Learning the Information Status of Noun Phrases in Spoken Dialogues

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

An entity in a dialogue may be old, new, or mediated/inferrable with respect to the hearer's beliefs. Knowing the information status of the entities participating in a dialogue can therefore facilitate its interpretation. We address the under-investigated problem of automatically determining the information status of discourse entities. Specifically, we extend Nissim's (2006) machine learning approach to information-status determination with lexical and structured features, and exploit learned knowledge of the information status of each discourse entity for coreference resolution. Experimental results on a set of Switchboard dialogues reveal that (1) incorporating our proposed features into Nissim's feature set enables our system to achieve state-of-the-art performance on information-status classification, and (2) the resulting information can be used to improve the performance of learning-based coreference resolvers.

1 Introduction

Information status is not a term unfamiliar to researchers working on discourse processing problems. It describes the extent to which a discourse entity, which is typically a noun phrase (NP), is *available* to the hearer given the speaker's assumptions about the hearer's beliefs. According to Nissim et al. (2004), a discourse entity can be *new*, *old*, or *mediated*. Informally, a discourse entity is (1) *old* to the hearer if it is known to the hearer and has previously been referred to in the dialogue, (2) *new* if it is unknown to her and has not been previously referred to; and (3) *mediated* if it is newly mentioned in the dialogue but she can infer its identity from a previously-mentioned entity. Information status is a subject that has received a lot of attention in theoretical linguistics (Halliday, 1976; Prince, 1981; Hajičová, 1984; Vallduví, 1992; Steedman, 2000).

Knowing the information status of discourse entities can potentially benefit many NLP applications. One such task is anaphora resolution. While there is general belief that definite descriptions are mostly anaphoric, Vieira and Poesio (2000) empirically show that only 30% of these NPs are anaphoric. Without being able to determine whether an NP is anaphoric, an anaphora resolver will attempt to resolve every NP, potentially damaging its precision. Since *new* entities are by definition new to the hearer and therefore cannot refer to a previously-introduced NP, knowledge of information status could be used to improve anaphora resolution.

Despite the potential usefulness of information status in NLP tasks, there has been little work on *learning* the information status of discourse entities. To investigate the plausibility of learning information status, Nissim et al. (2004) annotate a set of Switchboard dialogues with such information[1], and subsequently present a rule-based approach and a learning-based approach to acquiring such knowledge from the manual annotations (Nissim, 2006).

Our goals in this paper are two-fold. First, we describe a learning approach to the under-studied problem of determining the information status of discourse entities that extends Nissim's (2006) feature set with two novel types of features: lexical features and structured features based on syntactic parse trees. Second, we employ the automatically

[1]These and other linguistic annotations on the Switchboard dialogues were later released by the LDC as part of the NXT corpus, which is described in detail in Calhoun et al. (2010).

acquired knowledge of information status for coreference resolution. Experimental results on Nissim et al.'s (2004) corpus of Switchboard dialogues show that (1) adding our linguistic features to Nissim's feature set enables our system to outperform her system by 8.1% in F-measure, and (2) learned knowledge of information status can be used to improve coreference resolvers by 1.1–2.6% in F-measure.

The rest of this paper is organized as follows. We first illustrate with examples the concepts of *new*, *old*, and *mediated* entities. Then, we describe the dataset and the feature set that Nissim (2006) used in her approach. After that, we introduce our lexical and structured features. Finally, we evaluate the determination of information status as a standalone task and in the context of coreference resolution.

2 Old, New, and Mediated Entities

Since the concepts of *old*, *new*, and *mediated* entities are not widely known to researchers working outside the area of discourse processing, in this section we will explain them in more detail.

The terms *old* and *new* information have meant a variety of things over the years (Allerton, 1978; Prince, 1981; Horn, 1986). Since we use Nissim et al.'s (2004) corpus for training and evaluation, the definitions of these concepts we present here are those that Nissim et al. used to annotate their corpus. According to Nissim et al., their definitions are built upon Prince's (1981), and the categorization into *old*, *new*, and *mediated* entities resemble those of Strube (1998) and Eckert and Strube (2001).

Old. As mentioned before, an entity is *old* if it is both known to the hearer and has been mentioned in the conversation. More precisely, an entity is *old* if (1) it is coreferential with an entity introduced earlier, (2) it is a generic pronoun, or (3) it is a personal pronoun referring to the dialogue participants. To exemplify, consider the following sentences.

(1) I was angry that he destroyed **my** tent.

(2) **You** cannot leave until the test is over.

In Example 1, *my* is an *old* entity because it is coreferent with *I*. In Example 2, *You* is an *old* entity because it is a generic pronoun.

Mediated. An entity is *mediated* if it has not been previously introduced in the conversation, but can be inferred from already-mentioned entities or is generally known to the hearer. More specifically, an entity is *mediated* if (1) it is a generally known entity (e.g., *the Earth*, *China*, and most proper names), (2) it is a bound pronoun, or (3) it is an instance of *bridging* (i.e., an entity that is inferrable from a related entity mentioned earlier in the dialogue). As an example, consider the following sentences.

(3a) He passed by the door of Mary's house and saw that **the door** was painted purple.

(3b) He passed by Mary's house and saw that **the door** was painted purple.

In Example 3a, by the time the hearer processes the second occurrence of *the door*, she has already had a mental entity corresponding to *the door* (after processing the first occurrence). As a result, the second occurrence of *the door* is an *old* entity. In Example 3b, on the other hand, the hearer is not assumed to have any mental representation of the door in question, but she can infer that the door she saw was part of Mary's house. Hence, this occurrence of *the door* is a *mediated* entity. In general, an entity that is related to an earlier entity via a part-whole relation or a set-subset relation is *mediated*.

New. An entity is *new* if it has not been introduced in the dialogue and the hearer cannot infer it from previously mentioned entities.

In case more than one class is appropriate for a given entity, Nissim et al. employ additional tie-breaking rules. Suppose, for instance, that we have two occurrences of *China* in a dialogue. The second occurrence can be labeled as *old* (because it is coreferential with an earlier entity) or *mediated* (because it is a generally known entity). According to Nissim et al.'s rules, the entity will be labeled as *old*.

3 Dataset

We employ Nissim et al.'s (2004) dataset, which comprises 147 Switchboard dialogues. A total of 68,992 NPs are annotated with information status: 51.2% of them are labeled as *old*, 34.5% as *mediated* (henceforth *med*), and 14.3% as *new*. Nissim (2006) randomly split the instances created from these NPs into a training set (for classifier training), a development set (for feature development), and an evaluation set (for testing). Hence, the NPs from the same

	Training	Test
old	31358 (51.7%)	3931 (47.4%)
med	20778 (34.2%)	3036 (36.6%)
new	8567 (14.1%)	1322 (16.0%)
total	60703 (100%)	8289 (100%)

Table 1: Information status distribution of NPs.

document may be split across different sets.

Unlike Nissim (2006), we partition the 147 dialogues (rather than the instances) into a training set (117 dialogues) and a test set (30 dialogues). In other words, we do *not* randomize the instances, as we believe that it represents an unrealistic evaluation setting, for the following reasons. First, in practice, the test dialogues may not be available until test time. Second, we may want to examine how a system performs on a given dialogue. Finally, randomizing the instances does not allow us to apply learned knowledge of information status to coreference resolution, which needs to be performed for each dialogue. The information status distribution of the NPs in the training and test sets are shown in Table 1.

4 Baseline System

In this section, we describe our baseline system, which adopts a machine learning approach to determining the information status of a discourse entity.

Building SVM classifiers for information-status determination. We employ the support vector machine (SVM) learner as implemented in the SVM^{light} package (Joachims, 1999) to train three binary classifiers, one for predicting each of the three possible classes (i.e., *new*, *old*, and *med*), using a linear kernel in combination with the *one-versus-all* training scheme.[2] Each training instance represents a single NP and consists of the seven morpho-syntactic features that Nissim (2006) used in her learning-based approach (see Table 2 for an overview). Following Nissim, we extract the NPs directly from the gold-standard annotations, but the features are computed entirely automatically.

[2]SVM was chosen because it provides the option to employ kernels. The reason why we train three binary classifiers rather than just one multi-class classifier (using SVM^{multiclass}) is that SVM^{multiclass} does not permit the use of a non-linear kernel, which we will need to incorporate structured features later on.

Feature	Values
full prev mention	numeric
mention time	{first,second,more}
partial prev mention	{yes,no,NA}
determiner	{bare,def,dem,indef,poss,NA}
NP type	{pronoun,common,proper,other}
NP length	numeric
grammatical role	{subject,subjpass,pp,other}

Table 2: Nissim's feature set.

The seven features are all intuitively useful for determining information status. For instance, if an NP, NP$_k$, and a discourse entity that appears before it have the same string (full prev mention), then NP$_k$ is likely to be an *old* entity. Mention time is the categorical version of full prev mention and therefore serves to detect *old* entities. Partial prev mention is useful for detecting mediated entities, especially those that have a set-subset relation with a preceding entity. For instance, *your dogs* would be considered a partial previous mention of *my dogs* or *my three dogs*. The value "NA" stands for "not applicable", and is used for pronouns. Determiners and NP type are likely to be helpful for all three categories. For instance, indefinite NPs and pronouns are likely to be *new* and *old*, respectively. The "NP length" feature is motivated by the observation that *old* entities tend to contain less lexical materials than *new* entities. For instance, subsequent references to *Barack Obama* may simply be *Obama*.

Applying the classifiers. To determine the information status of an NP in a test dialogue, we create an instance for it as during training and present it independently to the three binary SVM classifiers, each of which returns a real value representing the signed distance of the instance from the hyperplane. We assign the instance to the class that is associated with the most positive classification value.

5 Our Features

We propose to extend Nissim's (2006) feature set with two types of features.

5.1 Lexical Features

As discussed, an entity should be labeled as *med* if it has not been introduced in the dialogue but is generally known to a human. Whether an entity is “generally known” may be easily determined by a human but not by a machine, since world knowledge is involved in the decision process. In particular, Nissim’s feature set does not contain any features that encode the notion of a “generally known” entity.

Hence, it would be desirable to augment Nissim’s feature set with features that indicate whether an entity is generally known or not. One way to do this is to (1) create a list of generally known entities, and then (2) create a binary feature that has the value *True* if and only if the entity under consideration appears in this list. The question, then, is: how can we obtain the list of generally known entities? We may manually assemble this list, but this could be a labor-intensive task. As a result, we propose to *acquire* this kind of world knowledge automatically from annotated data.

Specifically, we augment Nissim’s feature set with the set of *unigrams* that appear in the training data. Given a training/test instance (i.e., discourse entity), we compute the values of its unigram features (henceforth *lexical features*) as follows. For each unigram, we check if it appears in the string representing the discourse entity. If so, its feature value is 1; otherwise, its value is 0. For instance, if the entity is *the red hat*, then all of its lexical features except *the*, *red*, and *hat* will have a value of 0.

It should perhaps not be too difficult to see why these lexical features are useful for the information-status classifier: these features enable the SVM learner to determine the extent to which a unigram correlates with each class. For instance, from the annotated data, the learner will learn that any instance of *China* cannot be labeled as *new*, and the decision of whether it should be an *old* entity or a *med* entity depends on whether it is coreferential with a previously-mentioned entity. Hence, the use of lexical features allows the learner to implicitly acquire some world knowledge.

We believe that lexicalization is an important step towards building high-performance text-processing systems. In fact, lexicalized models have demonstrated their effectiveness in other areas of language processing, such as syntactic and semantic parsing. While lexicalized models may be less portable to new genres and domains than their unlexicalized counterparts, we believe that this issue can be handled via domain adaptation techniques and should not be a reason against lexicalization.

5.2 Structured Features

In Nissim’s (2006) feature set, there are a couple of features that capture NP-internal information, such as determiner, NP length, and NP type. However, there is only one feature that captures the syntactic context of an NP, grammatical role, which is computed based on the parse tree in which the NP resides. This is arguably a very shallow representation of its syntactic context. We hypothesize that we can train more accurate information-status classifiers if we have access to a richer representation of syntactic context. This motivates us to employ syntactic parse trees *directly* as features.

Before describing how this can be done, recall that in a traditional learning setting, the feature set employed by an off-the-shelf learning algorithm typically consists of *flat* features (i.e., features whose values are discrete- or real-valued, as the ones described in the previous section). Advanced machine learning algorithms such as SVMs, on the other hand, have enabled the use of *structured* features (i.e., features whose values are structures such as parse trees), owing to their ability to employ *kernels* to efficiently compute the similarity between two potentially complex structures.

Perhaps the main advantage of employing structured features is *simplicity*. To understand this advantage, consider learning in a setting where we can only employ flat features. If we want to use information from a parse tree as features in this setting, we will need to design heuristics to extract the desired parse-based features from parse trees. For certain tasks, designing a good set of heuristics can be time-consuming and sometimes difficult. On the other hand, SVMs enable a parse tree to be employed directly as a structured feature, obviating the need to design such heuristics.

Given two parse trees (as features), we compute their similarity using a convolution tree kernel (Collins and Duffy, 2001), which efficiently enumerates the number of common substructures in the two trees via dynamic programming. Note, however, that while we want to use a parse tree directly as a feature, we do *not* want to use the *entire* parse tree as a feature. Specifically, while using the entire parse tree enables a richer representation of the syntactic context than using a *partial* parse tree, the increased complexity of the tree also makes it more difficult for the SVM learner to make generalizations.

To strike a better balance between having a rich representation of the context and improving the learner's ability to generalize, we extract a substructure from a parse tree and use it as the value of the structured feature of an instance. Specifically, given an instance corresponding to discourse entity e, we extract the substructure from the parse tree containing e as follows. Let $n(e)$ be the root of the subtree that spans all and only the words in e, and let $Parent(n(e))$ be its immediate parent node. We (1) take the subtree rooted at $Parent(n(e))$, (2) replace each leaf node in this subtree with a node labeled X, (3) replace the subtree rooted at $n(e)$ with a leaf node labeled Y, and (4) use the subtree rooted at $Parent(n(e))$ as the structured feature for the instance corresponding to e. Intuitively, the first three steps aim to provide generalizations by simplifying the tree. For instance, step (1) allows us to focus on using a small window as the context. Steps (2) and (3) help generalization by ignoring the words within e and its context. Note that using two labels, X and Y, enables the kernel to distinguish the discourse entity under consideration from its context within this substructure. In addition, we simply use a single node (Y) to represent the discourse entity, since any NP-internal information has presumably been captured by the flat features. We compute these structured features using hand-annotated parse trees.

While structured features have been employed for a multitude of tasks in syntax, semantics, and information extraction such as syntactic parsing (e.g., Collins (2002)), semantic parsing (e.g., Moschitti (2004)), named entity recognition (e.g., Cumby and Roth (2003), and relation extraction (e.g., Zelenko et al. (2003)), the same is not true for discourse processing tasks. We hope that our use of structured features for information-status classification can promote their use in discourse processing.

5.3 Combining Kernels

Recall that the flat features are computed using a linear kernel, while the structured features are computed using a tree kernel. If we want our learner to make use of more than one of these types of features, we need to employ a *composite* kernel to combine them. Specifically, we define and employ the following composite kernel:

$$K_c(F_1, F_2) = K_1(F_1, F_2) + K_2(F_1, F_2),$$

where F_1 and F_2 are the full set of features that represent the two entities under consideration, and K_1 and K_2 are the kernels we are combining. To ensure that both kernels contribute equally to the composite kernel, we normalize the values they return to the range [0,1].

6 Evaluation

Next, we evaluate the effectiveness of our features in improving information-status classification.

6.1 Results and Discussion

Results of four information-status classification systems are shown in Table 3. Under Original Nissim, we have the results copied verbatim from Nissim's (2006) paper. Baseline is the aforementioned baseline system, which is trained using Nissim's feature set. Baseline+Lexical is the system trained using Nissim's feature set augmented with lexical features. Finally, Baseline+Both is the system trained using Nissim's feature set augmented with both lexical and structured features. For each system, we show the recall (R), precision (P), and F-measure (F) of each of the three classes: *old*, *new*, and *med*. Before we describe the results, two points deserve mention. First, as noted earlier, Nissim partitioned the dialogues into training and test folds in a different way than us. In particular, Original Nissim and the remaining three systems were not evaluated on the same set of test instances. Hence, the Original Nissim results are not directly comparable with those of the other systems. We show them here just to provide another point of reference. Second, the results of the remaining three systems were obtained by aggregating the results of three binary SVM classifiers, as described earlier.

Comparing Baseline and Baseline+Lexical, we see that augmenting Nissim's feature set with lexical features improves the F-measure scores on all three classes. In particular, the F-measure and recall for *med* rise considerably by 3.0 and 7.8, respectively. This provides indirect empirical support for our earlier hypothesis that the *med* class can benefit from

	Original Nissim			Baseline			Baseline+Lexical			Baseline+Both		
	R	P	F	R	P	F	R	P	F	R	P	F
old	91.5	94.1	92.8	91.2	85.8	88.5	88.7	91.7	90.2	93.0	95.2	**94.1**
med	87.6	68.1	76.6	84.7	62.7	72.1	92.5	63.2	75.1	89.1	70.9	**79.0**
new	22.3	56.3	32.0	30.2	66.4	41.5	32.1	68.3	43.7	34.4	71.5	**46.5**
Accuracy	79.5			74.1			76.3			**82.2**		

Table 3: Per-class performance of four information-status classifiers.

the shallow world knowledge that these lexical features help to "extract" from annotated data.

Comparing Baseline+Lexical and Baseline+Both, we see that the addition of structured features enables a further boost to performance: F-measure increases by 2.8–3.9 for the three classes. These results substantiate our hypothesis that employing a richer representation of syntactic context is beneficial to information-status classification.

Comparing Baseline and Baseline+Both, we see that F-measure improves considerably by 5–6.9 for the three classes. Overall, these results provide suggestive evidence that both types of features are effective at improving an information-status classifier that employs Nissim's features.

For further comparison, we show the classification accuracies of the four systems in the last row of Table 3. As we can see, adding lexical features to the baseline features improves accuracy by 2.2%, and adding structured features further improves accuracy by 5.9%. Our two types of features, when used in combination with Nissim's features, improve the baseline substantially by an accuracy of 8.1%.

Note that while our results and Original Nissim's are not directly comparable, the two systems are consistent in terms of the relative performance for the three classes: best for *old* and worst for *new*. The poor performance for *new* is largely a consequence of its low recall, which can in turn be attributed to its lower representation in the dataset. Since many *new* instances are misclassified, a natural question is: are these instances misclassified as *old* or *med*? Similar questions can be raised for *old* and *med*, despite their substantially higher recall values than *new*.

To answer these questions, we need to better understand the kind of errors made by our approach. Consequently, we show in Table 4 the confusion matrix generated from the test set for our best-performing information-status classifier, Baseline+Both. The rows and the columns correspond to the gold tags and the classifier tags, respectively. As we can see, these numbers seem to suggest the "in-between" nature of mediated entities: when an *old* or *new* entity is misclassified, it is typically misclassified as *med* (rows 1 and 3); however, when a *med* entity is misclassified, it is equally likely to be misclassified as *old* and *new* (row 2).

C→ G↓	old	med	new
old	3656	257	18
med	167	2706	163
new	17	850	455

Table 4: Confusion matrix for the Baseline+Both classifier. C=Classifier tag; G=Gold tag

These results are perhaps not surprisingly, since intuitively *med* entities bear some resemblance to both *old* and *new* entities. For instance, the similarity between *med* and *old* stems from the fact that different instances of the same entity (e.g., *China*) can receive one of these two labels, with the decision dependent on whether the entity was previously mentioned in the dialogue. On the other hand, *med* and *new* are similar in that it may sometimes be difficult even for a human to determine whether certain entities should be labeled as *med* or *new*, since the decision depends on whether she believes these entities are *generally known* or not.

6.2 Relation to Anaphoricity Determination

Anaphoricity determination refers to the task of determining whether an NP is anaphoric or not, where an NP is considered anaphoric if it is part of a (non-singleton) coreference chain but is not the head of the chain (Ng and Cardie, 2002). In other words, an

	Anaphoricity			Baseline+Ana			Baseline+Lexical+Ana			Baseline+Both+Ana		
	R	P	F	R	P	F	R	P	F	R	P	F
old	91.4	86.6	88.9	91.3	87.3	89.3	90.8	91.7	91.3	92.8	94.9	93.9
med	84.3	63.1	72.2	84.9	64.1	73.1	92.3	64.7	76.1	88.7	71.1	78.9
new	30.8	66.4	42.1	31.1	66.9	42.5	32.9	68.7	44.5	34.1	71.7	46.2
Accuracy	74.7			75.1			77.6			82.0		

Table 5: Impact of knowledge of anaphoricity on the information-status classifiers.

NP is anaphoric if and only if it has an antecedent.

Given this definition, anaphoricity determination bears resemblance to information-status classification. For instance, an *old* entity is anaphoric, since it has been introduced earlier in the conversation and therefore have an antecedent. Similarly, a *new* or *med* entity is non-anaphoric, since the entity has not been previously introduced in the conversation and therefore cannot have an antecedent.

There has been a lot of recent work on anaphoricity determination (e.g., Bean and Riloff (1999), Uryupina (2003), Ng (2004), Denis and Baldridge (2007), Versley et al. (2008), Ng (2009), Zhou and Kong (2009)). Given the similarity between this task and information-status classification, a natural question is: will the anaphoricity features previously developed by coreference researchers be helpful for information-status classification? To answer this question, we (1) assemble a feature set composed of the 26 anaphoricity features previously used by Rahman and Ng (2009),[3] and then (2) repeat the experiments in Table 3, except that we augment the feature set used in each of these experiments with the anaphoricity features we assembled in step (1).

Results with the anaphoricity features are shown in Table 5. Under Anaphoricity, we have the results obtained using only the 29 anaphoricity features. As we can see, these results are comparable to those obtained using the Baseline features. Comparing each of Baseline+Ana and Baseline+Lexical+Ana with the corresponding experiments in Table 3, we see that the addition of anaphoricity features yields a mild performance improvement, which is consistent over all three classes. However, comparing the last column of the two tables, we can see that in the presence of the structured features, the anaphoricity features do not contribute positively to overall performance. Hence, in the coreference experiments in the next section, we will not employ anaphoricity features for information-status classification.

[3]These 26 features are derived from those employed by Ng and Cardie's (2002) anaphoricity determination system. See Footnote 2 of Rahman and Ng (2009) for details.

7 Application to Coreference Resolution

Since the significance of information-status classification stems in part from the potential benefits it brings to higher-level NLP applications, we determine whether our information-status classification systems can offer benefits to learning-based coreference resolution. Since the 147 information-status annotated dialogues are also coreference annotated, we use them in our coreference evaluation. To our knowledge, our work represents the first attempt to report coreference results on this dataset.

7.1 Coreference Models

While the so-called mention-pair coreference model has dominated coreference research for more than a decade since its appearance in the mid-1990s, a number of new coreference models have been proposed in recent years. To investigate whether these newer, presumably more sophisticated, coreference models can better exploit the automatically acquired information-status information, we will evaluate the usefulness of information-status information when used in combination with two different coreference models, the aforementioned mention-pair model and the recently-developed cluster-ranking model.

7.1.1 Mention-Pair Model

The mention-pair (MP) model, proposed by Aone and Bennett (1995) and McCarthy and Lehnert (1995), is a classifier that determines whether two NPs are co-referring or not. Each instance $\boldsymbol{i}(\text{NP}_j, \text{NP}_k)$ corresponds to two NPs, NP_j and NP_k, and is represented by 39 features. Table 1 of Rahman and

Ng (2009) contains a detailed description of these features. Linguistically, they can be divided into four categories: string-matching, grammatical, semantic, and positional. They can also be categorized based on whether they are relational or not. Specifically, relational features capture the relationship between NP_j and NP_k, whereas non-relational features capture the linguistic property of one of these NPs.

We follow Soon et al.'s (2001) method for creating training instances. Specifically, we create (1) a positive instance for each anaphoric NP NP_k and its closest antecedent NP_j; and (2) a negative instance for NP_k paired with each of the intervening NPs, NP_{j+1}, NP_{j+2}, $\ldots$, NP_{k-1}. The classification associated with a training instance is either positive or negative, depending on whether the two NPs are coreferent. To train the MP model, we use the SVM learner from SVM^{light} (Joachims, 1999).[4]

After training, the classifier is used to identify an antecedent for an NP in a test text. Specifically, each NP, NP_k, is compared in turn to each preceding NP, NP_j, from right to left, and selects NP_j as its antecedent if the pair is classified as coreferent. The process terminates as soon as an antecedent is found for NP_k or the beginning of the text is reached.

Despite its popularity, the MP model has two major weaknesses. First, since each candidate antecedent for an NP to be resolved (henceforth an *active NP*) is considered independently of the others, this model only determines how good a candidate antecedent is relative to the active NP, but not how good a candidate antecedent is relative to other candidates. So, it fails to answer the critical question of which candidate antecedent is most probable. Second, it has limitations in its expressiveness: the information extracted from the two NPs alone may not be sufficient for making a coreference decision.

7.1.2 Cluster-Ranking Model

The cluster-ranking (CR) model, proposed by Rahman and Ng (2009), addresses the two weaknesses of the MP model by combining the strengths of the *entity-mention* model (e.g., Luo et al. (2004), Yang et al. (2008)) and the *mention-ranking* model (e.g., Denis and Baldridge (2008)). Specifically, the CR model ranks the preceding clusters for an active NP so that the highest-ranked cluster is the one to which the active NP should be linked. Employing a ranker addresses the first weakness, as a ranker allows all candidates to be compared *simultaneously*. Considering preceding clusters rather than antecedents as candidates addresses the second weakness, as *cluster-level* features (i.e., features that are defined over any subset of NPs in a preceding cluster) can be employed.

Since the CR model ranks preceding clusters, a training instance $\boldsymbol{i}(c_j, \text{NP}_k)$ represents a preceding cluster c_j and an anaphoric NP NP_k. Each instance consists of features that are computed based solely on NP_k as well as cluster-level features, which describe the relationship between c_j and NP_k. Motivated in part by Culotta et al. (2007), we create cluster-level features from the *relational* features in our feature set using four predicates: NONE, MOST-FALSE, MOST-TRUE, and ALL. Specifically, for each relational feature X, we first convert X into an equivalent set of binary-valued features if it is multi-valued. Then, for each resulting binary-valued feature X_b, we create four binary-valued cluster-level features: (1) NONE-X_b is true when X_b is false between NP_k and each NP in c_j; (2) MOST-FALSE-X_b is true when X_b is true between NP_k and less than half (but at least one) of the NPs in c_j; (3) MOST-TRUE-X_b is true when X_b is true between NP_k and at least half (but not all) of the NPs in c_j; and (4) ALL-X_b is true when X_b is true between NP_k and each NP in c_j.

We train a cluster ranker to jointly learn anaphoricity determination and coreference resolution using SVM^{light}'s ranker-learning algorithm. Specifically, for each NP, NP_k, we create a training instance between NP_k and *each* preceding cluster c_j using the features described above. Since we are learning a joint model, we need to provide the ranker with the option to start a new cluster by creating an additional training instance that contains features that solely describes NP_k. The rank value of a training instance $\boldsymbol{i}(c_j, \text{NP}_k)$ created for NP_k is the rank of c_j among the competing clusters. If NP_k is anaphoric, its rank is HIGH if NP_k belongs to c_j, and LOW otherwise. If NP_k is non-anaphoric, its rank is LOW unless it is the additional training instance described above, which has rank HIGH.

After training, the cluster ranker processes the NPs in a test text in a left-to-right manner. For each

[4]For this and subsequent uses of the SVM learner in our experiments, we set all parameters to their default values.

active NP, NP_k, we create test instances for it by pairing it with each of its preceding clusters. To allow for the possibility that NP_k is non-anaphoric, we create an additional test instance that contains features that solely describe the active NP (similar to what we did in the training step above). All these test instances are then presented to the ranker. If the additional test instance is assigned the highest rank value by the ranker, then NP_k is classified as non-anaphoric and will not be resolved. Otherwise, NP_k is linked to the cluster that has the highest rank.

7.2 Coreference Experiments

7.2.1 Experimental Setup

The training/test split we use in the coreference experiments is the same as that in the information-status experiments. Specifically, we use the training set to train both the information-status classifier and our coreference models, apply the information-status classifier to each discourse entity in the test set, and have the coreference models resolve all and only those NPs that are labeled as *old* by the information-status classifier. Our decision to allow the coreference models to resolve only the *old* entities is motivated by the fact that *med* and *new* entities have *not* been previously introduced in the conversation and therefore do not have antecedents. The NPs used by the coreference models are the same as those accessible to the information-status classifier.

We employ two scoring programs, B^3 (Bagga and Baldwin, 1998) and ϕ_3-CEAF (Luo, 2005), to score the output of a coreference model. Given a gold-standard (i.e., key) partition, KP, and a system-generated (i.e., response) partition, RP, B^3 computes the recall and precision of each NP and averages these values at the end. Specifically, for each NP, NP_j, B^3 first computes the number of NPs that appear in both KP_j and RP_j, the clusters containing NP_j in KP and RP, respectively, and then divides this number by $|KP_j|$ and $|RP_j|$ to obtain the recall and precision of NP_j, respectively. On the other hand, CEAF finds the best one-to-one alignment between the key clusters and the response clusters using the Kuhn-Munkres algorithm (Kuhn, 1955), where the weight of an edge connecting two clusters is equal to the number of NPs that appear in both clusters. Precision and recall are equal to the sum of the weights of the edges in the alignment divided by the total number of NPs in the response and the key, respectively.

7.2.2 Results and Discussion

As our baseline, we employ our coreference models to generate NP partitions on the test documents *without* using any knowledge of information status. Results, reported in terms of recall (R), precision (P), and F-measure (F) using B^3 and ϕ_3-CEAF, are shown in row 1 of Table 6.[5] As we can see, the baseline achieves B^3 F-measures of 69.2 (MP) and 74.5 (CR) and CEAF F-measures of 61.6 (MP) and 68.5 (CR). These results suggest that the CR model is stronger than the MP model, corroborating previous empirical findings (Rahman and Ng, 2009).

Next, we examine the impact of learned knowledge of information status on the performance of a coreference model. Since knowledge of information status enables a coreference model to focus on resolving only the *old* entities, we hypothesize that the resulting model will have a higher precision than one that does not employ such knowledge. An equally important question is: will the F-measure of the resulting model improve? Since we are employing knowledge of information status in a *pipeline* coreference architecture where information-status classification is performed prior to coreference resolution, errors made by the (upstream) information-status classifier may propagate to the (downstream) coreference system. Given this observation, we hypothesize that the answer to the aforementioned question depends in part on the accuracy of information-status classification. In particular, the higher the accuracy of information-status classification is, the more likely the F-measure of the downstream coreference model will improve. To test this hypothesis, we conduct experiments where we employ the knowledge provided by the three information-status classifiers which, as discussed earlier, perform at varying levels of accuracy — the first one using only Nissim's features, the second one using both lexical and Nissim's features, and the last one using Nissim's features in combination with lexical and parse-based features — for our coreference models.

[5]Since gold-standard NPs are used in our experiments, CEAF precision is always equal to CEAF recall. For brevity, we only report F-measure scores for CEAF in the table.

	Mention-Pair Model				Cluster-Ranking Model			
	B^3			CEAF	B^3			CEAF
System	R	P	F	F	R	P	F	F
No knowledge of information status	78.6	61.8	69.2	61.6	78.2	71.1	74.5	68.5
Nissim features only	73.4	67.3	70.2	62.1	73.6	77.4	75.4	69.7
Nissim+Lexical features	71.0	69.5	70.2	61.9	73.7	77.3	75.4	69.9
Nissim+Lexical+Parse features	74.1	66.8	70.3	62.3	77.3	74.0	75.6	71.1
Perfect information status	76.7	68.1	72.1	66.4	77.1	79.5	78.3	74.2

Table 6: B^3 and CEAF coreference results.

Results of the coreference models employing knowledge provided by the three information-status classifiers are shown in rows 2–4 of Table 6. As expected, B^3 precision increases in comparison to the baseline, regardless of the coreference model and the scoring program. In addition, employing knowledge of information status always improves coreference performance: F-measure scores increase by 1.0–1.1% (B^3) and 0.3–0.7% (CEAF) for the MP model, and by 0.9–1.1% (B^3) and 1.2–2.6% (CEAF) for the CR model. These results suggest that the three information-status classifiers have achieved the level of accuracy needed for the coreference models to improve. On the other hand, it is somewhat surprising that the three information-status classifiers have yielded coreference systems that perform at essentially the same level of performance.

To understand why better information-status classification results do not necessarily yield better coreference performance, we take a closer look at the results of the coreference resolver employing Nissim's features (henceforth NISSIM) and the resolver employing our Nissim+Lexical+Parse features (henceforth FULL-FEATURE). Among the *old* entities that were correctly classified using our features and incorrectly classified by Nissim's features, we found that the precision of the FULL-FEATURE system suffered (since in many cases the coreference models identified wrong antecedents for these *old* entities) whereas the NISSIM system remained unaffected (since the entities were misclassified and would not be resolved by the models). In addition, although many *med* and *new* entities were correctly classified using our features and incorrectly classified (as *old*) using Nissim's features, we found that in many cases no antecedents were identified for these misclassified entities and hence the precision of the NISSIM system was not adversely affected.

Finally, we investigate whether our coreference system could be improved if it had access to perfect knowledge of information status (taken directly from the gold-standard annotations). This experiment will allow us to determine whether the usefulness of knowledge of information status for coreference resolution is limited by the accuracy in computing such knowledge. Results are shown in the last row of Table 6. As we can see, using perfect information-status knowledge yields a coreference system that improves those that employs automatically acquired information-status knowledge by 1.8–4.1% (MP) and 2.7–3.1% (CR) in F-measure. This indicates that the accuracy in computing such knowledge does play a role in determining its usefulness for coreference resolution.

8 Conclusions

We examined the problem of automatically determining the information status of discourse entities in spoken dialogues. In particular, we augmented Nissim's feature set with two types of features: lexical features, which capture in a shallow manner world knowledge implicitly encoded in the annotated data; and syntactic parse trees, which provide a richer representation of the syntactic context in which a discourse entity appears than grammatical roles. Results on 147 Switchboard dialogues demonstrated the effectiveness of these features: we obtained a significant improvement of 8.1% in accuracy over a information-status classifier trained on Nissim's feature set. In addition, we evaluated information-status classification in the context of coreference resolution, and showed that automatically acquired knowledge of information status can be profitably used to improve coreference systems.

Acknowledgments

We thank the three reviewers for their invaluable comments on an earlier draft of the paper. This work was supported in part by NSF Grant IIS-0812261.

References

David J. Allerton. 1978. The notion of 'givenness' and its relations to presupposition and to theme. *Lingua*, 44:133–168.

Chinatsu Aone and Scott William Bennett. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 122–129.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at the First International Conference on Language Resources and Evaluation*, pages 563–566.

David Bean and Ellen Riloff. 1999. Corpus-based identification of non-anaphoric noun phrases. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 373–380.

Sasha Calhoun, Jean Carletta, Jason Brenier, Neil Mayo, Dan Jurafsky, Mark Steedman, and David Beaver. 2010. The NXT-format Switchboard corpus: A rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. *Language Resources and Evaluation*, 44(4):387–419.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pages 625–632.

Michael Collins. 2002. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496.

Aron Culotta, Michael Wick, and Andrew McCallum. 2007. First-order probabilistic models for coreference resolution. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 81–88.

Chad Cumby and Dan Roth. 2003. On kernel methods for relational learning. In *Proceedings of the 20th International Conference on Machine Learning*, pages 107–114.

Pascal Denis and Jason Baldridge. 2007. Global, joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 236–243.

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.

Miriam Eckert and Michael Strube. 2001. Dialogue acts, synchronising units and anaphora resolution. *Journal of Semantics*, 17(1):51–89.

Eva Hajičová. 1984. Topic and focus. In *Contributions to Functional Syntax, Semantics, and Language Comprehension (LLSEE 16)*, pages 189–202. John Benjamins, Amsterdam.

Michael A. K. Halliday. 1976. Notes on transitivity and theme in English. *Journal of Linguistics*, 3(2):199–244.

Laurence R. Horn. 1986. Presupposition, theme, and variations. In A. Farley et al., editor, *Papers from the Parasession on Pragmatics and Grammatical Theory at the 22nd Regional Meeting*, pages 168–192. CLS.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In Bernhard Scholkopf and Alexander Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 44–56. MIT Press.

Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(83).

Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the Bell tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 135–142.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32.

Joseph McCarthy and Wendy Lehnert. 1995. Using decision trees for coreference resolution. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pages 1050–1055.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 335–342.

Vincent Ng and Claire Cardie. 2002. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 730–736.

Vincent Ng. 2004. Learning noun phrase anaphoricity to improve conference resolution: Issues in representation and optimization. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 151–158.

Vincent Ng. 2009. Graph-cut-based anaphoricity determination for coreference resolution. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 575–583.

Malvina Nissim, Shipra Dingare, Jean Carletta, and Mark Steedman. 2004. An annotation scheme for information status in dialogue. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*.

Malvina Nissim. 2006. Learning information status of discourse entities. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 94–102.

Ellen Prince. 1981. Toward a taxonomy of given-new information. In P. Cole, editor, *Radical Pragmatics*, pages 223–255. New York, N.Y.: Academic Press.

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.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Mark Steedman. 2000. *The Syntactic Process*. The MIT Press, Cambridge, MA.

Michael Strube. 1998. Never look back: An alternative to centering. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, pages 1251–1257.

Olga Uryupina. 2003. High-precision identification of discourse new and unique noun phrases. In *Proceedings of the ACL Student Research Workshop*, pages 80–86.

Enric Vallduví. 1992. *The Informational Component*. Garland, New York.

Yannick Versley, Alessandro Moschitti, Massimo Poesio, and Xiaofeng Yang. 2008a. Coreference systems based on kernels methods. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 961–968.

Renata Vieira and Massimo Poesio. 2000. An empirically-based system for processing definite descriptions. *Computational Linguistics*, 26(4):539–593.

Xiaofeng Yang, Jian Su, Jun Lang, Chew Lim Tan, and Sheng Li. 2008. An entity-mention model for coreference resolution with inductive logic programming. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 843–851.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

GuoDong Zhou and Fang Kong. 2009. Global learning of noun phrase anaphoricity in coreference resolution via label propagation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 978–986.