR (Section 4.1) could be used to guess the directionality.

(74) X prevents Y
(75) X reduces Y risk

Roughly speaking, TEASE first identifies noun phrases that cooccur frequently with each term of the lexicon, excluding very common noun phrases. It then uses the terms and their cooccurring noun phrases as seed slot values to obtain templates, and then the new templates to obtain more slot values, much as in Figure 4. In TEASE, however, the templates are actually slotted dependency paths, and the method includes a stage that merges compatible templates to form more general ones.[42] If particular input templates are provided, TEASE can be used as a generator (Section 3.2).

[42] Template pairs produced by TEASE are available on-line.

Barzilay and McKeown (2001) also used a bootstrapping method, but to extract paraphrases from a *parallel* monolingual corpus; they used multiple English translations of novels. Unlike previously discussed bootstrapping approaches, their method involves two classifiers (in effect, two sets of rules). One classifier examines the words the candidate paraphrases consist of, and a second one examines their contexts. The two classifiers use different feature sets (different views of the data), and the output of each classifier is used to improve the performance of the other one in an iterative manner; this is a case of co-training (Blum & Mitchell, 1998). More specifically, a POS tagger, a shallow parser, and a stemmer are first applied to the corpus, and the sentences are aligned across the different translations. Words that occur in both sentences of an aligned pair are treated as seed positive lexical examples; all the other pairs of words from the two sentences become seed negative lexical examples. From the aligned sentences (76)–(77), we obtain three seed positive lexical examples, shown in (78)–(80), and many more seed negative lexical examples, two of which are shown in (81)–(82).[43] Although seed positive lexical examples are pairs of identical words, as the algorithm iterates new positive lexical examples are produced, and some of them may be synonyms (e.g., "comfort" and "console") or pairs of longer paraphrases, as will be explained below.

(76) He tried to comfort her.
(77) He tried to console Mary.
(78) $\langle expression_1 = \text{“he”}, expression_2 = \text{“he”}, + \rangle$
(79) $\langle expression_1 = \text{“tried”}, expression_2 = \text{“tried”}, + \rangle$
(80) $\langle expression_1 = \text{“to”}, expression_2 = \text{“to”}, + \rangle$
(81) $\langle expression_1 = \text{“he”}, expression_2 = \text{“tried”}, - \rangle$
(82) $\langle expression_1 = \text{“he”}, expression_2 = \text{“to”}, - \rangle$

The contexts of the positive (similarly, negative) lexical examples in the corresponding sentences are then used to construct positive (or negative) context rules, i.e., rules that can be used to obtain new pairs of positive (or negative) lexical examples. Barzilay and McKeown (2001) use the POS tags of the l words before and after the lexical examples as contexts, and in their experiments set $l = 3$. For simplicity, however, let us assume that $l = 2$; then, for instance, from (76)–(77) and the positive lexical example of (79), we obtain the positive context rule of (83). The rule says that if two aligned sentences contain two sequences of words, say λ_1 and λ_2, one from each sentence, and both λ_1 and λ_2 are preceded by the same pronoun, and both are followed by "to" and a (possibly different) verb, then λ_1 and λ_2 are positive lexical examples. Identical subscripts in the POS tags denote identical words; for example, (83) requires both λ_1 and λ_2 to be preceded by the *same* pronoun, but the verbs that follow them may be different.

(83) $\langle left_1 = (pronoun_1), right_1 = (to_1, verb), left_2 = (pronoun_1), right_1 = (to_1, verb), + \rangle$

In each iteration, only the k strongest positive and negative context rules are retained. The strength of each context rule is its precision, i.e., for positive context rules, the number of positive lexical examples whose contexts are matched by the rule divided by the number of both positive and negative lexical examples matched, and similarly for negative context rules. Barzilay and McKeown (2001) used $k = 10$, and they also discarded context rules whose strength was below 95%. The resulting (positive and negative) context rules are then used to identify new (positive and negative)

[43] Simplified example from the work of Barzilay and McKeown (2001).

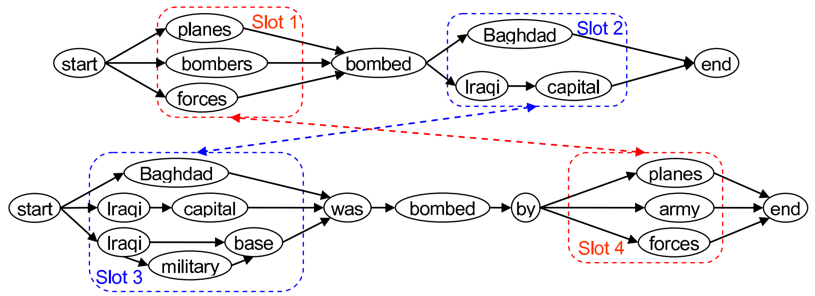

Figure 6: Word lattices obtained from sentence clusters in Barzilay and Lee's method.

lexical examples. From the aligned (84)–(85), the rule of (83) would figure out that "tried" is a synonym of "attempted"; the two words would be treated as a new positive lexical example, shown in (86).

(84) She tried to run away.
(85) She attempted to escape.
(86) $\langle expression_1 = \text{"tried"}, expression_2 = \text{"attempted"}, + \rangle$

The context rules may also produce multi-word lexical examples, like the one shown in (87). The obtained lexical examples are generalized by replacing their words by their POS tags, giving rise to paraphrasing rules. From (87) we obtain the positive paraphrasing rule of (88); again, POS subscripts denote identical words, whereas superscripts denote identical stems. The rule of (88) says that any sequence of words consisting of a verb, "to", and another verb is a paraphrase of any other sequence consisting of the same initial verb, "to", and another verb of the same stem as the second verb of the first sequence, provided that the two sequences occur in aligned sentences.

(87) $\langle expression_1 = \text{"start to talk"}, expression_2 = \text{"start talking"}, + \rangle$
(88) $\langle generalized\_expression_1 = (verb_0, to, verb^1), generalized\_expression_2 = (verb_0, verb^1), + \rangle$

The paraphrasing rules are also filtered by their strength, which is the precision with which they predict paraphrasing contexts. The remaining paraphrasing rules are used to obtain more lexical examples, which are also filtered by the precision with which they predict paraphrasing contexts. The new positive and negative lexical examples are then added to the existing ones, and they are used to obtain, score, and filter new positive and negative context rules, as well as to rescore and filter the existing ones. The resulting context rules are then employed to obtain more lexical examples, more paraphrasing rules, and so on, until no new positive lexical examples can be obtained from the corpus, or a maximum number of iterations is exceeded. Wang et al. (2009) added more scoring measures to Barzilay and McKeown's (2001) method to filter and rank the paraphrase pairs it produces, and used the extended method to extract paraphrases of technical terms from clusters of bug reports.

4.3 Extraction Methods Based on Alignment

Barzilay and Lee (2003) used two corpora of the same genre, but from different sources (news articles from two press agencies). They call the two corpora *comparable*, but they use the term with a slightly different meaning than in previously discussed methods; the sentences of each corpus were clustered separately, and each cluster was intended to contain sentences (from a single corpus) referring to events of the same *type* (e.g., bomb attacks), not sentences (or documents) referring to the same events (e.g., the same particular bombing). From each cluster, a word lattice was produced by aligning the cluster's sentences with Multiple Sequence Alignment (Durbin, Eddy, Krogh, & Mitchison, 1998; Barzilay & Lee, 2002). The solid lines of Figure 6 illustrate two possible resulting lattices, from two different clusters; we omit stop-words. Each sentence of a cluster corresponds to a path in the cluster's lattice. In each lattice, nodes that are shared by a high percentage (50% in Barzilay and Lee's experiments) of the cluster's sentences are considered *backbone nodes*. Parts of the lattice that connect otherwise consecutive backbone nodes are replaced by slots, as illustrated in Figure 6. The two lattices of our example correspond to the surface templates (89)–(90).

(89) X bombed Y.

(90) Y was bombed by X.

The encountered fillers of each slot are also recorded. If two slotted lattices (templates) from different corpora share many fillers, they are taken to be a pair of paraphrases (Figure 6). Hence, this method also uses the extended Distributional Hypothesis (Section 4.1).

Pang et al.'s method (2003) produces finite state automata very similar to Barzilay and Lee's (2003) lattices, but it requires a parallel monolingual corpus; Pang et al. used the Multiple-Translation Chinese Corpus (Section 3.1) in their experiments. The parse trees of aligned sentences are constructed and then merged as illustrated in Figure 7; vertical lines inside the nodes indicate sequences of necessary constituents, whereas horizontal lines correspond to disjunctions.[44] In the example of Figure 7, both sentences consist of a noun phrase (`NP`) followed by a verb phrase (`VP`); this is reflected to the root node of the merged tree. In both sentences, the noun phrase is a cardinal number (`CD`) followed by a noun (`NN`); however, the particular cardinal numbers and nouns are different across the two sentences, leading to leaf nodes with disjunctions. The rest of the merged tree is constructed similarly; consult Pang at al. for further details. Presumably one could also generalize over cardinal numbers, types of named entities etc.

Each merged tree is then converted to a finite state automaton by traversing the tree in a depth-first manner and introducing a ramification when a node with a disjunction is encountered. Figure 8 shows the automaton that corresponds to the merged tree of Figure 7. All the language expressions that can be produced by the automaton (all the paths from the start to the end node) are paraphrases. Hence, unlike other extraction methods, Pang et al.'s (2003) method produces automata, rather than pairs of templates, but the automata can be used in a similar manner. In recognition, for example, if two strings are accepted by the same automaton, they are paraphrases; and in generation, we could look for an automaton that accepts the input expression, and then output other expressions that can be generated by the same automaton. As with Barzilay and Lee's (2003) method, however, Pang et al.'s (2003) method is intended to extract mostly paraphrase, not simply textual entailment pairs.

Bannard and Callison-Burch (2005) point out that bilingual parallel corpora are much easier to obtain, and in much larger sizes, than the monolingual parallel or comparable corpora that some extraction methods employ. Hence, they set out to extract paraphrases from bilingual parallel corpora commonly used in statistical machine translation (SMT). As already discussed in Section 3.1, phrase-based SMT systems employ tables whose entries show how phrases of one language may be

[44] Example from Pang et al.'s work (2003).

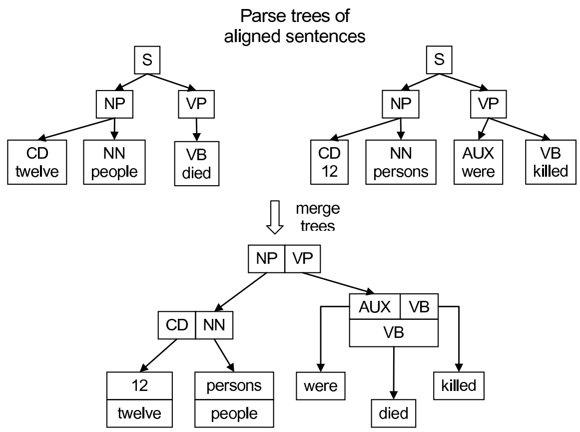

Figure 7: Merging parse trees of aligned sentences in Pang et al.'s method.

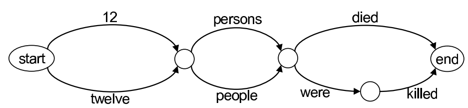

Figure 8: Finite state automaton produced by Pang et al.'s method.

replaced by phrases of another language; phrase tables of this kind may be produced by applying phrase alignment heuristics (Och & Ney, 2003; Cohn et al., 2008) to word alignments produced by the commonly used IBM models. In the case of an English-German parallel corpus, a phrase table may contain entries like the following, which show that "under control" has been aligned with "unter kontrolle" in the corpus, but "unter kontrolle" has also been aligned with "in check"; hence, "under control" and "in check" are a candidate paraphrase pair.[45]

English phrase	German phrase
...	...
under control	unter kontrolle
...	...
in check	unter kontrolle
...	...

More precisely, to paraphrase English phrases, Bannard and Callison-Burch (2005) employ a *pivot language* (German, in the example above) and a bilingual parallel corpus for English and the pivot language. They construct a phrase table from the parallel corpus, and from the table they estimate the probabilities $P(e|f)$ and $P(f|e)$, where e and f range over all of the English and pivot language phrases of the table. For example, $P(e|f)$ may be estimated as the number of entries (rows) that contain both e and f, divided by the number of entries that contain f, if there are multiple rows for multiple alignments of e and f in the corpus, and similarly for $P(f|e)$. The best paraphrase e_2^* of each English phrase e_1 in the table is then computed by equation (91), where f ranges over all the pivot language phrases of the phrase table T.

$$e_2^* = \arg\max_{e_2 \neq e_1} P(e_2|e_1) = \arg\max_{e_2 \neq e_1} \sum_{f \in T} P(f|e_1)P(e_2|f, e_1) \approx \arg\max_{e_2 \neq e_1} \sum_{f \in T} P(f|e_1)P(e_2|f) \tag{91}$$

[45]Example from the work of Bannard and Callison-Burch (2005).

Multiple bilingual corpora, for different pivot languages, can be used; (91) becomes (92), where C ranges over the corpora, and f now ranges over the pivot language phrases of C's phrase table.

$$e_2^* = \arg\max_{e_2 \neq e_1} \sum_C \sum_{f \in T(C)} P(f|e_1)P(e_2|f) \tag{92}$$

Bannard and Callison-Burch (2005) also considered adding a language model (Section 3.1) to their method to favour paraphrase pairs that can be used interchangeably in sentences; roughly speaking, the language model assesses how well one element of a pair can replace the other in sentences where the latter occurs, by scoring the grammaticality of the sentences after the replacement. In subsequent work, Callison-Burch (2008) extended their method to require paraphrases to have the same syntactic types, since replacing a phrase with one of a different syntactic type generally leads to an ungrammatical sentence.[46] Zhou et al. (2006b) employed a method very similar to Bannard and Callison-Burch's to extract paraphrase pairs from a corpus, and used the resulting pairs in SMT evaluation, when comparing machine-generated translations against human-authored ones. Riezler et al. (2007) adopted a similar pivot approach to obtain paraphrase pairs from bilingual phrase tables, and used the resulting pairs as paraphrasing rules to obtain paraphrases of (longer) questions submitted to a QA system; they also used a log-linear model (Section 3.1) to rank the resulting question paraphrases by combining the probabilities of the invoked paraphrasing rules, a language model score of the resulting question paraphrase, and other features.[47]

The pivot language approaches discussed above have been shown to produce millions of paraphrase pairs from large bilingual parallel corpora. The paraphrases, however, are typically short (e.g., up to four or five words), since longer phrases are rare in phrase tables. The methods can also be significantly affected by errors in automatic word and phrase alignment (Bannard & Callison-Burch, 2005). To take into consideration word alignment errors, Zhao et al. (2008) use a log-linear classifier to score candidate paraphrase pairs that share a common pivot phrase, instead of using equations (91) and (92). In effect, the classifier uses the probabilities $P(f|e_1)$ and $P(e_2|f)$ of (91)–(92) as features, but it also uses additional features that assess the quality of the word alignment between e_1 and f, as well as between f and e_2. In subsequent work, Zhao et al. (2009) also consider the English phrases e_1 and e_2 to be paraphrases, when they are aligned to *different* pivot phrases f_1 and f_2, provided that f_1 and f_2 are themselves a paraphrase pair in the pivot language. Figure 9 illustrates the original and extended pivot approaches of Zhao et al. The paraphrase pairs $\langle f_1, f_2 \rangle$ of the pivot language are extracted and scored from a bilingual parallel corpus as in the original approach, by reversing the roles of the two languages. The scores of the $\langle f_1, f_2 \rangle$ pairs, which roughly speaking correspond to $P(f_2|f_1)$, are included as additional features in the classifier that scores the resulting English paraphrases, along with scores corresponding to $P(f_1|e_1)$, $P(e_2|f_2)$, and features that assess the word alignments of the phrases involved.

Zhao et al.'s (2008, 2009) method also extends Bannard and Callison-Burch's (2005) by producing pairs of *slotted* templates, whose slots can be filled in by words of particular parts of speech (e.g., "*Noun*$_1$ is considered by *Noun*$_2$" $\approx$ "*Noun*$_2$ considers *Noun*$_1$").[48] Hence, Zhao et al.'s patterns are more general, but a reliable parser of the language we paraphrase in is required; let us assume again that we paraphrase in English. Roughly speaking, the slots are formed by removing subtrees

[46] An implementation of Callison-Burch's (2008) method and paraphrase rules it produced are available on-line.

[47] Riezler et al. (2007) also employ a paraphrasing method based on an SMT system trained on question-answer pairs.

[48] A collection of template pairs produced by Zhao et al.'s method is available on-line.

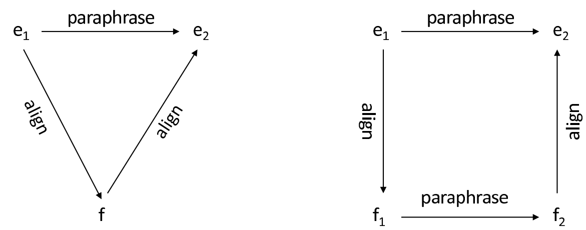

Figure 9: Illustration of Zhao et al.'s pivot approaches to paraphrase extraction.

from the dependency trees of the English sentences and replacing the removed subtrees by the POS tags of their roots; words of the pivot language sentences that are aligned to removed words of the corresponding English sentences are also replaced by slots. A language model is also used, when paraphrases are replaced in longer sentences. Zhao et al.'s experiments show that their method outperforms DIRT, and that it is able to output as many paraphrase pairs as the method of Bannard and Callison-Burch, but with better precision, i.e., fewer wrongly produced pairs. Most of the generated paraphrases (93%), however, contain only one slot, and the method is still very sensitive to word alignment errors (Zhao et al., 2009), although the features that check the word alignment quality alleviate the problem.

Madnani et al. (2007) used a pivot approach similar to Bannard and Callison-Burch's (2005) to obtain synchronous (normally bilingual) English-to-English context-free grammar rules from bilingual parallel corpora. Parsing an English text with the English-to-English synchronous rules automatically paraphrases it; hence the resulting synchronous rules can be used in paraphrase generation (Section 3). The rules have associated probabilities, which are estimated from the bilingual corpora. A log-linear combination of the probabilities and other features of the invoked rules is used to guide parsing. Madnani et al. employed the English-to-English rules to parse and, thus, paraphrase human-authored English reference translations of Chinese texts. They showed that using the additional automatically generated reference translations when tuning a Chinese-to-English SMT system improves its performance, compared to using only the human-authored references.

We note that the alignment-based methods of this section appear to have been used to extract only paraphrase pairs, not (unidirectional) textual entailment pairs.

4.4 Evaluating Extraction Methods

When evaluating extraction methods, we would ideally measure both their precision (what percentage of the extracted pairs are correct paraphrase or textual entailment pairs) and their recall (what percentage of all the correct pairs that could have been extracted have actually been extracted). As in generation, however, recall cannot be computed, because the number of all correct pairs that could have been extracted from a large corpus (by an ideal method) is unknown. Instead, one may again count the number of extracted pairs (the total yield of the method), possibly at different precision levels. Different extraction methods, however, produce pairs of different kinds (e.g., surface strings, slotted surface templates, or slotted dependency paths) from different kinds of corpora (e.g., monolingual or multilingual parallel or comparable corpora); hence, direct comparisons of extraction methods may be impossible. Furthermore, different scores are obtained, depending on whether the extracted pairs are considered in particular contexts or not, and whether they are required to be interchangeable in grammatical sentences (Bannard & Callison-Burch, 2005; Barzilay & Lee, 2003;

Callison-Burch, 2008; Zhao et al., 2008). The output of an extraction method may also include pairs with relatively minor variations (e.g., active vs. passive, verbs vs. nominalizations, or variants such as "the *X* company bought *Y* vs. "*X* bought *Y*"), which may cause methods that produce large numbers of minor variants to appear better than they really are; these points also apply to the evaluation of generation methods (Section 3.3), though they have been discussed mostly in the extraction literature. Detecting and grouping such variants (e.g., turning all passives and nominalizations to active forms) may help avoid this bias and may also improve the quality of the extracted pairs by making the occurrences of the (grouped) expressions less sparse (Szpektor & Dagan, 2007).

As in generation, in principle one could use a paraphrase or textual entailment recognizer to automatically score the extracted pairs. However, recognizers are not yet accurate enough; hence, human judges are usually employed. When extracting slotted textual entailment rules (e.g., "*X* painted *Y*" textually entails "*Y* is the work of *X*"), Szpektor et al. (2007) report that human judges find it easier to agree whether or not particular instantiations of the rules (in particular contexts) are correct or incorrect, as opposed to asking them to assess directly the correctness of the rules. A better evaluation strategy, then, is to show the judges multiple sentences that match the left-hand side of each rule, along with the corresponding transformed sentences that are produced by applying the rule, and measure the percentage of these sentence pairs the judges consider correct textual entailment pairs; this measure can be thought of as the precision of each individual rule. Rules whose precision exceeds a (high) threshold can be considered correct (Szpektor et al., 2007).

Again, one may also evaluate extraction methods indirectly, for example by measuring how much the extracted pairs help in information extraction (Bhagat & Ravichandran, 2008; Szpektor & Dagan, 2007, 2008) or when expanding queries (Pasca & Dienes, 2005), by measuring how well the extracted pairs, seen as paraphrasing rules, perform in phrase alignment in monolingual parallel corpora (Callison-Burch et al., 2008), or by measuring to what extent SMT or summarization evaluation measures can be improved by taking into consideration the extracted pairs (Callison-Burch et al., 2006a; Kauchak & Barzilay, 2006; Zhou et al., 2006b).

5. Conclusions

Paraphrasing and textual entailment is currently a popular research topic. Paraphrasing can be seen as bidirectional textual entailment and, hence, similar methods are often used for both. Although both kinds of methods can be described in terms of logical entailment, they are usually intended to capture human intuitions that may not be as strict as logical entailment; and although logic-based methods have been developed, most methods operate at the surface, syntactic, or shallow semantic level, with dependency trees being a particularly popular representation.

Recognition methods, which classify input pairs of natural language expressions (or templates) as correct or incorrect paraphrases or textual entailment pairs, often rely on supervised machine learning to combine similarity measures possibly operating at different representation levels (surface, syntactic, semantic). More recently, approaches that search for sequences of transformations that connect the two input expressions are also gaining popularity, and they exploit paraphrasing or textual entailment rules extracted from large corpora. The RTE challenges provide a significant thrust to recognition work, and they have helped establish benchmarks and attract more researchers.

Generation methods, meaning methods that generate paraphrases of an input natural language expression (or template), or expressions that entail or are entailed by the input expression, are currently based mostly on bootstrapping or ideas from statistical machine translation. There are fewer

Main ideas discussed	R-TE	R-P	G-TE	G-P	E-TE	E-P
Logic-based inferencing	X	X				
Vector space semantic models		X				
Surface string similarity measures	X	X				
Syntactic similarity measures	X	X				
Similarity measures on symbolic meaning representations	X	X				
Machine learning algorithms	X	X		X		X
Decoding (transformation sequences)	X	X		X		
Word/sentence alignment	X			X		X
Pivot language(s)				X		X
Bootstrapping			X	X	X	X
Distributional hypothesis			X	X	X	X
Synchronous grammar rules				X		X

Table 3: Main ideas discussed and tasks they have mostly been used in. R: recognition; G: generation, E: extraction; TE: textual entailment, P: paraphrasing.

publications on generation, compared to recognition (and extraction), and most of them focus on paraphrasing; furthermore, there are no established challenges or benchmarks, unlike recognition. Nevertheless, generation may provide opportunities for novel research, especially to researchers with experience in statistical machine translation, who may for example wish to develop alignment or decoding techniques especially for paraphrasing or textual entailment generation.

Extraction methods extract paraphrases or textual entailment pairs (also called "rules") from corpora, usually off-line. They can be used to construct resources (e.g., phrase tables or collections of rules) that can be exploited by recognition or generation methods, or in other tasks (e.g., statistical machine translation, information extraction). Many extraction methods are based on the Distributional Hypothesis, though they often operate at different representation levels. Alignment techniques originating from statistical machine translation are recently also popular and they allow existing large bilingual parallel corpora to be exploited. Extraction methods also differ depending on whether they require parallel, comparable, or simply large corpora, monolingual or bilingual. As in generation, most extraction research has focused on paraphrasing, and there are no established challenges or benchmarks.

Table 3 summarizes the main ideas we have discussed per task, and Table 4 lists the corresponding main resources that are typically required. The underlying ideas of generation and extraction methods are in effect the same, as shown in Table 3, even if the methods perform different tasks; recognition work has relied on rather different ideas. Generation and extraction have mostly focused on paraphrasing, as already noted, which is why fewer ideas have been explored in generation and extraction for (unidirectional) textual entailment.

We expect to see more interplay among recognition, generation, and extraction methods in the near future. For example, recognizers and generators may use extracted rules to a larger extent; recognizers may be used to filter candidate paraphrases or textual entailment pairs in extraction or generation approaches; and generators may help produce more monolingual parallel corpora or recognition benchmarks. We also expect to see paraphrasing and textual entailment methods being used more often in larger natural language processing tasks, including question answering, information extraction, text summarization, natural language generation, and machine translation.

Main ideas discussed	Main typically required resources
Logical-based inferencing	Parser producing logical meaning representations, inferencing engine, resources to extract meaning postulates and common sense knowledge from.
Vector space semantic models	Large monolingual corpus, possibly parser.
Surface string similarity measures	Only preprocessing tools, e.g., POS tagger, named-entity recognizer, which are also required by most other methods.
Syntactic similarity measures	Parser.
Similarity measures operating on symbolic meaning representations	Lexical semantic resources, possibly parser and/or semantic role labeling to produce semantic representations.
Machine learning algorithms	Training/testing datasets, components/resources needed to compute features.
Decoding (transformation sequences)	Synonyms, hypernyms-hyponyms, paraphrasing/TE rules.
Word/sentence alignment	Large parallel or comparable corpora (monolingual or multilingual), possibly parser.
Pivot language(s)	Multilingual parallel corpora.
Bootstrapping	Large monolingual corpus, recognizer.
Distributional hypothesis	Monolingual corpus (possibly parallel or comparable).
Synchronous grammar rules	Monolingual parallel corpus.

Table 4: Main ideas discussed and main resources they typically require.

Acknowledgments

We thank the three anonymous reviewers for their valuable comments. This work was funded by the Greek PENED project "Combined research in the areas of information retrieval, natural language processing, and user modeling aiming at the development of advanced search engines for document collections", which was co-funded by the European Union (80%) and the Greek General Secretariat for Research and Technology (20%).

Appendix A. On-line Resources Mentioned

A.1 Bibliographic Resources, Portals, Tutorials

ACL 2007 tutorial on textual entailment: http://www.cs.biu.ac.il/∼dagan /TE-Tutorial-ACL07.ppt.

ACL Anthology: http://www.aclweb.org/anthology/.

Textual Entailment Portal: http://www.aclweb.org/aclwiki/index.php? title=Textual_Entailment_Portal.

A.2 Corpora, Challenges, and their Datasets

Cohn et al.'s paraphrase corpus: Word-aligned paraphrases; http://www.dcs.shef.ac.uk/∼tcohn/paraphrase_corpus.html.

FATE: The RTE-2 dataset with FrameNet annotations; http://www.coli.uni-saarland.de/projects/salsa/fate/.

MSR Paraphrase Corpus: Paraphrase recognition benchmark dataset; http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042/.

Multiple-Translation Chinese Corpus: Multiple English translations of Chinese news articles; http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T01.

RTE challenges, PASCAL Network of Excellence: Textual entailment recognition challenges and their datasets; http://pascallin.ecs.soton.ac.uk/Challenges/.

RTE track of NIST's Text Analysis Conference: Continuation of PASCAL's RTE; http://www.nist.gov/tac/tracks/.

Written News Compression Corpus: Sentence compression corpus; http://jamesclarke.net/research/.

A.3 Implementations of Machine Learning Algorithms

LIBSVM: SVM implementation; http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.

Stanford's Maximum Entropy classifier: http://nlp.stanford.edu/software/index.shtml.

SVM-Light: SVM implementation; http://svmlight.joachims.org/.

Weka: Includes implementations of many machine learning algorithms; http://www.cs.waikato.ac.nz/ml/weka/.

A.4 Implementations of Similarity Measures

EDITS: Suite to recognize textual entailment by computing edit distances; http://edits.fbk.eu/.

WordNet::Similarity: Implementations of WordNet-based similarity measures; http://wn-similarity.sourceforge.net/.

A.5 Parsers, POS Taggers, Named Entity Recognizers, Stemmers

Brill's POS tagger: http://en.wikipedia.org/wiki/Brill_tagger.

Charniak's parser: http://flake.cs.uiuc.edu/∼cogcomp/srl/CharniakServer.tgz.

Collin's parser: http://people.csail.mit.edu/mcollins/code.html.

Link Grammar Parser: http://www.abisource.com/projects/link-grammar/.

MaltParser: http://w3.msi.vxu. se/∼nivre/research/MaltParser.html.

MINIPAR: http://www.cs.ualberta.ca/∼lindek/minipar.htm.

Porter's stemmer: http://tartarus.org/∼martin/PorterStemmer/.

Stanford's named-entity recognizer, parser, tagger: http://nlp.stanford.edu/software/index.shtml.

A.6 Statistical Machine Translation Tools and Resources

Giza++: Often used to train IBM models and align words; http://www.fjoch.com/GIZA++.html.

Koehn's Statistical Machine Translation site: Pointers to commonly used SMT tools, resources; http://www.statmt.org/.

Moses: Frequently used SMT system that includes decoding facilities; http://www.statmt.org/moses/.

SRILM: Commonly used to create language models; http://www.speech.sri.com/projects/srilm/.

A.7 Lexical Resources, Paraphrasing and Textual Entailment Rules

Callison-Burch's paraphrase rules: Paraphrase rules extracted from multilingual parallel corpora via pivot language(s); the implementation of the method used is also available; http://cs.jhu.edu/∼ccb/.

DIRT rules: Template pairs produced by DIRT; http://demo.patrickpantel.com/.

Extended WordNet: Includes meaning representations extracted from WordNet's glosses; http://wordnet.princeton.edu/.

FrameNet: http://framenet.icsi.berkeley.edu/.

Nomlex: English nominalizations of verbs; http://nlp.cs.nyu.edu/nomlex/

TEASE rules: Textual entailment rules produced by TEASE; http://www.cs.biu.ac.il/∼szpekti/TEASE_collection.zip.

VerbNet: http://verbs.colorado.edu/∼mpalmer/projects/verbnet.html.

WordNet: http://xwn.hlt.utdallas.edu/.

Zhao et al.'s paraphrase rules: Paraphrase rules with slots corresponding to POS tags, extracted from multilingual parallel corpora via pivot language(s); http://ir.hit.edu.cn/phpwebsite/index.php?module=documents&JAS_DocumentManager_op=viewDocument&JAS_Document_id=268.

References

Alpaydin, E. (2004). *Introduction to Machine Learning*. MIT Press.

Androutsopoulos, I., Oberlander, J., & Karkaletsis, V. (2007). Source authoring for multilingual generation of personalised object descriptions. *Nat. Lang. Engineering*, *13*(3), 191–233.

Androutsopoulos, I., Ritchie, G. D., & Thanisch, P. (1995). Natural language interfaces to databases – an introduction. *Nat. Lang. Engineering*, *1*(1), 29–81.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proc. of the 17th Int. Conf. on Comp. Linguistics*, pp. 86–90, Montreal, Quebec, Canada.

Bannard, C., & Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proc. of the 43rd Annual Meeting of* ACL, pp. 597–604, Ann Arbor, MI.

Bar-Haim, R., Berant, J., & Dagan, I. (2009). A compact forest for scalable inference over entailment and paraphrase rules. In *Proc. of the Conf. on* EMNLP, pp. 1056–1065, Singapore.

Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., & Szpektor, I. (2006). The 2nd PASCAL recognising textual entailment challenge. In *Proc. of the 2nd* PASCAL *Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

Bar-Haim, R., Dagan, I., Greental, I., & Shnarch, E. (2007). Semantic inference at the lexical-syntactic level. In *Proc. of the 22nd Conf. on Artificial Intelligence*, pp. 871–876, Vancouver, BC, Canada.

Barzilay, R., & Elhadad, N. (2003). Sentence alignment for monolingual comparable corpora. In *Proc. of the Conf. on* EMNLP, pp. 25–32, Sapporo, Japan.

Barzilay, R., & Lee, L. (2002). Bootstrapping lexical choice via multiple-sequence alignment. In *Proc. of the Conf. on* EMNLP, pp. 164–171, Philadelphia, PA.

Barzilay, R., & Lee, L. (2003). Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proc. of the* HLT *Conf. of* NAACL, pp. 16–23, Edmonton, Canada.

Barzilay, R., & McKeown, K. (2001). Extracting paraphrases from a parallel corpus. In *Proc. of the 39th Annual Meeting of* ACL, pp. 50–57, Toulouse, France.

Barzilay, R., & McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Comp. Linguistics*, *31*(3), 297–327.

Bateman, J., & Zock, M. (2003). Natural language generation. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 15, pp. 284–304. Oxford University Press.

Bensley, J., & Hickl, A. (2008). Workshop: Application of LCC's GROUNGHOG system for RTE-4. In *Proc. of the Text Analysis Conference*, Gaithersburg, MD.

Bergmair, R. (2009). A proposal on evaluation measures for RTE. In *Proc. of the* ACL *Workshop on Applied Textual Inference*, pp. 10–17, Singapore.

Berwick, R. C. (1991). Principles of principle-based parsing. In Berwick, R. C., Abney, S. P., & Tenny, C. (Eds.), *Principle-Based Parsing: Computation and Psycholinguistics*, pp. 1–37. Kluwer, Dordrecht, Netherlands.

Bhagat, R., Pantel, P., & Hovy, E. (2007). LEDIR: An unsupervised algorithm for learning directionality of inference rules. In *Proc. of the Conf. on* EMNLP *and the Conf. on Computational Nat. Lang. Learning*, pp. 161–170, Prague, Czech Republic.

Bhagat, R., & Ravichandran, D. (2008). Large scale acquisition of paraphrases for learning surface patterns. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 674–682, Columbus, OH.

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

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proc. of the 11th Annual Conf. on Computational Learning Theory*, pp. 92–100, Madison, WI.

Bos, J., & Markert, K. (2005). Recognising textual entailment with logical inference. In *Proc. of the Conf. on* HLT *and* EMNLP, pp. 628–635, Vancouver, BC, Canada.

Brill, E. (1992). A simple rule-based part of speech tagger. In *Proc. of the 3rd Conf. on Applied Nat. Lang. Processing*, pp. 152–155, Trento, Italy.

Brockett, C., & Dolan, W. (2005). Support Vector Machines for paraphrase identification and corpus construction. In *Proc. of the 3rd Int. Workshop on Paraphrasing*, pp. 1–8, Jeju island, Korea.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., & Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Comp. Linguistics*, *19*(2), 263–311.

Budanitsky, A., & Hirst, G. (2006). Evaluating WordNet-based measures of lexical semantic relatedness. *Comp. Linguistics*, *32*(1), 13–47.

Burchardt, A., & Pennacchiotti, M. (2008). FATE: A FrameNet-annotated corpus for textual entailment. In *Proc. of the 6th Language Resources and Evaluation Conference*, Marrakech, Marocco.

Burchardt, A., Pennacchiotti, M., Thater, S., & Pinkal, M. (2009). Assessing the impact of frame semantics on textual entailment. *Nat. Lang. Engineering*, *15*(4).

Burchardt, A., Reiter, N., Thater, S., & Frank, A. (2007). A semantic approach to textual entailment: System evaluation and task analysis. In *Proc. of the* ACL-PASCAL *Workshop on Textual Entailment and Paraphrasing*, pp. 10–15, Prague, Czech Republic. ACL.

Califf, M., & Mooney, R. (2003). Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, *4*, 177–210.

Callison-Burch, C. (2008). Syntactic constraints on paraphrases extracted from parallel corpora. In *Proc. of the Conf. on* EMNLP, pp. 196–205, Honolulu, HI.

Callison-Burch, C., Cohn, T., & Lapata, M. (2008). ParaMetric: An automatic evaluation metric for paraphrasing. In *Proc. of the 22nd Int. Conf. on Comp. Linguistics*, pp. 97–104, Manchester, UK.

Callison-Burch, C., Dagan, I., Manning, C., Pennacchiotti, M., & Zanzotto, F. M. (Eds.). (2009). *Proc. of the* ACL-IJCNLP *Workshop on Applied Textual Inference*. Singapore.

Callison-Burch, C., Koehn, P., & Osborne, M. (2006a). Improved statistical machine translation using paraphrases. In *Proc. of the* HLT *Conf. of the* NAACL, pp. 17–24, New York, NY.

Callison-Burch, C., Osborne, M., & Koehn, P. (2006b). Re-evaluating the role of BLEU in machine translation research. In *Proc. of the 11th Conf. of* EACL, pp. 249–256, Trento, Italy.

Carnap, R. (1952). Meaning postulates. *Philosophical Studies*, *3*(5).

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proc. of the 1st Conf. of* NAACL, pp. 132–139, Seattle, WA.

Chevelu, J., Lavergne, T., Lepage, Y., & Moudenc, T. (2009). Introduction of a new paraphrase generation tool based on Monte-Carlo sampling. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 249–252, Singapore.

Clarke, D. (2009). Context-theoretic semantics for natural language: an overview. In *Proc. of the* EACL *workshop on Geometrical Models of Nat. Lang. Semantics*, pp. 112–119, Athens, Greece.

Clarke, J., & Lapata, M. (2008). Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research,*, *31*(1), 399–429.

Cohn, T., Callison-Burch, C., & Lapata, M. (2008). Constructing corpora for the development and evaluation of paraphrase systems. *Comp. Linguistics*, *34*(4), 597–614.

Cohn, T., & Lapata, M. (2008). Sentence compression beyond word deletion. In *Proc. of the 22nd Int. Conf. on Comp. Linguistics*, Manchester, UK.

Cohn, T., & Lapata, M. (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, *34*(1), 637–674.

Collins, M. (2003). Head-driven statistical models for natural language parsing. *Comput. Linguistics*, *29*(4), 589–637.

Corley, C., & Mihalcea, R. (2005). Measuring the semantic similarity of texts. In *Proc. of the* ACL *Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pp. 13–18, Ann Arbor, MI.

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Culicover, P. (1968). Paraphrase generation and information retrieval from stored text. *Mechanical Translation and Computational Linguistics*, *11*(1–2), 78–88.

Dagan, I., Dolan, B., Magnini, B., & Roth, D. (2009). Recognizing textual entailment: Rational, evaluation and approaches. *Nat. Lang. Engineering*, *15*(4), i–xvii. Editorial of the special issue on Textual Entailment.

Dagan, I., Glickman, O., & Magnini, B. (2006). The PASCAL recognising textual entailment challenge. In Quiñonero-Candela, J., Dagan, I., Magnini, B., & d'Alche' Buc, F. (Eds.), *Machine Learning Challenges. Lecture Notes in Computer Science*, Vol. 3944, pp. 177–190. Springer-Verlag.

Das, D., & Smith, N. A. (2009). Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 468–476, Singapore.

de Marneffe, M., Rafferty, A., & Manning, C. (2008). Finding contradictions in text. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 1039–1047, Columbus, OH.

Deléger, L., & Zweigenbaum, P. (2009). Extracting lay paraphrases of specialized expressions from monolingual comparable medical corpora. In *Proc. of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, pp. 2–10, Singapore.

Dolan, B., & Dagan, I. (Eds.). (2005). *Proc. of the* ACL *workshop on Empirical Modeling of Semantic Equivalence and Entailment*. Ann Arbor, MI.

Dolan, B., Quirk, C., & Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Eploiting massively parallel news sources. In *Proc. of the 20th Int. Conf. on Comp. Linguistics*, pp. 350–356, Geneva, Switzerland.

Dolan, W. B., & Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *Proc. of the 3rd Int. Workshop on Paraphrasing*, pp. 9–16, Jeju island, Korea.

Dras, M. (1998). Search in constraint-based paraphrasing. In *Proc. of the 2nd Int. Conf. on Natural Lang. Processing and Industrial Applications*, pp. 213–219, Moncton, Canada.

Drass, M., & Yamamoto, K. (Eds.). (2005). *Proc. of the 3rd Int. Workshop on Paraphrasing*. Jeju island, Korea.

Duboue, P. A., & Chu-Carroll, J. (2006). Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proc. of the* HLT *Conf. of* NAACL, pp. 33–36, New York, NY.

Duclaye, F., Yvon, F., & Collin, O. (2003). Learning paraphrases to improve a question-answering system. In *Proc. of the* EACL *Workshop on Nat. Lang. Processing for Question Answering*, pp. 35–41, Budapest, Hungary.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological Sequence Analysis*. Cambridge University Press.

Elhadad, N., & Sutaria, K. (2007). Mining a lexicon of technical terms and lay equivalents. In *Proc. of the Workshop on BioNLP*, pp. 49–56, Prague, Czech Republic.

Erk, K., & Padó, S. (2006). Shalmaneser – a toolchain for shallow semantic parsing. In *Proc. of the 5th Language Resources and Evaluation Conference*, Genoa, Italy.

Erk, K., & Padó, S. (2009). Paraphrase assessment in structured vector space: Exploring parameters and datasets. In *Proc. of the* EACL *Workshop on Geometrical Models of Nat. Lang. Semantics*, pp. 57–65, Athens, Greece.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Finch, A., Hwang, Y. S., & Sumita, E. (2005). Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proc. of the 3rd Int. Workshop on Paraphrasing*, pp. 17–24, Jeju Island, Korea.

Freund, Y., & Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proc. of the 2nd European Conf. on Computational Learning Theory*, pp. 23–37, Barcelona, Spain.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, *28*(2), 337–374.

Fung, P., & Cheung, P. (2004). Multi-level bootstrapping for extracting parallel sentences from a quasi-comparable corpus. In *Proc. of the 20th Int. Conf. on Comp. Linguistics*, pp. 1051–1057, Geneva, Switzerland.

Galanis, D., & Androutsopoulos, I. (2010). An extractive supervised two-stage method for sentence compression. In *Proc. of the* HLT *Conf. of* NAACL, Los Angeles, CA.

Gale, W., & Church, K. (1993). A program for aligning sentences in bilingual corpora. *Comp. Linguistics*, *19*(1), 75–102.

Germann, U., Jahr, M., Knight, K., Marcu, D., & Yamada, K. (2001). Fast decoding and optimal decoding for machine translation. In *Proc. of the 39th Annual Meeting on* ACL, pp. 228–235, Toulouse, France.

Giampiccolo, D., Dang, H., Magnini, B., Dagan, I., & Dolan, B. (2008). The fourth PASCAL recognizing textual entailment challenge. In *Proc. of the Text Analysis Conference*, pp. 1–9, Gaithersburg, MD.

Giampiccolo, D., Magnini, B., Dagan, I., & Dolan, B. (2007). The third PASCAL recognizing textual entailment challenge. In *Proc. of the* ACL-*Pascal Workshop on Textual Entailment and Paraphrasing*, pp. 1–9, Prague, Czech Republic.

Glickman, O., & Dagan, I. (2004). Acquiring lexical paraphrases from a single corpus. In Nicolov, N., Bontcheva, K., Angelova, G., & Mitkov, R. (Eds.), *Recent Advances in Nat. Lang. Processing III*, pp. 81–90. John Benjamins.

Grishman, R. (2003). Information extraction. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 30, pp. 545–559. Oxford University Press.

Habash, N., & Dorr, B. (2003). A categorial variation database for english. In *Proc. of the* HLT *Conf. of* NAACL, pp. 17–23, Edmonton, Canada.

Haghighi, A. D. (2005). Robust textual inference via graph matching. In *Proc. of the Conf. on* EMNLP, pp. 387–394, Vancouver, BC, Canada.

Harabagiu, S., & Hickl, A. (2006). Methods for using textual entailment in open-domain question answering. In *Proc. of the 21st Int. Conf. on Comp. Linguistics and the 44th Annual Meeting of* ACL, pp. 905–912, Sydney, Australia.

Harabagiu, S., Hickl, A., & Lacatusu, F. (2006). Negation, contrast and contradiction in text processing. In *Proc. of the 21st National Conf. on Artificial Intelligence*, pp. 755–762, Boston, MA.

Harabagiu, S., & Moldovan, D. (2003). Question answering. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 31, pp. 560–582. Oxford University Press.

Harabagiu, S. M., Maiorano, S. J., & Pasca, M. A. (2003). Open-domain textual question answering techniques. *Nat. Lang. Engineering*, *9*(3), 231–267.

Harmeling, S. (2009). Inferring textual entailment with a probabilistically sound calculus. *Nat. Lang. Engineering*, *15*(4), 459–477.

Harris, Z. (1964). Distributional Structure. In Katz, J., & Fodor, J. (Eds.), *The Philosphy of Linguistics*, pp. 33–49. Oxford University Press.

Hashimoto, C., Torisawa, K., Kuroda, K., De Saeger, S., Murata, M., & Kazama, J. (2009). Large-scale verb entailment acquisition from the Web. In *Proc. of the Conf. on* EMNLP, pp. 1172–1181, Singapore.

Hearst, M. (1998). Automated discovery of Wordnet relations. In Fellbaum, C. (Ed.), *WordNet: An Electronic Lexical Database*. MIT Press.

Herbelot, A. (2009). Finding word substitutions using a distributional similarity baseline and immediate context overlap. In *Proc. of the Student Research Workshop of the 12th Conf. of* EACL, pp. 28–36, Athens, Greece.

Hickl, A. (2008). Using discourse commitments to recognize textual entailment. In *Proc. of the 22nd Int. Conf. on Comp. Linguistics*, pp. 337–344, Manchester, UK.

Hobbs, J. (1986). Resolving pronoun references. In *Readings in Nat. Lang. Processing*, pp. 339–352. Morgan Kaufmann.

Hovy, E. (2003). Text summarization. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 32, pp. 583–598. Oxford University Press.

Huffman, S. (1995). Learning information extraction patterns from examples. In *Proc. of the* IJCAI *Workshop on New Approaches to Learning for Nat. Lang. Processing*, pp. 127–142, Montreal, Quebec, Canada.

Ibrahim, A., Katz, B., & Lin, J. (2003). Extracting structural paraphrases from aligned monolingual corpora. In *Proc. of the* ACL *Workshop on Paraphrasing*, pp. 57–64, Sapporo, Japan.

Iftene, A. (2008). UAIC participation at RTE4. In *Proc. of the Text Analysis Conference*, Gaithersburg, MD.

Iftene, A., & Balahur-Dobrescu, A. (2007). Hypothesis transformation and semantic variability rules used in recognizing textual entailment. In *Proc. of the* ACL-PASCAL *Workshop on Textual Entailment and Paraphrasing*, pp. 125–130, Prague, Czech Republic.

Inui, K., & Hermjakob, U. (Eds.). (2003). *Proc. of the 2nd Int. Workshop on Paraphrasing: Paraphrase Acquisition and Applications*. Sapporo, Japan.

Jacquemin, C., & Bourigault, D. (2003). Term extraction and automatic indexing. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 33, pp. 599–615. Oxford University Press.

Joachims, T. (2002). *Learning to Classify Text Using Support Vector Machines: Methods, Theory, Algorithms*. Kluwer.

Jurafsky, D., & Martin, J. H. (2008). *Speech and Language Processing* (2nd edition). Prentice Hall.

Kauchak, D., & Barzilay, R. (2006). Paraphrasing for automatic evaluation. In *Proc. of the* HLT *Conf. of* NAACL, pp. 455–462, New York, NY.

Klein, D., & Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proc. of the 41st Annual Meeting of* ACL, pp. 423–430, Sapporo, Japan.

Knight, K., & Marcu, D. (2002). Summarization beyond sentence extraction: A probalistic approach to sentence compression. *Artificial Intelligence*, *139*(1), 91–107.

Koehn, P. (2004). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proc. of the 6th Conf. of the Association for Machine Translation in the Americas*, pp. 115–124, Washington, DC.

Koehn, P. (2009). *Statistical Machine Translation*. Cambridge University Press.

Koehn, P., Och, F. J., & Marcu, D. (2003). Statistical phrase-based translation. In *Proc. of the* HLT *Conf. of* NAACL, pp. 48–54, Edmonton, Canada. ACL.

Kohomban, U., & Lee, W. (2005). Learning semantic classes for word sense disambiguation. In *Proc. of the 43rd Annual Meeting of* ACL, pp. 34–41, Ann Arbor, MI.

Kouylekov, M., & Magnini, B. (2005). Recognizing textual entailment with tree edit distance algorithms. In *Proc. of the* PASCAL *Recognising Textual Entailment Challenge*.

Kubler, S., McDonald, R., & Nivre, J. (2009). *Dependency Parsing*. Synthesis Lectures on HLT. Morgan and Claypool Publishers.

Lappin, S., & Leass, H. (1994). An algorithm for pronominal anaphora resolution. *Comp. Linguistics*, *20*(4), 535–561.

Leacock, C., Miller, G., & Chodorow, M. (1998). Using corpus statistics and WordNet relations for sense identification. *Comp. Linguistics*, *24*(1), 147–165.

Lepage, Y., & Denoual, E. (2005). Automatic generation of paraphrases to be used as translation references in objective evaluation measures of machine translation. In *Proc. of the 3rd Int. Workshop on Paraphrasing*, pp. 57–64, Jesu Island, Korea.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physice-Doklady*, *10*, 707–710.

Lin, D. (1994). PRINCIPAR: an efficient, broad-coverage, principle-based parser. In *Proc. of the 15th Conf. on Comp. Linguistics*, pp. 482–488, Kyoto, Japan. ACL.

Lin, D. (1998a). Automatic retrieval and clustering of similar words. In *Proc. of the the 36th Annual Meeting of* ACL *and 17th Int. Conf. on Comp. Linguistics*, pp. 768–774, Montreal, Quebec, Canada.

Lin, D. (1998b). An information-theoretic definition of similarity. In *Proc. of the 15th Int. Conf. on Machine Learning*, pp. 296–304, Madison, WI. Morgan Kaufmann, San Francisco, CA.

Lin, D. (1998c). An information-theoretic definition of similarity. In *Proc. of the 15th Int. Conf. on Machine Learning*, pp. 296–304, Madison, WI.

Lin, D., & Pantel, P. (2001). Discovery of inference rules for question answering. *Nat. Lang. Engineering*, *7*, 343–360.

Lonneker-Rodman, B., & Baker, C. (2009). The FrameNet model and its applications. *Nat. Lang. Engineering*, *15*(3), 414–453.

MacCartney, B., Galley, M., & Manning, C. (2008). A phrase-based alignment model for natural language inference. In *Proc. of the Conf. on* EMNLP, pp. 802–811, Honolulu, Hawaii.

MacCartney, B., & Manning, C. (2009). An extended model of natural logic. In *Proc. of the 8th Int. Conf. on Computational Semantics*, pp. 140–156, Tilburg, The Netherlands.

Madnani, N., Ayan, F., Resnik, P., & Dorr, B. J. (2007). Using paraphrases for parameter tuning in statistical machine translation. In *Proc. of 2nd Workshop on Statistical Machine Translation*, pp. 120–127, Prague, Czech Republic.

Malakasiotis, P. (2009). Paraphrase recognition using machine learning to combine similarity measures. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, Singapore.

Malakasiotis, P., & Androutsopoulos, I. (2007). Learning textual entailment using SVMs and string similarity measures. In *Proc. of the* ACL-PASCAL *Workshop on Textual Entailment and Paraphrasing*, pp. 42–47, Prague. ACL.

Mani, I. (2001). *Automatic Summarization*. John Benjamins.

Manning, C. D. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Manning, C. D., & Schuetze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT press.

Màrquez, L., Carreras, X., Litkowski, K. C., & Stevenson, S. (2008). Semantic role labeling: an introduction to the special issue. *Comp. Linguistics*, *34*(2), 145–159.

Marton, Y., Callison-Burch, C., & Resnik, P. (2009). Improved statistical machine translation using monolingually-derived paraphrases. In *Proc. of Conf. on* EMNLP, pp. 381–390, Singapore.

McCarthy, D., & Navigli, R. (2009). The English lexical substitution task. *Lang. Resources & Evaluation*, *43*, 139–159.

McDonald, R. (2006). Discriminative sentence compression with soft syntactic constraints. In *Proc. of the 11th Conf. of* EACL, pp. 297–304, Trento, Italy.

McKeown, K. (1983). Paraphrasing questions using given and new information. *Comp. Linguistics*, *9*(1).

Mehdad, Y. (2009). Automatic cost estimation for tree edit distance using particle swarm optimization. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 289–292, Singapore.

Melamed, D. (1999). Bitext maps and alignment via pattern recognition. *Comp. Linguistics*, *25*(1), 107–130.

Melcuk, I. (1987). *Dependency Syntax: Theory and Practice*. State University of New York Press.

Meyers, A., Macleod, C., Yangarber, R., Grishman, R., Barrett, L., & Reeves, R. (1998). Using NOMLEX to produce nominalization patterns for information extraction. In *Proc. of the* COLING-ACL *workshop on the Computational Treatment of Nominals*, Montreal, Quebec, Canada.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 1003–1011, Singapore.

Mirkin, S., Dagan, I., & Shnarch, E. (2009a). Evaluating the inferential utility of lexical-semantic resources. In *Proc. of the 12th Conf. of* EACL, pp. 558–566, Athens, Greece.

Mirkin, S., Specia, L., Cancedda, N., Dagan, I., Dymetman, M., & Szpektor, I. (2009b). Source-language entailment modeling for translating unknown terms. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 791–799, Singapore.

Mitchell, J., & Lapata, M. (2008). Vector-based models of semantic composition. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 236–244, Columbus, OH.

Mitchell, T. (1997). *Machine Learning*. Mc-Graw Hill.

Mitkov, R. (2003). Anaphora resolution. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 14, pp. 266–283. Oxford University Press.

Moens, M. (2006). *Information Extraction: Algorithms and Prospects in a Retrieval Context*. Springer.

Moldovan, D., & Rus, V. (2001). Logic form transformation of WordNet and its applicability to question answering. In *Proc. of the 39th Annual Meeting of* ACL, pp. 402–409, Toulouse, France.

Mollá, D., Schwitter, R., Rinaldi, F., Dowdall, J., & Hess, M. (2003). Anaphora resolution in EXTRANS. In *Proc. of the Int. Symposium on Reference Resolution and Its Applications to Question Answering and Summarization*, pp. 23–25, Venice, Italy.

Mollá, D., & Vicedo, J. (2007). Question answering in restricted domains: An overview. *Comp. Linguistics*, *33*(1), 41–61.

Moore, R. C. (2001). Towards a simple and accurate statistical approach to learning translation relationships among words. In *Proc. of the* ACL *Workshop on Data-Driven Machine Translation*, Toulouse, France.

Moschitti, A. (2009). Syntactic and semantic kernels for short text pair categorization. In *Proc. of the 12th Conf. of* EACL, pp. 576–584, Athens, Greece.

Munteanu, D. S., & Marcu, D. (2006). Improving machine translation performance by exploiting non-parallel corpora. *Comp. Linguistics*, *31*(4), 477–504.

Muslea, I. (1999). Extraction patterns for information extraction tasks: a survey. In *Proc. of the* AAAI *Workshop on Machine Learning for Information Extraction*, Orlando, FL.

Navigli, R. (2008). A structural approach to the automatic adjudication of word sense disagreements. *Nat. Lang. Engineering*, *14*(4), 547–573.

Nelken, R., & Shieber, S. M. (2006). Towards robust context-sensitive sentence alignment for monolingual corpora. In *Proc. of the 11th Conf. of* EACL, pp. 161–168, Trento, Italy.

Nielsen, R., Ward, W., & Martin, J. (2009). Recognizing entailment in intelligent tutoring systems. *Nat. Lang. Engineering*, *15*(4), 479–501.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kuebler, S., Marinov, S., & Marsi, E. (2007). MALTPARSER: a language-independent system for data-driven dependency parsing. *Nat. Lang. Engineering*, *13*(2), 95–135.

Och, F. J., & Ney, H. (2003). A systematic comparison of various stat. alignment models. *Comp. Ling.*, *29*(1), 19–21.

O'Donnell, M., Mellish, C., Oberlander, J., & Knott, A. (2001). ILEX: An architecture for a dynamic hypertext generation system. *Nat. Lang. Engineering*, *7*(3), 225–250.

Padó, S., Galley, M., Jurafsky, D., & Manning, C. D. (2009). Robust machine translation evaluation with entailment features. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 297–305, Singapore.

Padó, S., & Lapata, M. (2007). Dependency-based construction of semantic space models. *Comp. Ling.*, *33*(2), 161–199.

Palmer, M., Gildea, D., & Kingsbury, P. (2005). The Propositional Bank: an annotated corpus of semantic roles. *Comp. Linguistics*, *31*(1), 71–105.

Pang, B., Knight, K., & Marcu, D. (2003). Syntax-based alignment of multiple translations: extracting paraphrases and generating new sentences. In *Proc. of the Human Lang. Techn. Conf. of* NAACL, pp. 102–109, Edmonton, Canada.

Pantel, P., Bhagat, R., Coppola, B., Chklovski, T., & Hovy, E. H. (2007). ISP: Learning inferential selectional preferences. In *Proc. of the* HLT *Conf. of* NAACL, pp. 564–571, Rochester, NY.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting on* ACL, pp. 311–318, Philadelphia, PA.

Pasca, M. (2003). *Open-domain question answering from large text collections* (2nd edition). Center for the Study of Language and Information.

Pasca, M., & Dienes, P. (2005). Aligning needles in a haystack: Paraphrase acquisition across the Web. In *Proc. of the 2nd Int. Joint Conf. on Nat. Lang. Processing*, pp. 119–130, Jeju Island, Korea.

Perez, D., & Alfonseca, E. (2005). Application of the BLEU algorithm for recognizing textual entailments. In *Proc. of the* PASCAL *Challenges Worshop on Recognising Textual Entailment*, Southampton, UK.

Porter, M. F. (1997). An algorithm for suffix stripping. In Jones, K. S., & Willet, P. (Eds.), *Readings in Information Retrieval*, pp. 313–316. Morgan Kaufmann.

Power, R., & Scott, D. (2005). Automatic generation of large-scale paraphrases. In *Proc. of the 3rd Int. Workshop on Paraphrasing*, pp. 73–79, Jesu Island, Korea.

Qiu, L., Kan, M. Y., & Chua, T. (2006). Paraphrase recognition via dissimilarity significance classification. In *Proc. of the Conf. on* EMNLP, pp. 18–26, Sydney, Australia.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Quirk, C., Brockett, C., & Dolan, W. B. (2004). Monolingual machine translation for paraphrase generation. In *Proc. of the Conf. on* EMNLP, pp. 142–149, Barcelona, Spain.

Ravichandran, D., & Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proc. of the 40th Annual Meeting on* ACL, pp. 41–47, Philadelphia, PA.

Ravichandran, D., Ittycheriah, A., & Roukos, S. (2003). Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proc. of the* HLT *Conf. of* NAACL, pp. 85–87, Edmonton, Canada.

Reiter, E., & Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press.

Resnik, P. (1999). Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, *11*, 95–130.

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., & Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Proc. of the 45th Annual Meeting of* ACL, pp. 464–471, Prague, Czech Republic.

Riloff, E. (1996a). Automatically generating extraction patterns from untagged text. In *Proc. of the 13th National Conf. on Artificial Intelligence*, pp. 1044–1049, Portland, OR.

Riloff, E. (1996b). An empirical study of automated dictionary construction for information extraction in three domains. *Artificial Intelligence*, *85*(1–2), 101–134.

Riloff, E., & Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Proc. of the 16th National Conf. on Artificial Intelligence*, pp. 474–479, Orlando, FL.

Rinaldi, F., Dowdall, J., Kaljurand, K., Hess, M., & Molla, D. (2003). Exploiting paraphrases in a question answering system. In *Proc. of the 2nd Int. Workshop in Paraphrasing*, pp. 25–32, Saporo, Japan.

Sato, S., & Nakagawa, H. (Eds.). (2001). *Proc. of the Workshop on Automatic Paraphrasing*. Tokyo, Japan.

Schohn, G., & Cohn, D. (2000). Less is more: active learning with Support Vector Machines. In *Proc. of the 17th Int. Conf. on Machine Learning*, pp. 839–846, Stanford, CA.

Schuler, K. K. (2005). *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, Univ. of Pennsylvania.

Sekine, S., Inui, K., Dagan, I., Dolan, B., Giampiccolo, D., & Magnini, B. (Eds.). (2007). *Proc. of the* ACL-PASCAL *Workshop on Textual Entailment and Paraphrasing*. Prague, Czech Republic.

Sekine, S., & Ranchhod, E. (Eds.). (2009). *Named Entities – Recognition, Classification and Use*. John Benjamins.

Selkow, S. (1977). The tree-to-tree editing problem. *Information Processing Letters*, *6*(6), 184–186.

Shinyama, Y., & Sekine, S. (2003). Paraphrase acquisition for information extraction. In *Proc. of the* ACL *Workshop on Paraphrasing*, Sapporo, Japan.

Siblini, R., & Kosseim, L. (2008). Using ontology alignment for the TAC RTE challenge. In *Proc. of the Text Analysis Conference*, Gaithersburg, MD.

Sleator, D. D., & Temperley, D. (1993). Parsing English with a link grammar. In *Proc. of the 3rd Int. Workshop on Parsing Technologies*, pp. 277–292, Tilburg, Netherlands and Durbuy, Belgium.

Soderland, S. (1999). Learning inf. extraction rules for semi-structured and free text. *Mach. Learning*, *34*(1–3), 233–272.

Soderland, S., Fisher, D., Aseltine, J., & Lehnert, W. G. (1995). CRYSTAL: Inducing a conceptual dictionary. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence*, pp. 1314–1319, Montreal, Quebec, Canada.

Stevenson, M., & Wilks, Y. (2003). Word sense disambiguation. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 13, pp. 249–265. Oxford University Press.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proc. of the 7th Int. Conf. on Spoken Language Processing*, pp. 901–904, Denver, CO.

Szpektor, I., & Dagan, I. (2007). Learning canonical forms of entailment rules. In *Proc. of Recent Advances in Natural Lang. Processing*, Borovets, Bulgaria.

Szpektor, I., & Dagan, I. (2008). Learning entailment rules for unary templates. In *Proc. of the 22nd Int. Conf. on Comp. Linguistics*, pp. 849–856, Manchester, UK.

Szpektor, I., Dagan, I., Bar-Haim, R., & Goldberger, J. (2008). Contextual preferences. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 683–691, Columbus, OH.

Szpektor, I., Shnarch, E., & Dagan, I. (2007). Instance-based evaluation of entailment rule acquisition. In *Proc. of the 45th Annual Meeting of* ACL, pp. 456–463, Prague, Czech Republic.

Szpektor, I., Tanev, H., Dagan, I., & Coppola, B. (2004). Scaling Web-based acquisition of entailment relations. In *Proc. of the Conf. on* EMNLP, Barcelona, Spain.

Tai, K.-C. (1979). The tree-to-tree correction problem. *Journal of* ACM, *26*(3), 422–433.

Tatu, M., Iles, B., Slavick, J., Novischi, A., & Moldovan, D. (2006). COGEX at the second recognizing textual entailment challenge. In *Proc. of the 2nd* PASCAL *Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

Tatu, M., & Moldovan, D. (2005). A semantic approach to recognizing textual entailment. In *Proc. of the Conf. on* HLT *and* EMNLP, pp. 371–378, Vancouver, Canada.

Tatu, M., & Moldovan, D. (2007). COGEX at RTE 3. In *Proc. of the* ACL-PASCAL *Workshop on Textual Entailment and Paraphrasing*, pp. 22–27, Prague, Czech Republic.

Tomuro, N. (2003). Interrogative reformulation patterns and acquisition of question paraphrases. In *Proc. of the 2nd Int. Workshop on Paraphrasing*, pp. 33–40, Sapporo, Japan.

Tong, S., & Koller, D. (2002). Support Vector Machine active learning with applications to text classification. *Machine Learning Research*, *2*, 45–66.

Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of the* HLT *Conf. of* NAACL, pp. 173–180, Edmonton, Canada.

Tsatsaronis, G. (2009). *Word Sense Disambiguation and Text Relatedness Based on Word Thesauri*. Ph.D. thesis, Department of Informatics, Athens University of Economics and Business.

Tsatsaronis, G., Varlamis, I., & Vazirgiannis, M. (2010). Text relatedness based on a word thesaurus. *Artificial Intelligence Research*, *37*, 1–39.

Turney, P., & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Artificial Intelligence Research*, *37*, 141–188.

Vapnik, V. (1998). *Statistical learning theory*. John Wiley.

Vendler, Z. (1967). Verbs and Times. In *Linguistics in Philosophy*, chap. 4, pp. 97–121. Cornell University Press.

Vogel, S., Ney, H., & Tillmann, C. (1996). HMM-based word alignment in statistical translation. In *Proc. of the 16th Conf. on Comp. Linguistics*, pp. 836–841, Copenhagen, Denmark.

Voorhees, E. (2001). The TREC QA track. *Nat. Lang. Engineering*, *7*(4), 361–378.

Voorhees, E. (2008). Contradictions and justifications: Extensions to the textual entailment task. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 63–71, Columbus, OH.

Wan, S., Dras, M., Dale, R., & Paris, C. (2006). Using dependency-based features to take the "parafarce" out of paraphrase. In *Proc. of the Australasian Language Technology Workshop*, pp. 131–138, Sydney, Australia.

Wang, R., & Neumann, G. (2008). An divide-and-conquer strategy for recognizing textual entailment. In *Proc. of the Text Analysis Conference*, Gaithersburg, MD.

Wang, X., Lo, D., Jiang, J., Zhang, L., & Mei, H. (2009). Extracting paraphrases of technical terms from noisy parallel software corpora. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 197–200, Singapore.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

Wu, D. (2000). Alignment. In Dale, R., Moisl, H., & Somers, H. (Eds.), *Handbook of Nat. Lang. Processing*, pp. 415–458. Marcel Dekker.

Wubben, S., van den Bosch, A., Krahmer, E., & Marsi, E. (2009). Clustering and matching headlines for automatic paraphrase acquisition. In *Proc. of the 12th European Workshop on Nat. Lang. Generation*, pp. 122–125, Athens, Greece.

Xu, F., Uszkoreit, H., & Li, H. (2007). A seed-driven bottom-up machine learning framework for extracting relations of various complexity. In *Proc. of the 45th Annual Meeting of the Association of Comp. Linguistics*, pp. 584–591, Prague, Czech Republic.

Yang, X., Su, J., & Tan, C. L. (2008). A twin-candidate model for learning-based anaphora resolution. *Comp. Linguistics*, *34*(3), 327–356.

Yarowski, D. (2000). Word-sense disambiguation. In Dale, R., Moisl, H., & Somers, H. (Eds.), *Handbook of Nat. Lang. Processing*, pp. 629–654. Marcel Dekker.

Zaenen, A., Karttunen, L., & Crouch, R. (2005). Local textual inference: Can it be defined or circumscribed?. In *Proc. of the* ACL *workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pp. 31–36, Ann Arbor, MI.

Zanzotto, F. M., & Dell' Arciprete, L. (2009). Efficient kernels for sentence pair classification. In *Proc. of the Conf. on* EMNLP, pp. 91–100, Singapore.

Zanzotto, F. M., Pennacchiotti, M., & Moschitti, A. (2009). A machine-learning approach to textual entailment recognition. *Nat. Lang. Engineering*, *15*(4), 551–582.

Zhang, K., & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal of Computing*, *18*(6), 1245–1262.

Zhang, Y., & Patrick, J. (2005). Paraphrase identification by text canonicalization. In *Proc. of the Australasian Language Technology Workshop*, pp. 160–166, Sydney, Australia.

Zhang, Y., & Yamamoto, K. (2005). Paraphrasing spoken Chinese using a paraphrase corpus. *Nat. Lang. Engineering*, *11*(4), 417–434.

Zhao, S., Lan, X., Liu, T., & Li, S. (2009). Application-driven statistical paraphrase generation. In *Proc. of the 47th Annual Meeting of* ACL *and the 4th Int. Joint Conf. on Nat. Lang. Processing of* AFNLP, pp. 834–842, Singapore.

Zhao, S., Wang, H., Liu, T., & Li, S. (2008). Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proc. of the 46th Annual Meeting of* ACL*:* HLT, pp. 780–788, Columbus, OH.

Zhitomirsky-Geffet, M., & Dagan, I. (2009). Bootstrapping distributional feature vector quality. *Computational Linguistics*, *35*, 435–461.

Zhou, L., Lin, C.-Y., & Hovy, E. (2006a). Re-evaluating machine translation results with paraphrase support. In *Proc. of the Conf. on* EMNLP, pp. 77–84.

Zhou, L., Lin, C.-Y., Munteanu, D. S., & Hovy, E. (2006b). PARAEVAL: Using paraphrases to evaluate summaries automatically. In *Proc. of the* HLT *Conf. of* NAACL, pp. 447–454, New York, NY.