Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Jeju Island, South Korea, July 2012, pp. 777--789.

Resolving Complex Cases of Definite Pronouns: The Winograd Schema Challenge

Altaf Rahman and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{altaf,vince}@hlt.utdallas.edu

Abstract

We examine the task of resolving complex cases of definite pronouns, specifically those for which traditional linguistic constraints on coreference (e.g., Binding Constraints, gender and number agreement) as well as commonly-used resolution heuristics (e.g., string-matching facilities, syntactic salience) are not useful. Being able to solve this task has broader implications in artificial intelligence: a restricted version of it, sometimes referred to as the Winograd Schema Challenge, has been suggested as a conceptually and practically appealing alternative to the Turing Test. We employ a knowledge-rich approach to this task, which yields a pronoun resolver that outperforms state-of-the-art resolvers by nearly 18 points in accuracy on our dataset.

1 Introduction

Despite the significant amount of work on pronoun resolution in the natural language processing community in the past forty years, the problem is still far from being solved. Its difficulty stems in part from its reliance on sophisticated knowledge sources and inference mechanisms. The sentence pair below, which we will subsequently refer to as the *shout* example, illustrates how difficult the problem can be:

(1a) Ed shouted at Tim because he crashed the car.
(1b) Ed shouted at Tim because he was angry.

The pronoun *he* refers to *Tim* in 1a and *Ed* in 1b. Humans can resolve the pronoun easily, but state-of-the-art coreference resolvers cannot. The reason is that humans have the kind of *world knowledge* needed to resolve the pronouns that machines do not. Our world knowledge tells us that if someone is angry, he may shout at other people. Since *Ed* shouted, he should be the one who was angry. Our world knowledge also tells us that we may shout at someone who made a mistake and that crashing a car is a mistake. Combining these two pieces of evidence, we can easily infer that *Tim* crashed the car.

Our goal in this paper is to examine the resolution of *complex* cases of definite pronouns that appear in sentences exemplified by the *shout* example. Specifically, (1) each sentence has two clauses separated by a discourse connective (i.e., the connective appears *between* the two clauses, just like *because* in the *shout* example), where the first clause contains two or more candidate antecedents (e.g., *Ed* and *Tim*), and the second clause contains the target pronoun (e.g., *he*); and (2) the target pronoun agrees in gender, number, and semantic class with each candidate antecedent, but does not have any overlap in content words with any of them. For convenience, we will refer to the target pronoun that appears in this kind of sentences as a *difficult* pronoun.

Note that many traditional linguistic constraints on coreference are no longer useful for resolving difficult pronouns. For instance, syntactic constraints such as the Binding Constraints will not be useful, since the pronoun and the candidate antecedents appear in different clauses separated by a discourse connective; and constraints concerning agreement in gender, number, and semantic class will not be useful, since the pronoun and the candidate antecedents are compatible with respect to all these attributes. Traditionally important clues provided by various

I(a)	**The city councilmen** refused the demonstrators a permit because *they* feared violence.
I(b)	The city councilmen refused **the demonstrators** a permit because *they* advocated violence.
II(a)	James asked **Robert** for a favor, but *he* refused.
II(b)	**James** asked Robert for a favor, but *he* was refused.
III(a)	**Keith** fired Blaine but *he* did not regret.
III(b)	Keith fired **Blaine** although *he* is diligent.
IV(a)	Emma did not pass the ball to **Janie**, although *she* was open.
IV(b)	**Emma** did not pass the ball to Janie, although *she* should have.
V(a)	Medvedev will cede the presidency to **Putin** because *he* is more popular.
V(b)	**Medvedev** will cede the presidency to Putin because *he* is less popular.

Table 1: Sample twin sentences. The target pronoun in each sentence is italicized, and its antecedent is boldfaced.

string-matching facilities will not be useful either, since the pronoun and its candidate antecedents do not have any words in common.

As in the *shout* example, we ensure that each sentence has a *twin*. Twin sentences were used extensively by researchers in the 1970s to illustrate the difficulty of pronoun resolution (Hirst, 1981). We consider two sentences as twins if (1) they are identical up to and possibly including the discourse connective; and (2) the difficult pronouns in them are lexically identical but have different antecedents. The presence of twins implies that syntactic salience, a commonly-used heuristic in pronoun resolution that prefers the selection of syntactically salient candidate antecedents, may no longer be useful, since the candidate in the subject position is not more likely to be the correct antecedent than the other candidates.

To enable the reader to get a sense of how hard it is to resolve difficult pronouns, Table 1 shows sample twin sentences from our dataset. Note that state-of-the-art pronoun resolvers (e.g., JavaRAP (Qiu et al., 2004), GuiTaR (Poesio and Kabadjov, 2004), as well as those designed by Mitkov (2002) and Charniak and Elsner (2009)) and coreference resolvers (e.g., BART (Versley et al., 2008), CherryPicker (Rahman and Ng, 2009), Reconcile (Stoyanov et al., 2010), the Stanford resolver (Raghunathan et al., 2010; Lee et al., 2011)) cannot accurately resolve the difficult pronouns in these structurally simple sentences, as they do not have the mechanism to capture the fine distinctions between twin sentences. In other words, when given these sentences, the best that the existing resolvers can do to resolve the pronouns is guessing. This could be surprising to a non-coreference researcher, but it is indeed the state of the art.

A natural question is: why do existing resolvers not attempt to handle difficult pronouns? One reason could be that these difficult pronouns do not appear frequently in standard evaluation corpora such as MUC, ACE, and OntoNotes (Bagga, 1998; Haghighi and Klein, 2009). In fact, the Stanford coreference resolver (Lee et al., 2011), which won the CoNLL-2011 shared task on coreference resolution, adopts the once-popular rule-based approach, resolving pronouns simply with rules that encode the aforementioned traditional linguistic constraints on coreference, such as the Binding constraints and gender and number agreement.

The infrequency of occurrences of difficult pronouns in these standard evaluation corpora by no means undermines their significance, however. In fact, being able to automatically resolve difficult pronouns has broader implications in artificial intelligence. Recently, Levesque (2011) has argued that the problem of resolving the difficult pronouns in a carefully chosen set of twin sentences, which he refers to as the Winograd Schema Challenge[1], could serve as a conceptually and practically appealing alternative to the well-known Turing Test (Turing,

[1]Levesque (2011) defines a Winograd Schema as a small reading comprehension test involving the question of which of the two candidate antecedents for the definite pronoun in a given sentence is its correct antecedent. Levesque names this challenge after Winograd because of his pioneering attempt to use a well-known pair of twin sentences — specifically the first pair in Table 1 — to illustrate the difficulty of natural language understanding (Winograd, 1972). Strictly speaking, we are addressing a relaxed version of the Challenge: while Levesque focuses solely on definite pronouns whose resolution requires background knowledge *not* expressed in the words of a sentence, we do not impose such a condition on a sentence.

1950). The reason should perhaps be clear given the above discussion: this is an easy task for a subject who can "understand" natural language but a challenging task for one who can only make intelligent guesses. Levesque believes that "with a very high probability", anything that can resolve correctly a series of difficult pronouns "is thinking in the full-bodied sense we usually reserve for people". Hence, being able to make progress on this task enables us to move one step closer to building an intelligent machine that can truly understand natural language.

To sum up, an important contribution of our work is that it opens up a new line of research involving a problem whose solution requires a deeper understanding of a text. With recent advances in knowledge extraction from text, we believe that time is ripe to tackle this problem. It is worth noting that some researchers have focused on other kinds of anaphors that are hard to resolve, including bridging anaphors (e.g., Poesio et al. (2004)) and anaphors referring to abstract entities, such as those realized by verb phrases in dialogs (e.g., Byron (2002), Strube and Müller (2003), Müller (2007)). Nevertheless, to our knowledge, there has been little work that specifically targets difficult pronouns.

Given the complexity of our task, we investigate a variety of sophisticated knowledge sources for resolving difficult pronouns, and combine them via a machine learning approach. Note that there has been a recent surge of interest in extracting world knowledge from online encyclopedias such as Wikipedia (e.g., Ponzetto and Strube (2006, 2007), Poesio et al. (2007)), YAGO (e.g., Bryl et al. (2010), Rahman and Ng (2011), Uryupina et al. (2011)), and Freebase (e.g., Lee et al. (2011)). However, the resulting extractions are primarily IS-A relations (e.g., *Barack Obama* IS-A *U. S. president*), which would not be useful for resolving definite pronouns.

2 Dataset Creation

We asked 30 undergraduate students who are not affiliated with this research to compose sentence pairs (i.e., twin sentences) that conform to the constraints specified in the introduction. Each student was also asked to annotate the candidate antecedents, the target pronoun, and the correct antecedent for each sentence she composed. Note that a sentence may contain multiple pronouns, but exactly one of them — the one explicitly annotated by its author — is the target pronoun. Each sentence pair was cross-checked by one other student to ensure that it (1) conforms to the desired constraints and (2) does not contain pronouns with ambiguous antecedents (in other words, a human should not be confused as to which candidate antecedent is the correct one). At the end of the process, 941 sentence pairs were considered acceptable, and they formed our dataset. These sentences cover a variety of topics, ranging from real events (e.g., Iran's plan to attack the Saudi ambassador to the U.S.), to events and characters in movies (e.g., Batman and Robin), to purely imaginary situations (e.g., the *shout* example). We partition these sentence pairs into a training set and a test set following a 70/30 ratio.

While not requested by us, the students annotated exactly two candidate antecedents for each sentence. For ease of exposition, we will assume below that there are two candidate antecedents per sentence.

3 Machine Learning Framework

Since our goal is to determine which of the two candidate antecedents is the correct antecedent for the target pronoun in each sentence, our system assumes as input the sentence, the target pronoun, and the two candidate antecedents.

We employ machine learning to combine the features derived from different knowledge sources. Specifically, we employ a *ranking-based* approach. Ranking-based approaches have been shown to outperform their classification-based counterparts (Denis and Baldridge, 2007, 2008; Iida et al., 2003; Yang et al., 2003). Given a pronoun and two candidate antecedents, we aim to train a ranking model that *ranks* the two candidates such that the correct antecedent is assigned a higher rank.

More formally, given training sentence S_k containing target pronoun A_k, correct antecedent C_k and incorrect antecedent I_k, we create two feature vectors, $\mathbf{x}_{CA_k}$ and $\mathbf{x}_{IA_k}$, where $\mathbf{x}_{CA_k}$ is generated from A_k and C_k, and $\mathbf{x}_{IA_k}$ is generated from A_k and I_k. The training set consists of ordered pairs of feature vectors ($\mathbf{x}_{CA_k}$, $\mathbf{x}_{IA_k}$), and the goal of the training procedure is to acquire a ranker that minimizes the number of violations of pairwise rankings provided in the training set. We train this ranker using Joachims' (2002) SVM^{light} package. It is worth noting that we do *not* exploit the fact that each sentence has a twin in training or testing.

After training, the ranker can be applied to the test instances, which are created in the same way as the training instances. For each test instance, the target pronoun is resolved to the higher-ranked candidate antecedent.

Component	# Features	Features
Narrative Chains	1	NC
Google	4	G1, G2, G3, G4
FrameNet	4	FN1, FN2, FN3, FN4
Heuristic Polarity	3	HPOL1, HPOL2, HPOL3
Learned Polarity	3	LPOL1, LPOL2, LPOL3
Connective-Based Relation	1	CBR
Semantic Compat.	3	SC1, SC2, SC3
Lexical Features	68,331	antecedent- independent and dependent features

Table 2: Summary of the features described in Section 4.

4 Linguistic Features

We derive linguistic features for resolving difficult pronouns from eight components, as described below. To enable the reader to keep track of these features more easily, we summarize them in Table 2.

4.1 Narrative Chains

Consider the following sentence:

(2) Ed punished Tim because he tried to escape.

Humans resolve *he* to *Tim* by exploiting the world knowledge that someone who tried to escape is bad and therefore should be punished. Such kind of knowledge can be extracted from *narrative chains*.

Narrative chains are partially ordered sets of events centered around a common *protagonist*, aiming to encode the kind of knowledge provided by *scripts* (Schank and Abelson, 1977). While scripts are hand-written, narrative chains can be learned from unannotated text. Below is a chain learned by Chambers and Jurafsky (2008):

borrow-s invest-s spend-s pay-s raise-s lend-s

As we can see, a narrative chain is composed of a sequence of events (verbs) together with the roles of the protagonist. Here, "s" denotes the subject role, even though a chain can contain a mix of "s" and "o" (the object role). From this chain, we know that the person who borrows something (probably money) may invest, spend, pay, or lend it.

We employ narrative chains to heuristically predict the antecedent for the target pronoun, and encode the prediction as a feature. The heuristic decision procedure operates as follows. Given a sentence, we first determine the event the target pronoun participates in *and* its role in the event. As an example, we determine that in sentence (2) *he* participates in the *try* event and the *escape* event as a *subject*.[2] Second, we determine the event(s) that the candidate antecedents participate in. In (2), both candidate antecedents participate in the *punish* event. Third, we pair each event participated by each candidate antecedent with each event participated by the pronoun. In our example, we would create two pairs, (*punish*, *try-s*) and (*punish*, *escape-s*). Note that *try* and *escape* are associated with the role of the pronoun that we extracted in the first step. Fourth, for each such pair, we extract all the narrative chains containing both elements in the pair from Chambers and Jurafsky's output.[3] This step results in one chain being extracted, which contains punish-o and escape-s. In other words, the protagonist in this chain is the subject of an *escape* event and the object of a *punish* event. Fifth, from the extracted chain, we obtain the role played by the pronoun (i.e., the protagonist) in the event in which the candidate antecedents participate. In our example, the pronoun plays an object role in the *punish* event. Finally, we extract the candidate antecedent that plays the extracted role, which in our example is the second antecedent, *Tim*.[4]

We create a binary feature, NC, which encodes this heuristic decision, and compute its value as follows. Assume in the rest of the paper that i_1 and i_2 are the feature vectors corresponding to the first candidate antecedent and the second candidate an-

[2] Throughout the paper, the subject/object of an event refers to its *deep* rather than *surface* subject/object. We determine the grammatical role of an NP using the Stanford dependency parser (de Marneffe et al., 2006) and a set of simple heuristics.

[3] We employ narrative chains of length 12, which are available from `http://cs.stanford.edu/people/nc/schemas/schemas-size12`.

[4] For an alternative way of using narrative chains for coreference resolution, see Irwin et al. (2011).

tecedent, respectively.[5] For our running example, since *Tim* is predicted to be the antecedent of *he*, the value of NC in i_2 is 1, and its value in i_1 is 0. For notational convenience, we write NC(i_1)=0 and NC(i_2)=1, and will follow this convention when describing the features in the rest of the paper.

Finally, we note that NC(i_1) and NC(i_2) will both be set to zero if (1) the pronoun and the antecedents do not participate in events, or (2) no narrative chains can be extracted in step 4 above, or (3) step 4 enables us to extract more than one chain and these chains indicate that the candidate antecedent can have both a subject role and an object role.

4.2 Google

Consider the following sentences:

(3a) Lions eat zebras because they are predators.
(3b) The knife sliced through the flesh because it was sharp.

Humans resolve *they* to *Lions* in (3a) by exploiting the world knowledge that predators attack and eat other animals. Similarly, humans resolve *it* to *the knife* in (3b) by exploiting the world knowledge that the word *sharp* can be used to describe a knife but not flesh. To acquire this kind of world knowledge, we learn patterns of word usage from the Web by issuing search queries. To facilitate our discussion, let us first introduce some notation. Let a sentence S be denoted by a triple (Z_1, $Conn$, Z_2), where Z_1 and Z_2 are the clauses preceding and following the discourse connective $Conn$, respectively; $A \in Z_2$ be the pronoun governed by the verb V; W be the sequence of words following V in S; and $C_1, C_2 \in Z_1$ be the candidate antecedents.

Given a sentence, we generate four queries: (Q1) C_1V; (Q2) C_2V; (Q3) C_1VW; and (Q4) C_2VW. If v is a verb-to-be followed by an adjective J, we generate two more queries: (Q5) JC_1 and (Q6) JC_2. To exemplify, six queries are generated for (3b): (Q1) "knife was"; (Q2) "flesh was"; (Q3) "knife was sharp"; (Q4) "flesh was sharp"; (Q5) "sharp knife"; and (Q6) "sharp flesh". On the other hand, only four queries are generated for (3a): (Q1) "lions are"; (Q2) "zebras are"; (Q3) "lions are predators"; and (Q4) "zebras are predators".

Using the counts returned by Google for these queries, we create four features, G1, G2, G3, and G4, whose values are determined by Rules 1, 2, 3, and 4, respectively, as described below.

Rule 1: **if** count(Q1) $>$ count(Q2) by at least $x\%$ **then** G1(i_1)=1 and G1(i_2)=0; **else if** count(Q2) $>$ count(Q1) by at least $x\%$ **then** G1(i_2)=1 and G1(i_1)=0; **else** G1(i_1)=G1(i_2)=0.

Rule 2: **if** count(Q3) $>$ count(Q4) by at least $x\%$ **then** G2(i_1)=1 and G2(i_2)=0; **else if** count(Q4) $>$ count(Q3) by at least $x\%$ **then** G2(i_2)=1 and G2(i_1)=0; **else** G2(i_1)=G2(i_2)=0.

Rule 3: **if** count(Q5) $>$ count(Q6) by at least $x\%$ **then** G3(i_1)=1 and G3(i_2)=0; **else if** count(Q6) $>$ count(Q5) by at least $x\%$ **then** G3(i_2)=1 and G3(i_1)=0; **else** G3(i_1)=G3(i_2)=0.

Rule 4: **if** one of G1(i_1) and G1(i_2) is 1, **then** G4(i_1)=G1(i_1) and G4(i_2)=G1(i_2); **else if** one of G2(i_1) and G2(i_2) is 1, **then** G4(i_1)=G2(i_1) and G4(i_2)=G2(i_2); **else if** one of G3(i_1) and G3(i_2) is 1, **then** G4(i_1)=G3(i_1) and G4(i_2)=G3(i_2); **else** G4(i_1)=G4(i_2)=0.

The role of the threshold x should be obvious: it ensures that a heuristic decision is made only if the difference between the counts for the two queries are sufficiently large, because otherwise there is no reason for us to prefer one candidate antecedent to the other. In all of our experiments, we set x to 20.

Note that other researchers have also used lexico-syntactic patterns to generate search queries for bridging anaphora resolution (e.g., Poesio et al. (2004)), other-anaphora resolution (e.g., Modjeska et al. (2003)), and learning selectional preferences for pronoun resolution (e.g., Yang et al. (2005)). However, in each of these three cases, the target *relations* (e.g., the part-whole relation in the case of bridging anaphora resolution, and the subject-verb and verb-object relations in the case of selectional preferences) are specific enough that they can be effectively captured by specific patterns. For example,

[5] The nth candidate antecedent in a sentence is the nth annotated NP encountered when processing the sentence in a left-to-right manner. In sentence (2), *Ed* is the first candidate antecedent and *Tim* is the second.

to determine whether *the wheel* is part of *the car* in bridging anaphora resolution, Poesio et al. employ queries of the form "X of Y", where X and Y would be replaced with *the wheel* and *the car*, respectively. On the other hand, we are not targeting a particular type of relation. Rather, we intend to capture world knowledge like *lions rather than zebras are predators*. Such knowledge may not be expressed as a relation and hence may not be easily captured using specific patterns. For this reason, we need to employ patterns as general as those such as Q3 and Q4.

4.3 FrameNet

If we generate search queries as described in the previous subsection for the *shout* example, it is unlikely that Google will return meaningful counts to us. The reason is that both candidate antecedents in the sentence are proper names belonging to the same type (which in this case is PERSON).

However, in some cases, we may be able to generate more meaningful queries from such kind of sentences. Consider the following sentence:

(4) John killed Jim, so he was arrested.

To generate meaningful queries, we make one observation: *John* and *Jim* played different roles in a *kill* event. Hence, we can replace these proper names with their roles. We propose to obtain these roles from FrameNet (Baker et al., 1998). More generally, for each proper name e in a given sentence, we (1) determine the event in which e is involved (using the Stanford dependency parser); (2) search for the FrameNet frame corresponding to the event as well as e's role in the event; and (3) replace the name with its FrameNet role. In our example, since both names are involved in the *kill* event, we retrieve the FrameNet frame for *kill*. Given that *John* and *Jim* are the subject and object of *kill*, we can extract their semantic roles directly from the frame, which are *killer* and *victim*, respectively.[6] Consequently, we replace the two names with their extracted semantic roles, and generate the search queries from the resulting sentence in the same way as before.

Note that if no frames can be found for the verb in the first clause, no search queries will be generated. After obtaining the query counts, we generate four binary features, FN1, FN2, FN3, FN4, whose values are computed based on the same four heuristic rules that were discussed in the previous subsection.

4.4 Heuristic Polarity

Some sentences involve comparing the two candidate antecedents. Consider the following sentences:

(5a) John was defeated by Jim in the election even though he is more popular.

(5b) John was defeated by Jim in the election because he is more popular.

The pronoun *he* refers to *John* in (5a) and *Jim* in (5b). To see how we can design an algorithm for resolving these pronouns, it would be useful to understand how humans resolve them. The phrase *more popular* has a positive sentiment. In (5a), the use of *even though* yields a clause of concession, which flips the polarity of *more popular* (from positive to negative), whereas in (5b), the use of *because* yields a clause of cause, which does not change the polarity of *more popular* (i.e., *more popular* remains positive). Since *more popular* is used to describe *he*, *he* is "better" in (5b) but "worse" in (5a). Now, the word *defeat* has a positive sentiment, and since *Jim* is the *deep subject* of *defeat*, *Jim* is "better" and *John* is "worse". Finally, in (5b), *he* and *Jim* are "better", so *he* is resolved to *Jim*; on the other hand, in (5a), *he* and *John* are "worse", so *he* is resolved to *John*.

We automate this (human) method for resolving pronouns as follows. We begin by determining whether we can assign a *rank value* (i.e., "better" or "worse") to the pronoun and the two candidate antecedents. For instance, to determine the rank value of the pronoun A, we first determine the polarity value p_A of its *anchor* word w_A, which is either the verb v for which A serves as the deep subject, or the adjective modifying A if v does not exist,[7] using Wilson et al.'s (2005b) subjectivity lexicon.[8] If p_A is not NEUTRAL, we check whether it can be flipped by the context of w_A. We consider three kinds of polarity-reversing context: negation, comparative adverb, and discourse connective. Specifically, we determine whether w_A is negated using the Stanford dependency parser, which explic-

[6] We heuristically map grammatical roles to semantic roles.

[7] In the sentiment analysis and opinion mining literature, (w_A, p_A) is known as an opinion-target pair.

[8] The lexicon contains 8221 words, each of which is hand labeled with a polarity of POSITIVE, NEGATIVE, or NEUTRAL.

itly annotates instances of negation; we determine the existence of a comparative adverb (e.g., "more", "less") using the POS tag "RBR"; and we determine whether A exists in a clause headed by a polarity-reversing connective, such as *although*. After flipping p_A by context, we can infer A's rank value from it. Specifically, A's rank value is "better" if p_A is positive; "worse" if p_A is negative; and "cannot be determined" if p_A is neutral. The polarity values of the two candidate antecedents can be determined in a similar fashion. Note that sometimes we may need to infer rank values. For example, given the sentence "Jane is prettier than Jill", *prettier* has a positive polarity, so its modifying NP, *Jane*, has a "better" rank, and we can infer that *Jill*'s rank is "worse".

We create three features, HPOL1, HPOL2, and HPOL3, based on our heuristic polarity determination component. Specifically, if the rank value of the pronoun or the rank value of one or both of the candidate antecedents cannot be determined, the values of all three binary features will be set to zero for both i_1 and i_2. Otherwise, we compute the values of the three features as follows. To compute HPOL1, which is a binary feature, we (1) employ a heuristic resolution procedure, which resolves the pronoun to the candidate antecedent with the same rank value, and then (2) encode the outcome of this heuristic procedure as the value of HPOL1. For example, since the first candidate antecedent, *John*, is predicted to be the antecedent in (5a), HPOL1(i_1)=1 and HPOL1(i_2)=0. The value of HPOL2 is the concatenation of the polarity values determined for the pronoun and the candidate antecedent. Referring again to (5a), HPOL2(i_1)=positive-positive and HPOL2(i_2)=positive-negative. To compute HPOL3 for a given instance, we simply take its HPOL2 value and append the connective to it. Using (5a) as an example, HPOL3(i_1)=positive-positive-even-though and HPOL3(i_1)=positive-negative-even-though.

4.5 Machine-Learned Polarity

In the previous subsection, we compute the polarity of a word by updating its prior polarity heuristically with contextual information. We hypothesized that polarity could be computed more accurately by employing a sentiment analyzer that can capture richer contextual information. For this reason, we employ OpinionFinder (Wilson et al., 2005a), which has a pre-trained classifier for annotating the phrases in a sentence with their contextual polarity values.

Given a sentence and the polarity values of the phrases annotated by OpinionFinder, we determine the rank values of the pronoun and the two candidate antecedents by mapping them to the polarized phrases using the dependency relations provided by the Stanford dependency parser. We create three binary features, LPOL1, LPOL2, and LPOL3, whose values are computed in the same way as HPOL1, HPOL2, and HPOL3, respectively, except that the computation here is based on the machine-learned polarity values rather than the heuristically determined polarity values.

4.6 Connective-Based Relations

Consider the following sentences:

(6a) Google bought Motorola because they want its customer base.
(6b) Google bought Motorola because they are rich.

Humans resolve *they* to *Google* in (6a) by exploiting the world knowledge that there is a causal relation (signaled by the discourse connective *because*) between the *want* event and the *buy* event. A similar mechanism is used to resolve *they* to *Google* in (6b): from world knowledge we know that there is a causal relation between *rich* and *buy*.

We automate this (human) method for resolving pronouns as follows. First, we gather connective-based relations of this kind from a large, unannotated corpus. In our experiments, we use as our unannotated corpus the documents in three text corpora (namely, BLLIP, Reuters, and English Gigaword), but retain only those sentences that contain a single discourse connective and do not begin with the connective. From these sentences, we collect triples and their frequencies of occurrences in the corpus. Each triple is of the form (V,$Conn$,X), where $Conn$ is a discourse connective, V is a stemmed verb in the clause preceding $Conn$, and X is a stemmed verb or an adjective in the clause following $Conn$. Each triple essentially denotes a relation between V and X expressed by $Conn$. Conceivably, the strength of the relation in a triple increases with its frequency count.

We use the frequency counts of these triples to heuristically predict the correct antecedent for a target pronoun. Given a sentence where $Conn$ is the discourse connective, X is the stemmed verb governing the target pronoun A or the adjective modifying A (if X is a *to be* verb), and V is the stemmed verb governing the candidate antecedents, we retrieve the frequency count of the triple (V,$Conn$,X). If the count is at least 100, we employ a procedure for heuristically selecting the antecedent for the target anaphor. Specifically, if X is a verb, then it resolves the target pronoun to the candidate antecedent that has the same grammatical role as the pronoun. However, if X is an adjective and the sentence does not involve comparison, then it resolves the target pronoun to the candidate antecedent serving as the subject of V.

We create a binary feature, CBR, that encodes this heuristic decision. In our running example, the triple (*buy*, *because*, *want*) occurs 860 times in our corpus, so the pronoun *they* is resolved to the candidate antecedent that occurs as the subject of *buy*. Hence, CBR(i_1)=1 and CBR(i_2)=0. However, had the triple occurred less than 100 times, both of these features would have been set to zero.

4.7 Semantic Compatibility

Some of the queries generated by the Google component, such as Q1 and Q2, aim to capture the semantic compatibility between a candidate antecedent, C, and the verb governing the target pronoun, V. However, using web search queries to estimate semantic compatibility has potential problems, including (1) a *precision* problem: the fact that C and V appear next to each other in a query does not necessarily imply that a subject-verb relation exists between them; and (2) a *recall* problem: these queries fail to capture subject-verb relations where C and V are not immediately adjacent to each other.

To address these potential problems, we compute knowledge of selectional preferences from a large, unannotated corpus. As before, we create our unannotated corpus using the documents in BLLIP, Reuters, and English Gigaword. Specifically, we first parse each sentence in the corpus using the Stanford dependency parser. Then, for each stemmed verb v and each stemmed noun n in the corpus, we collect the following statistics: (1) the number of times n is the subject of v; (2) the number of times n is the direct object of v; (3) the mutual information (MI) of v and n (with n as the subject of v); and (4) the MI of v and n (with n as the direct object of v).[9]

To understand how we use these statistics to generate features for resolving pronouns, consider the following sentence:

(7) The man stole the neighbor's bike because he needed one.

Assuming that the target pronoun and its governing verb V has grammatical relation GR, we create three features, SC1, SC2, and SC3, based on our semantic compatibility component. SC1 encodes the MI value of the head noun of a candidate antecedent and V (and GR). SC2 is a binary feature whose value indicates which of the candidate antecedents has a larger MI value with V (and GR). SC3 is the same as SC2, except that MI is replaced with corpus frequency. In other words, SC2 and SC3 employ different measures to heuristically predict the correct antecedent for the target pronoun. If the target pronoun is governed by a *to be* verb, the values of these three features will all be set to zero.

Given our running example, we first retrieve the following corpus-based statistics: MI(*need*:subj, *man*)=0.6322; MI(*need*:subj, *neighbor*)=0.3975; count(*need*:subj, *man*)=474; and count(*need*:subj, *neighbor*)=68. Using these statistics, we can then compute the aforementioned features for our example. Specifically, SC1(i_1)=0.6322, SC1(i_2)=0.3975, SC2(i_1)=1, SC2(i_2)=0, SC3(i_1)=1, and SC3(i_2)=0.

4.8 Lexical Features

We exploit the coreference-annotated training documents by creating *lexical* features from them. These lexical features can be divided into two categories, depending on whether they are computed based on the candidate antecedents.

Let us begin with the *antecedent-independent* features. Assuming that W is an arbitrary word in a sentence S that is not part of a candidate antecedent and $Conn$ is the connective in S, we create three types of binary-valued antecedent-independent features, namely (1) *unigrams*, where we create one

[9] We use the same formula as described in Section 4.2 of Bergsma and Lin (2006) to compute MI values.

feature for each W; (2) *word pairs*, where we create features by pairing each W appearing before $Conn$ with each W appearing after $Conn$, excluding adjective-noun and noun-adjective pairs[10]; and (3) *word triples*, where we augment each word pair in (2) with $Conn$. The value of each feature f indicates the presence or absence of f in S.

Next, we compute the *antecedent-dependent* features. Let (1) H_{C_1} and H_{C_2} be the head words of candidate antecedents C_1 and C_2, respectively; (2) V_{C_1}, V_{C_2}, and V_A be the verbs governing C_1, C_2, and the target pronoun A, respectively; and (3) J_{C_1}, J_{C_2}, and J_A be the adjectives modifying C_1, C_2, and A, respectively.[11] We create from each candidate antecedent four features, each of which is a word pair. From C_1, we create (H_{C_1}, V_{C_1}), (H_{C_1}, J_{C_1}), (H_{C_1}, V_A), and (H_{C_1}, J_A), all of which will appear in the feature vector corresponding to C_1. A similar set of four features are created from C_2. These antecedent-dependent features are all binary-valued.

It is worth mentioning that while we also considered word triples in the connective-based relations component and word pairs in the semantic compatibility component, in those components we determine their usefulness in an unsupervised manner, whereas by employing them as lexical features we determine their usefulness in a supervised manner.

5 Evaluation

5.1 Experimental Setup

Dataset. We report results on the test set, which comprises 30% of our hand-annotated sentence pairs (see Section 2 for details).

Evaluation metrics. Results are expressed in terms of accuracy, which is the percentage of correctly resolved target pronouns. We also report the percentages of these pronouns that are (1) not resolved and (2) incorrectly resolved.

5.2 Results and Discussion

The Random baseline. Our first baseline is a resolver that randomly guesses the antecedent for the target pronoun in each sentence. Since there are two candidate antecedents per sentence, the Random baseline should achieve an accuracy of 50%.

The Stanford resolver. Our second baseline is the Stanford resolver (Lee et al., 2011), which achieves the best performance in the CoNLL 2011 shared task (Pradhan et al., 2011). As a rule-based resolver, it does not exploit any coreference-annotated data.

Recall from Section 3 that our system assumes as input not only a sentence containing a target pronoun but also the two candidate antecedents. To ensure a fair comparison, the same input is provided to this and other baselines. Hence, if the Stanford resolver decides to resolve the target pronoun, it will resolve it to one of the two candidate antecedents. However, if it does not have enough confidence about resolving it, it will leave it unresolved. Its performance on the test set is shown in the "Unadjusted Scores" column in row 1 of Table 3. As we can see, it correctly resolves 40.1% of the pronouns, incorrectly resolves 29.8% of them, and does not make any decision on the remaining 30.1%.

Given that the Random baseline correctly resolves 50% of pronouns and the Stanford resolver correctly resolves only 40.1% of the pronouns, it is tempting to conclude that Stanford does not perform as well as Random. However, recall that Stanford leaves 30.1% of the pronouns unresolved. Hence, to ensure a fairer comparison, we produce "adjusted" scores for the Stanford resolver, where we "force" it to resolve all of the unresolved target pronouns by assuming that probabilistically half of them will be resolved correctly. This adjusted score is shown in the "Adjusted Scores" column in row 1 of Table 3. As we can see, Stanford achieves an accuracy of 55.1%, which is 5.1 points higher than that of Random.

The Baseline Ranker. To understand whether the somewhat unsatisfactory Stanford results can be attributed to its inability to exploit the training data, we employ as our third baseline a mention ranker that is trained in the same way as our system (see Section 3), except that it employs 39 commonly-used linguistic features for learning-based coreference resolution (see Table 1 of Rahman and Ng (2009) for a description of these features). Hence, the performance difference between this Baseline Ranker and our system can be attributed entirely

[10]Pairing an adjective A in one clause with a noun N in another clause may mislead the learner into thinking that N is modified by A, and hence we do not create such pairs.

[11]If C_1, C_2, and A are not modified by adjectives, no adjective-based features will be created.

		Unadjusted Scores			Adjusted Scores		
	Coreference System	Correct	Wrong	No Decision	Correct	Wrong	No Decision
1	Stanford	40.07%	29.79%	30.14%	55.14%	44.86%	0.00%
2	Baseline Ranker	47.70%	47.16%	5.14%	50.27%	49.73%	0.00%
3	Stanford+Baseline Ranker	53.49%	43.12%	3.39%	55.19%	44.77%	0.00%
4	Our system	**73.05%**	26.95%	0.00%	**73.05%**	26.95%	0.00%

Table 3: Results of the Stanford resolver, the Baseline Ranker, the Combined resolver, and our system.

to the difference between the two linguistic feature sets. Results of the Baseline Ranker are shown in row 2 of Table 3. Before score adjustment, it correctly resolves 47.7% of the target pronouns, incorrectly resolves 47.2% of them, and leaves the remaining 5.1% unresolved. (Note that we output "no decision" if the ranker assigns the same rank value to both candidate antecedents.) After score adjustment, its accuracy is 50.3%, which is 0.3 points higher than that of Random but statistically indistinguishable from it.[12] On the other hand, its accuracy is 4.9 points lower than that of Stanford, and the difference between their performance is significant. While it seems somewhat surprising that a supervised resolver does not perform as well as a rule-based resolver, neither of them employs knowledge sources that are particularly useful for our dataset. In other words, despite given access to annotated data, the Baseline Ranker may not be able to make effective use of it due to the lack of useful features.

The Combined resolver. We create a fourth baseline by combining the Stanford resolver and the Baseline Ranker. The motivation is that the former can provide better precision and the latter can provide better recall by handling "no decision" cases not covered by the former. Note that the Baseline Ranker will be applied to resolve only those pronouns that are left unresolved by Stanford. Results in row 3 of Table 3 show that the adjusted accuracy of this Combined resolver is 55.2%, which is statistically indistinguishable from Stanford's adjusted accuracy. Hence, these results show that the addition of the Baseline Ranker does not help improve Stanford's resolution accuracy.

Our system. Results of our system, which is trained using the features described in Section 4 in combination with a ranking model, are shown in row 4 of Table 3. As we can see, our system achieves an accuracy of 73.1%, significantly outperforming the Combined resolver by 17.9 points in accuracy. These results suggest that our features are more useful for resolving difficult pronouns than those commonly used for coreference resolution.

[12]All statistical significance test results in this paper are obtained using the paired t-test, with $p < 0.05$.

Feature Type	Correct	Wrong	No Decision
All features	73.05%	26.95%	0.00%
−Narrative Chains	68.97%	31.03%	0.00%
−Google	65.96%	34.04%	0.00%
−FrameNet	72.16%	27.84%	0.00%
−Heuristic Polarity	71.45%	28.55%	0.00%
−Learned Polarity	72.70%	27.30%	0.00%
−Connective-Based Rel.	71.28%	28.72%	0.00%
−Semantic Compat.	71.81%	28.19%	0.00%
−Lexical Features	60.11%	25.35%	14.54%

Table 4: Results of feature ablation experiments.

5.3 Feature Analysis

In an attempt to gain additional insight into the performance contribution of each of the eight types of features used in our system, we conduct feature ablation experiments. The unadjusted scores of these experiments are shown in Table 4, where each row shows the performance of the model trained on all types of features except for the one shown in that row. For easy reference, the performance of the model trained on all types of features is shown in row 1 of the table.

A few points deserve mention. First, performance drops significantly whichever feature type is removed. This suggests that all eight feature types are contributing positively to overall accuracy. Second, the *Google*-based features and the *Lexical Features* are the most useful, and those generated via *FrameNet* and *Learned Polarity* are the least useful *in the presence of* other feature types. While it is somewhat surprising that *Learned Polarity* is not more useful than *Heuristic Polarity*, we speculate the reason can be attributed to the fact that the corpus on which OpinionFinder was trained was quite different from ours. Finally, even without using the

Feature Type	Correct	Wrong	No Decision
Narrative Chains	30.67%	24.47%	44.86%
Google	33.16%	7.09%	59.75%
FrameNet	7.27%	4.08%	88.65%
Learned Polarity	4.79%	2.66%	92.55%
Heuristic Polarity	7.27%	1.77%	90.96%
Connective-Based Rel.	14.01%	8.69%	77.30%
Semantic Compat.	23.58%	13.12%	63.30%
Lexical Features	56.91%	43.09%	0.00%

Table 5: Results of single-feature coreference models.

Lexical Features, our system still outperforms all the baseline resolvers: as can been implied from the last row of Table 4, in the absence of the *Lexical Features*, our resolver achieves an adjusted accuracy of 67.4%, which is only 5.7 points less than that obtained when the full feature set is employed. Hence, while the *Lexical Features* are useful, their importance should not be over-emphasized.

To get a better idea of the utility of each feature type, we conduct another experiment in which we train eight models, each of which employs exactly one type of features. Their unadjusted scores are shown in Table 5. As we can see, *Learned Polarity* has the smallest contribution, whereas the *Lexical Features* have the largest contribution.

5.4 Error Analysis

While our resolver significantly outperforms state-of-the-art resolvers, there is a lot of room for improvement. To help direct future research on the resolution of difficult pronouns, we analyze the major sources of errors made by our resolver.

Our analysis reveals that many of the errors correspond to cases that cannot be handled by any of the eight components of our resolver. To understand these cases, consider first the strengths and weaknesses of *Narrative Chains* and *Google*, the two components that contribute the most to overall performance after *Lexical Features*.

Google is especially good at capturing facts, such as *lions are predators* and *zebras are not predators*, helping us correctly resolve sentences such as (5a) and (5b), as well as those in sentence pair (I) in Table 1. However, it may not be good at handling pronouns whose resolution requires an understanding of the connection between the facts or events described in the two clauses of a sentence. The reason is that establishing such a connection requires that we construct a search query composed of information extracted from both clauses, and the resulting, possibly long, query is likely to receive no hit count due to data sparseness. Investigating how to construct such queries while avoiding data sparseness would be an interesting line of future work.

Narrative chains, on the other hand, are useful for capturing the relationship between the events described in the two clauses. However, they are computed over verbs, and therefore cannot capture such a relationship when one or both of the events involved are not described by verbs. For example, narrative chains fail to capture the causal relation between the event expressed by *angry* and *shout* in sentence (1b). It is also worth mentioning that some pronouns that could have been resolved using narrative chains are not owing to the *coverage* and *accuracy* of Chambers and Jurafsky's (2008) chains, but we believe that these recall and precision problems could be addressed by (1) inducing chains from a larger corpus and (2) using semantic roles rather than grammatical roles in the induction process.

Some resolution errors arise from errors in polarity analysis. This can be attributed to the simplicity of our *Heuristic Polarity* component: determining the polarity of a word based on its prior polarity is too naïve. Fine-grained polarity analysis would be a promising solution to this problem (see Pang and Lee (2008) and Liu (2012) for related work).

6 Conclusions

We investigated the resolution of complex cases of definite pronouns, a problem that was under extensive discussion by coreference researchers in the 1970s but has received revived interest owing in part to its relevance to the Turing Test. Our experimental results indicate that it is a challenge for state-of-the-art resolvers, and while we proposed new knowledge sources for addressing this challenge, our resolver still has a lot of room for improvement. In particular, our error analysis indicates that further gains could be achieved via more accurate sentiment analysis and induction of world knowledge from corpora or the Web. In addition, we plan to integrate our resolver into a general-purpose coreference system and evaluate the resulting resolver on standard evaluation corpora such as MUC, ACE, and OntoNotes.

Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-0812261 and IIS-1147644.

References

Amit Bagga. 1998. *Coreference, Cross-Document Coreference, and Information Extraction Methodologies*. Ph.D. thesis, Duke University.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, pages 86–90.

Shane Bergsma and Dekang Lin. 2006. Bootstrapping path-based pronoun resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 33–40.

Volha Bryl, Claudio Guiliano, Luciano Serafini, and Kateryna Tymoshenko. 2010. Using background knowledge to support coreference resolution. In *Proceedings of the 19th European Conference on Artificial Intelligence*, pages 759–764.

Donna K. Byron. 2002. Resolving pronominal reference to abstract entities. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 80–87.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 787–797.

Eugene Charniak and Micha Elsner. 2009. EM works for pronoun anaphora resolution. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 148–156.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 449–454.

Pascal Denis and Jason Baldridge. 2007. A ranking approach to pronoun resolution. In *Proceedings of the Twentieth International Conference on Artificial Intelligence*, pages 1588–1593.

Pascal Denis and Jason Baldridge. 2008. Specialized models and ranking for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 660–669.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1152–1161.

Graeme Hirst. 1981. *Anaphora in Natural Language Understanding*. Springer Verlag.

Ryu Iida, Kentaro Inui, Hiroya Takamura, and Yuji Matsumoto. 2003. Incorporating contextual cues in trainable models for coreference resolution. In *Proceedings of the EACL Workshop on The Computational Treatment of Anaphora*.

Joseph Irwin, Mamoru Komachi, and Yuji Matsumoto. 2011. Narrative schema as world knowledge for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 86–92.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34.

Hector J. Levesque. 2011. The Winograd Schema Challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

Ruslan Mitkov, Richard Evans, and Constantin Orasan. 2002. A new, fully automatic version of Mitkov's knowledge-poor pronoun resolution method. In Al. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, pages 169–187. Springer.

Natalia N. Modjeska, Katja Markert, and Malvina Nissim. 2003. Using the web in machine learning for other-anaphora resolution. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 176–183.

Christoph Müller. 2007. Resolving it, this, and that in unrestricted multi-party dialog. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 816–823.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2(1–2):1–135.

Massimo Poesio and Mijail A. Kabadjov. 2004. A general-purpose, off-the-shelf anaphora resolution module: Implementation and preliminary evaluation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 663–666.

Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. 2004. Learning to resolve bridging references. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 143–150.

Massimo Poesio, David Day, Ron Artstein, Jason Duncan, Vladimir Eidelman, Claudio Giuliano, Rob Hall, Janet Hitzeman, Alan Jern, Mijail Kabadjov, Stanley Yong Wai Keong, Gideon Mann, Alessandro Moschitti, Simone Ponzetto, Jason Smith, Josef Steinberger, Michael Strube, Jian Su, Yannick Versley, Xiaofeng Yang, and Michael Wick. 2007. ELERFED: Final report of the research group on Exploiting Lexical and Encyclopedic Resources For Entity Disambiguation. Technical report, Summer Workshop on Language Engineering, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the Human Language Technology Conference and Conference of the North American Chapter of the Association for Computational Linguistics*, pages 192–199.

Simone Paolo Ponzetto and Michael Strube. 2007. Knowledge derived from Wikipedia for computing semantic relatedness. *Journal of Artificial Intelligence Research*, 30:181–212.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 Shared Task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–27.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2004. A public reference implementation of the RAP anaphora resolution algorithm. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 291–294.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 968–977.

Altaf Rahman and Vincent Ng. 2011. Coreference resolution with world knowledge. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 814–824.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010. RECONCILE: A coreference resolution research platform. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 156–161.

Michael Strube and Christoph Müller. 2003. A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 168–175.

Alan M. Turing. 1950. Computing machinery and intelligence. *Mind*, 59:433–460.

Olga Uryupina, Massimo Poesio, Claudio Giuliano, and Kateryna Tymoshenko. 2011. Disambiguation and filtering methods in using Web knowledge for coreference resolution. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference*, pages 317–322.

Yannick Versley, Simone Paolo Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang, and Alessandro Moschitti. 2008. BART: A modular toolkit for coreference resolution. In *Proceedings of the ACL-08: HLT Demo Session*, pages 9–12.

Theresa Wilson, Paul Hoffmann, Swapna Somasundaran, Jason Kessler, Janyce Wiebe, Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005a. Opinionfinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP 2005 Interactive Demonstrations*, pages 34–35.

Theresa Wilson, Janyce M. Wiebe, and Paul Hoffmann. 2005b. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Joint Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 347–354.

Terry Winograd. 1972. *Understanding Natural Language*. Academic Press, Inc., New York.

Xiaofeng Yang, Guodong Zhou, Jian Su, and Chew Lim Tan. 2003. Coreference resolution using competitive learning approach. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 176–183.

Xiaofeng Yang, Jian Su, and Chew Lim Tan. 2005. Improving pronoun resolution using statistics-based semantic compatibility information. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 165–172.