Machine Learning for Constituency Test of Coordinating Conjunctions in Requirements Specifications

Richa Sharma
School of IT
Indian Institute of Technology, Delhi
India
sricha@gmail.com

Jaspreet Bhatia
School of IT
Indian Institute of Technology, Delhi
India
jaspreet2709@gmail.com

K.K. Biswas
Dept. of Computer Science and Engg.
Indian Institute of Technology, Delhi
India
kkb@cse.iitd.ac.in

ABSTRACT
Coordinating conjunctions have been a major source of ambiguity in Natural Language statements and the concern has been a major research focus in English Linguistics. Natural Language is also the most common form of expressing the requirements for an envisioned software system. These requirement documents also suffer from similar concern of coordination ambiguity. Presence of nocuous coordination ambiguity is a major concern for the requirements analysts. In this paper, we explore the applicability of constituency test for identifying coordinating conjunction instances in the requirements documents. We show through our study how identification of nocuous and innocuous coordinating conjunctions can be improved using semantic similarity heuristics and machine learning. Our study indicates that Naïve Bayes classifier outperforms other machine learning algorithms.

Categories and Subject Descriptors
D.2.1 [**Software Engineering**]: Requirements/Specifications – *tools;* I.2.7 [**Artificial Intelligence**]: Natural Language processing – *text analysis.*

General Terms
Languages, Documentation, Experimentation

Keywords
Coordination Ambiguity, Conjunctions, Constituency Test, Requirements Specifications, Machine Learning.

1. INTRODUCTION
Natural Language (NL) is the most common and the most preferred form of expressing the requirements for an envisioned software system. The fact that NL is understood by all the stakeholders involved in software development and is quite expressive makes NL a suitable choice for documenting the requirements specifications. However, NL is inherently ambiguous in nature. NL statements often lead to different interpretations by different readers. Ambiguity concern is more intensified in context of software development [1]. Requirements for the software correspond to a certain business domain and the requirements specifications make extensive use of domain-relevant terms. The software development team may not be familiar and conversant with the business domain relevant terms. Such ambiguities are often either overlooked or lead to more misunderstandings while documenting and reviewing the requirements specifications. These specifications are large in size, thereby making the ambiguity problem in requirements more intractable. We are motivated by this industry problem of ambiguities in requirements specifications. We are interested in training the machine learning classifier for automatic identification of ambiguous requirements. Our focus in this paper is to address the concern of coordination ambiguity arising due to the use of '*and*' and '*or*'. Coordination ambiguity arises due to the complex syntactic structure linking together two or more elements through coordinating conjunctions like and, or, but, for, yet etc. Coordination ambiguity is known to be a pernicious source of structural ambiguity in English [2]. Though not all statements making use of coordinating conjunctions are interpreted differently by different readers, yet a small portion of coordinating statements can potentially be interpreted differently and can be thought of as instances of *nocuous ambiguity* [3]. Coordination ambiguity concern has been researched extensively in English linguistics by several authors like [4], [5], [6], [7] etc. The authors have suggested algorithms and heuristics to identify coordination ambiguities. Corpus-based distributional heuristics have been used to detect coordination ambiguities for requirements specifications too [8], [9].

In this paper, we present an approach for identifying and learning the instances of nocuous coordinating ambiguity from requirements corpus. We have isolated nocuous coordination ambiguity instances from innocuous instances by making use of *constituency test* for coordinating conjunctions. A constituent, in syntactic analysis, refers to a group of words that function as a single unit [10]. In our study, we have used word-based similarity heuristics to perform the constituency test. The labeled instances of nocuous and innocuous coordination ambiguities have been used to learn the nocuous ones using (ML) classification algorithms. The count of coordinating statements is relatively low as compared to the size of requirements corpus. Our evaluation study is, therefore, based on semi-supervised learning [11]. We have explored following questions in our study:

1. Which ML algorithm is best suited for identifying nocuous coordination ambiguities?
2. Which heuristics serve best to identify nocuous coordination ambiguities?

RAISE'14, June 3, 2014, Hyderabad, India
Copyright 2014 ACM 978-1-4503-2846-3/14/06...$15.00
http://dx.doi.org/10.1145/2593801.2593806

3. Are distributional heuristics for identifying nocuous coordination ambiguity instances helpful in context of requirements specifications?

The rest of the paper is organized as follows: section 2 presents an overview of coordination ambiguity; current state-of-art in English linguistics and its application to NL requirements specifications. Section 3 discusses our approach including the heuristics used and an overview of the ML algorithms used. In section 4, we present our evaluation study along with results and observations. Section 5 finally presents the conclusion.

2. COORDINATION AMBIGUITY

2.1 Background

Ambiguity is a pervasive problem in NL. Coordinating structures are widely acknowledged for syntactic ambiguity in NL statements. Identifying the scope of coordinating structures and associated modifying as well as modified attachments is a challenge that has been addressed in various ways like lexical similarity and structural parallelism [12], syntactic and lexical cues [4], [6], distributional heuristics [8]. Semantic similarity based heuristics for detecting coordination ambiguity has been explored in [9]. We first present a brief overview of these approaches towards resolving coordination ambiguity in English Linguistics in the following sub-section followed by adoption of some of these approaches for NL requirements.

2.2 Current State-of-art in English Linguistics

Coordinating structures can occur between two or more nouns (or phrases) (NP), verbs (or phrases) (VP), two adjectives (JJ) and two sentences. Study of coordinating structures has shown that these structures can be ambiguous due to the scope of attachment of the coordinating terms or units with other structural units present in a NL statement. For example:

(1) This role is a combination of three roles – Analyst, ***Regional Manager and Reviewer****.*

In the above requirements statement, there can be following two interpretations for coordinating terms – *Manager* and *Reviewer*:

1. (Regional (Manager and Reviewer))
2. ((Regional Manager) and Reviewer)

Therefore, the above statement can be referred to as an instance of nocuous ambiguity as it can be interpreted differently by different readers. However, coordination between sentences referred to as S-conjunction is not ambiguous for it connects two independent sentences. For example:

(2) If ***the Call Period has passed*** *and* ***there are no subsequent Call Periods****, then Call Option field will default to 'No'*

In this statement, coordinating conjunction, '*and'* connects two independent statements:

S1: ***the Call Period has passed***

S2: ***there are no subsequent Call Periods***

Therefore, statements of above type are not ambiguous. Research efforts for disambiguating coordinating structures have focused on the type (1) statements. Identifying potentially ambiguous coordinating structures has been of interest to English linguists. Wu et al. [4] have suggested linguistic and syntactic rules to disambiguate the coordinating structures in a statement. They have devised algorithms to automate the suggested rules. They have been able to get an accuracy of 85.3% while disambiguating (NP1 and NP2 + NP3) coordinating structure and, an accuracy of 87.7% with (adjective + NP1 and NP2) structure. Goldberg [5] has shown the use of unsupervised statistical model for detecting ambiguous coordination with noun phrases of the form (NP1 preposition NP2 and NP3). Her model performs with an accuracy of 72% for un-annotated 1988 Wall Street Journal text. Resnik [2] has explored syntactic structure of the form (NP1 and NP2 + N3). Resnik has considered semantic similarity between the coordinating terms in order to resolve the coordination ambiguity. The approach has been shown to work with an accuracy of 72%. Recently, Brouwer et al. [7] have explored the role of syntactic and lexical probabilities specified in Probabilistic Context Free Grammar to account for NP coordination preference. Banik [8] has analyzed VP coordination and quantifier scope in Lexicalized Tree Adjoining Grammar for simple sentences.

2.3 Related work for NL Requirements

The concern of coordination ambiguity in NL requirements has been explored by Chantree et al. [3], [8] and Yang et al. [9]. Chantree et al. have investigated the role of distributional similarity between the head words involved in coordination by comparing the rankings provided by the Sketch Engine[1] [13], which obtains the information about distribution of the headwords from British National Corpus[2] (BNC). They have suggested Distributional Similarity, Collocation Frequency and other heuristics – all based on the rankings provided by Sketch Engine, which draws distribution information from BNC. The distributional similarity and coordination match heuristics have provided an accuracy of 75% for innocuous ambiguities in their study. Yang et al. have studied the role of local collocation frequency and the semantic similarity measure suggested by Resnik [2] in addition to the heuristics suggested by Chantree et al. in [3]. Yang et al. have worked on identifying nocuous ambiguity using LogitBoost algorithm. Their study has shown a precision of 75%.

3. OUR APPROACH

Ambiguity concern, as suggested in [1], becomes more intensified in requirements documents owing to the presence of domain-related context as well as the use of technical words that slightly differ from routine conversational language. The syntactic and lexical cues as suggested in earlier works on coordination ambiguity also do not hold much relevance for requirements specifications. We encountered various instances of coordinating terms where distributional similarity heuristics as well as lexical and syntactic cues were of no help. Instead, semantic similarity owing to either domain-relevance or technical-relevance hinted towards coordinating constituent. Let us consider one such interesting example from our dataset:

(3) retrieve and edit

Here, the coordinating units or terms – '*retrieve*' and '*edit*' are not found to be related according to BNC distribution frequency. However, both of these coordinating terms are frequently used in requirements specifications to specify how a particular entity details should be updated. These are reported semantically similar

[1] http://www.sketchengine.co.uk

[2] http://natcorp.ox.ac.uk

by the semantic similarity heuristics. Therefore, we have explored semantic similarity based heuristics in our study.

Our approach is based on the considering coordination ambiguity as a semantic property of NL statements. In [14], the author discusses nature of ambiguity ranging from lexical to syntactic and semantic challenges. The author argues that an expression is ambiguous if it is associated with more than one region of the meaning space. Applying the same theory to coordinating structures, we can safely assume that a coordinating expression will not lead to ambiguity if the coordinating units, i.e. the words or phrases to the left and right of '*and*' and '*or*' convey more or less the same meaning. Consequently, these units conjoined by '*and*' and '*or*' act as a constituent. In our approach, we have made use of semantic similarity measures for coordinating terms to identify coordinating constituents. We argue that higher the score of similarity between coordinating units, the more are the chances of these units of acting as a 'constituent'; and, when the coordinating units act as a constituent, the scope of the coordinating terms becomes less ambiguous within the sentence. Let us consider the corresponding statement to the example (3) to understand this point:

The GUI should allow administrator to 'retrieve and edit' existing painting information.

All the subjects involved in the study marked the above statement as unambiguous. The viewpoint of the subjects states:

(i) *administrator* only retrieves and edits, i.e. *'retrieve and edit'* act as constituent and, therefore the scope of both *retrieve* and *edit* extends to the actor *administrator*.

(ii) Similarly, the scope of actions: '*retrieve*' and '*edit*' extends to the object: '*existing painting information'*.

Our study is focused on finding the suitable similarity heuristics that prove useful in identifying coordinating constituents. We have considered coordinating structures of the form: (NP1 and/or NP2), (VP1 and/or VP2) and (JJ1 and/or JJ2) in our study. We present below a brief overview of the semantic similarity heuristics that we have used.

3.1 Semantic Similarity Heuristics

3.1.1 Path Length (path)

Path Length heuristic is a simple node-counting scheme in hierarchical dictionary structure. We have used the WordNet implementation of this heuristic as available in Wordnet::Similarity package [17].

The similarity score is inversely proportional to the number of nodes along the shortest path between the concepts. The shortest possible path occurs when the two concepts are the same and, the corresponding path length is 1 in that case. Consequently, the maximum similarity value is 1.

3.1.2 jcn

The relatedness measure, suggested by Jiang and Conrath [18], is defined as:

jcn = 1 / jcn_distance,

where, jcn_distance = IC(concept1) + IC(concept2) - 2 * IC(LCS)

Here, IC refers to the information content of the coordinating terms and LCS refers to the Least Common Subsumer of the coordinating terms.

3.1.3 lin

The similarity heuristic, as proposed by Lin [19], defines the score of relatedness as:

lin = 2 * IC(LCS) / (IC(concept1) + IC(concept2)),

where, IC is the information content of the coordinating terms and LCS is the Least Common Subsumer. The relatedness value provided by this heuristic is always greater-than or equal-to zero and less-than or equal-to one.

3.1.4 res

For Resnik measure [2], the relatedness value is equal to the information content (IC) of the Least Common Subsumer (LCS) of the two coordinating units, i.e. the most informative subsumer, i.e:

res = IC (LCS)

3.1.5 wup

The heuristic, proposed by Wu and Palmer [20], measures the relatedness score of two terms by considering the depths of the two concepts (s1 and s2) in the WordNet taxonomies, along with the depth of the LCS.

wup = 2*depth(LCS) / (depth(s1) + depth(s2)).

3.1.6 lch

The relatedness measure, as proposed by Leacock and Chodrow [21], is defined as:

lch = -log (length / (2 * D)),

where, length refers to the shortest path between the two concepts obtained by using node-counting and, D refers the maximum depth of the taxonomy in which the concepts are found.

3.1.7 lesk

The lesk measure works by finding overlaps between the definitions of the two concepts as provided by the WordNet dictionary. The relatedness score is the sum of the squares of the overlap lengths. It is based on an algorithm proposed in [22] for word sense disambiguation. For example, a single word overlap results in a score of 1. Two single-words overlap results in a score of 2. A two word overlap (i.e., two consecutive words) results in a score of 4. A three word overlap results in a score of 9.

3.1.8 hso

The hso heuristic, proposed by Hirst and St-Onge [23], is based on finding the lexical chains linking the two word senses.

3.1.9 Gloss Vector (vec)

The Gloss Vector [24] measure works by forming second-order co-occurrence vectors from the glosses or WordNet definitions of concepts. The relatedness of the two concepts is determined as the cosine of the angle between their gloss vectors. In order to get around the data sparsity issues presented by extremely short glosses, this measure augments the glosses of concepts with glosses of adjacent concepts as defined by WordNet relations head.

3.1.10 Gloss Vector – pairwise (vpair)

The pair-wise Gloss Vector measure is quite similar to the regular Gloss Vector measure, except that it augments the glosses of concepts with adjacent glosses. The regular Gloss Vector measure first combines the adjacent glosses to form one large 'super-gloss' and creates a single vector corresponding to each of the two concepts from the two 'super-glosses'. The pair-wise Gloss Vector measure, on the other hand, forms separate vectors corresponding to each of the adjacent glosses, i.e. it does not form a single super gloss.

3.1.11 Distributional Similarity (distr)

Distributional Similarity refers to the similarity score of the coordinating units based on the frequency with which the two units appear together in a corpus. This heuristic is same as that has been used in [3] and we have calculated it in the similar manner for our dataset of coordinating structures. To compute the value of this heuristic, we search for one of the root words of the coordinating terms in the BNC using the Sketch Engine's Word Sketch facility [13] and note the rankings of the other head word in the list of matches that is returned. We perform similar procedure of rank-recording with the root word of the other coordinating term and use higher of the two rankings as our heuristic value.

3.2 Semi-supervised Learning

We are interested in automatic classification of the requirements statements into potentially ambiguous (nocuous) and unambiguous (innocuous) categories. The ML classification algorithms can provide the foundation for the purpose of our study. ML algorithms can be classified into two broad categories: supervised and unsupervised learning. Supervised algorithms make use of the guiding function that maps inputs to desired outputs (also referred to as labels, because these are often provided by human experts labeling the training set). On the other hand, unsupervised learning models a set of inputs by grouping or clustering common instances/patterns.

We have used semi-supervised learning (SSL), as suggested for smaller training sets in [11], for our study. Using labels for supervised study proves useful in the presence of large amount of labeled data. But, statements with potentially ambiguous coordinating structures are only a small fraction of large-sized documents having few thousand statements. SSL is considered halfway between supervised and unsupervised learning. Of the various approaches for SSL described in [16], one of the approaches that we found applicable to our study considers SSL as supervised learning with additional information on distribution of the training-set. In our evaluation study, we have worked with a confidence function (for labels) to decide nocuous and innocuous labels for our dataset of ambiguous requirements statements. The input to the classifier is a feature vector corresponding to the coordinating structure present in a statement. This feature vector is composed of the computed values for the heuristics presented in section 3.1. The values for similarity heuristics have been obtained using the scripts available from the link to similarity project of University of Minnesota.[3]. In addition to the confidence function, we have obtained additional information for suitable heuristics to be used for the study by making use of attribute evaluator. The feature or attribute evaluator assists in finding the best heuristic contributing more towards training the classifier for classifying nocuous and innocuous instances of ambiguity. We have also performed a validation check for these more suited features or attributes for classification. We have carried out unsupervised clustering with features identified using evaluator to verify that actually two (or, at the most three) clusters corresponding to nocuous and innocuous instances of ambiguity are obtained. If we get two or at the most three clusters, then it indicates that features or attributes (heuristics in our case) identified by the evaluator provide results closer to the labeled data.

We have made use of Naïve Bayes, K-nearest neighbors (K-NN), Random Tree and Random Forest algorithms in our study. Naïve Bayes classifier is a probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumption. It assumes that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. Despite this assumption, Naive Bayes classifiers can be trained very efficiently in a supervised learning setting. K-NN is an instance-based classifier that learns by relating the unlabeled to the labeled training set according to some distance or similarity function. Random tree algorithm works by constructing multiple decision trees randomly, where each node of the tree records class distributions. Random forest algorithm is an ensemble learning method for classification (and regression) that operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes output by individual trees.

3.3 Evaluation Metrics

For evaluation of our approach, we use three metrics: Precision, Recall and F-Measure. Precision is the fraction of predicted ambiguous statements that are relevant, while recall is the fraction of ambiguous statements that are retrieved. F-measure is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score. F-Measure score can be interpreted as a weighted average of the precision and recall, where it reaches its best value at 1 and worst score at 0.

Precision= True Positive / (True Positive + False Positive)

Recall = True Positive / (True Positive + False Negative)

F-Measure = 2 * (Precision * Recall) (Precision + Recall)

We have used cross validation technique with 10 folds as recommended in [15] for our evaluation study. Cross-Validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. We have used Weka[4] tool for our study.

Of these three metrics, recall plays an important role in our study as our goal is to identify the nocuous ambiguities from the requirements specifications. Once automatically identified, these ambiguous statements can be presented to the domain experts for necessary corrections.

[3] http://maraca.d.umn.edu/cgi-bin/similarity/similarity.cgi

[4] http://www.cs.waikato.ac.nz/ml/weka/

4. EVALUATION STUDY

We carried out our evaluation study on requirements corpus from various domains like medical, academics, human-resource and finance. We extracted the statements containing '*and*' and '*or*' from this corpus. Though the count of such statements is proportionately less (nearly 20.8%), nevertheless, the ambiguity concern posed by them needs to be addressed. We present below the details of our dataset used for evaluation study.

4.1 Dataset and Methodology

We extracted 647 such statements from a corpus of 3100 statements having coordinating conjunctions '*and*' and '*or*'. As discussed in section 2.1, S-conjunctions are not nocuous in nature; therefore we removed such statements with S-conjunctions. S-conjunctions contributed to 26% of the total 647 statements having coordinating structures. Manual analysis of the set of 647 statements indicated that searching for '*and*' and '*or*' has yielded in few statements having correlating conjunctions like '*either-or*' and '*both-and*'; these constructs are also not ambiguous. Therefore, we dropped nearly 36 such statements from the set of 647 extracted statements. We also dropped statements having phrases (158 statements) to either side of the coordinating conjunctions. The reason for dropping such statements is that the heuristics used in our study to identify coordinating constituent are word-based similarity measures. The presence of phrases was verified by the subjects involved in annotation task during discussions and, the decision to drop any such statement was taken unanimously. An example of such a statement is:

(4) Such issues in requirements included the draft Web Form XYZ and several other documents.

Here, '*and*' conjoins two phrases, namely: '*the draft Web Form XYZ'* and '*several other documents'*. The idea of pre-processing the dataset before proceeding with the study was to ensure that we have instances of those coordinating structures that conjoin two words (nouns, adjectives or verbs) and are potentially ambiguous. The exercise of pre-processing the dataset resulted in 284 potentially ambiguous statements out of 647 statements.

We presented the set of these potentially ambiguous statements to seven subjects for annotating the statements as having nocuous or innocuous ambiguity. We chose subjects from varying background to ensure unbiased labeling. Two of the subjects are software professionals, one teaches Software Engineering and one is an English linguist. Rest of the subjects includes master students of Software Engineering. Since manual annotation is a subjective matter, therefore it could be a potential threat to the validity of our results. We mitigated this threat in two ways. First, we explained nocuous and innocuous ambiguity to the subjects in a meeting. We ensured that subjects are familiar and comfortable with the annotation task by performing a validity check. We requested the subjects to label a random sample of 50 statements and, resolved the doubts and queries of the subjects in that meeting. Performing peer review of these annotations indicated that the subjects are comfortable with the annotation task. Second, we used confidence function to decide final labels for the training dataset.

Since we have limited set of labeled training set for learning, we adopted semi-supervised method of learning and training the classifier as discussed in section 3.2. We first designed a confidence function to decide whether a statement should be marked as innocuously ambiguous or as a nocuous statement. We took the counts of nocuous and innocuous labels for each statement and divided that count by the total number of subjects. If the resultant value is above a threshold value, then we consider the statement as potentially ambiguous (nocuous). We observed that keeping threshold at 57% yields better results. Our approach of final labeling is illustrated through table – 1:

Table 1. Identifying nocuous and innocuous labels

Coordinating Structure	**Judgments**		
Sample scenarios	**Nocuous**	**Innocuous**	**> 57%**
preparation and *submission*	4	3	YES
number and *license*	5	2	YES
terms and *conditions*	2	5	NO
retrieve and *view*	0	7	NO

Our approach of labeling is similar to the one followed in [9]. However, we differ in our approach from them in two ways. The authors in [9] have judged ambiguities on three criterions: high attachment of modifier, low attachment of modifier and ambiguous. On the contrary, our judging criterion is whether the coordinating units can be considered as a constituent or not. If the coordinating units cannot be considered as a constituent, then we have an instance of nocuous ambiguity else it is an instance of innocuous ambiguity. Second, we have further improved our learning by identifying the heuristics that yield in better classification and clustering. As discussed in section 3.2, we have used attribute evaluator algorithm (Gain Ratio Attribute Evaluator in our case) to identify the most contributing heuristics towards clustering and classification of our dataset of requirements statements. We found that 'jcn' and 'lin' heuristics are top ranking features for classifying the requirements. These heuristics are then followed by 'wup' and 'res'. We also validated these observations by performing unsupervised clustering (using the top ranking heuristics). We evaluated the clusters using 'Expectation-Maximization' (EM) and 'Simple K-Means'. The results of unsupervised clustering further strengthened our observations regarding better heuristics to use for learning and training the classifiers. Table 2 summarizes our observations of unsupervised clustering:

Table 2. Unsupervised Clustering

Algorithm	**Number of Clusters**	**Degree of Belongingness**
EM	3	(42%, 45% and 10%)
Simple K-Means	2	(34% and 63%)

4.2 Results and Observations

We present the results and observations in terms of the research questions we mentioned in the introduction section:

1. Which ML algorithm is best suited for identifying nocuous coordination ambiguities?

- Naïve Bayes algorithm is best suited for identifying the nocuous coordination ambiguities as indicated through

evaluation metrics presented in table 3. Since Naïve Bayes algorithm gives us a recall of 90.4% and we are interested in having higher recall, therefore it is best suited for our goal. Naïve Bayes algorithm is based on the assumption that presence or absence of a particular feature is unrelated to the presence or absence of any other feature. The algorithm has been proved to be quite efficient in supervised setting with feature-independence assumption and also, even when the features do not follow conditional independence assumption [25]. Our observation that Naïve Bayes outperforms other classifiers in our case because the heuristics that form feature vector in our case are independent of each other..

Table 3. Classification Results using all Heuristics except distr

Classifier	Precision	Recall	F-Measure
Naïve Bayes	0.387	**0.904**	**0.542**
K-NN	0.469	0.51	0.488
Random Tree	**0.486**	0.519	0.502
Random Forest	0.464	0.433	0.448

2. Which heuristics serve best to identify nocuous coordination ambiguities?

- Our observations after executing attribute evaluator and unsupervised clustering indicate that 'jcn' and 'lin' are best suited to identify nocuous coordinating terms. The recall for ambiguous (nocuous) requirements is quite high using 'jcn' alone and using 'jcn' and 'lin' together. The best recall is achieved using 'wup', 'jcn' and 'lin' together as shown in table 4. The results in table 4 correspond to Naïve Bayes classifier only.

Table 4. Classification Results using combination of Heuristics

Heuristic	Precision	Recall	F-Measure
jcn	0.371	**0.981**	0.538
jcn-lin	0.37	0.981	0.538
wup-jcn-lin	0.376	**0.99**	**0.545**
hso-wup-jcn-lin	0.374	0.971	0.537

3. Are distributional heuristics for identifying nocuous coordination ambiguity instances helpful in context of requirements specifications?

- We have discussed in detail in section 3 that distributional similarity heuristics are quite useful for identifying nocuous and innocuous coordination ambiguities for general purpose like news and discourse analysis. However, for requirements that pertain to a particular domain and are expressed in technical language, referring to a general corpus will not prove very beneficial. Our viewpoint is supported by our evaluation study where we found that distributional heuristic (measured over BNC) yields zero recall individually. It also lowers the value of recall when combined with other combinations of similarity heuristics. The corresponding results with Naïve Bayes classifier are shown in table 5:

Table 5. Classification Results with Distributional Similarity Heuristics

Classifier	Precision	Recall	F-Measure
Distr	0	**0**	0
distr-all hesuristic	0.389	0.**894**	0.542
distr-jcn-lin	0.375	**0.971**	0.542
distr-wup-jcn-lin	0.374	0.971	0.542

5. CONCLUSION

In this paper, we have presented an approach towards identifying nocuous coordination ambiguities in the requirements specifications and we have presented results based on our approach. Our evaluation study indicates that constituency test for coordinating conjunctions serves as a relatively better tool for designing nocuous coordination ambiguity classifier. Our approach has resulted in 99% recall for nocuous coordination ambiguity instances using the three similarity measures, namely proposed by Wu and Palmer; Jiang and Conrath; and, Lin, together. Such a high value of recall can relieve analysts from reviewing the requirements specification documents repetitively and carefully to find ambiguous statements with coordinating conjunctions. Our approach can further assist analysts by presenting him with the identified nocuous coordination ambiguities, which is only a small subset of large requirements document. The analyst can, then, correct and refine the identified requirements statements without worrying for the complete document. Our study also indicates that word-based similarity measures are relatively effective than distributional similarity measure for identifying nocuous coordination ambiguities in context of requirements specifications. We further aim to test our approach by using statements from a particular business domain as a test-set to our trained Naïve Bayes classifier.

6. ACKNOWLEDGMENTS

We would like to thank Dr. Tulika Chandra, Faculty in English Department at Shiv Nadar University for her help in our work on coordination ambiguity.

7. REFERENCES

[1] Kamsties, E. 2001. *Surfacing Ambiguity in Natural Language Requirements*. Doctoral thesis, Fraunhofer IESE, Kaiserslauttern, Germany.

[2] Resnik, P. 1999. Semantic Similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence*, 11, 95-130.

[3] Chantree, F., Nuseibeh, B., Roeck, de, A. and Willis, A. 2006. Identifying Nocuous Ambiguities in Natural Language Requirements. In *Proceedings of IEEE Conference on Requirements Engineering* (Minnesota, USA, September 11-15, 2006). RE'06. IEEE, 59-68.

[4] Wu, H. and Furugori, T. 1998. A Computational Method for Resolving Ambiguities in Coordinate Structures. In *Proceedings of Pacific Asia Conference on Language,*

Information and Computation (February 18-20, 1998). PACLIC, 1998. 263-270.

[5] Goldberg, M. 1999. An unsupervised model for statistically determining coordinate phrase attachment. In *Proceedings of 37th Annual Meeting on Association for Computational Linguistics* (Maryland, USA, June 20-26, 1999). Association for Computational Linguistics, 610-614.

[6] Banik, E. 2004. Semantics of VP coordination in LTAG. In *Proceedings of 7th International Workshop on Tree Adjoining Grammar and Related Formalism* (Vancouver, Canada, May 20-22, 2004). 118-125.

[7] Brouwer, H., Fitz, H. and Hoeks, C.J.J. 2010. In *Proceedings of Workshop on Cognitive Modeling and Computational Linguistics* (Uppsala, Sweden, July 15, 2010). Association for Computational Linguistics, 72-80.

[8] Chantree, F., Kilgarriff, A., Roeck, de, A. and Willis, A. 2005. Disambiguating Coordinations Using Word Distribution Information. In *Proceedings of International Conference on Recent Advances in Natural Language Processing* (Borovets, Bulgaria, September 21-23, 2005). RANLP-2005. 59-68.

[9] Yang, H., Willis, A., Roeck, de, A. and Nuseibeh, B. 2010. Automatic Detection of Nocuous Coordination Ambiguities in Natural Language Requirements. In *Proceedings of 25th IEEE/ACM International Conference on Automated Software Engineering* (Antwerp, Belgium, September 20-24, 2010). ASE-2010. ACM, 53-62.

[10] Carnie, A. 2013. *Syntax: A Generative Introduction, 3rd edition.* Oxford: Wiley Blackwell.

[11] Kocaguneli, E., Cukic, B. and Lu, H. 2013. Predicting More from Less: Synergies of Learning, In *Proceedings of 2nd Workshop on Realizing Artificial Intelligence Synergies in Software Engineering* (California, USA, May 25-26, 2013) RAISE-2013, IEEE, 42-48.

[12] Kurohashi, N. and Nagao, M. 1994. A Syntactic Analysis Method of Long Japanese Sentences Based on the Detection of Conjunctive Structures. *Computational Linguistics,* 20(4), 507-534.

[13] Kilgaariff, A., Rychly, P., Smrz, P. and Tugwell, D. 2004. The sketch engine. In *Proceedings of 11th European Association for Lexicography International Congress* (France, Europe, July 6-10, 2004) EURALEX-2004, EURALEX.

[14] Wasow, T., Perfors. A. and Beaver, D. 2005. The Puzzle of Ambiguity. In *Orgun, O. and Sells, P. (eds) Morphology and The Web of Grammar: Essays in Memory of Steven G. Lapointe*. CSLI Publications. 2005.

[15] Han, J., Kamber, M. and Pei, J. 2011. *Data Mining: Concepts and Techniques,* 3rd edition, Morgan Kaufmann.

[16] Chapelle, O., Scholkopf, B. and Zien, A. 2006, *Semi-Supervised Learning*, MIT Press.

[17] Patwardhan, S., Banerjee, S. and Pedersen, T. 2003. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics* (Mexico City, Mexico, February 16-22, 2003) CICLing 2003, 241-257.

[18] Jiang, J. and Conrath, D. 1997. Semantic Similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference on Research in Computational Linguistics* (Taiwan).

[19] Lin, D. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning* (Madison, WI) ICML-1998.

[20] Wu, Z. and Palmer, M. 1994. Verb semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, (Las Cruces, New Mexico).

[21] Leacock, C. and Chodorow, M. 1998. Combining local context and WordNet sense similarity for word sense identification. In *WordNet, An Electronic Lexical Database.* The MIT Press.

[22] Lesk, M.E. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the SIGDOC Conference* (Toronto, Canada, June).

[23] Hirst, G. and St-Onge, D. 1998. Lexical Chains as representations of context for the detection and correction of malaproprisms. In Fellbaum, C. (ed.) *WordNet, An Electronic Lexical Database.* The MIT Press, Cambridge, MA, 305-332.

[24] Patwardhan, S. 2003. *Incorporating dictionary and corpus information into a context vector measure of semantic relatedness*. Master's Thesis, University of Minnesota, Duluth.

[25] Zhang, H. 2004. The Optimality of Naïve Bayes, In *Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference* (Florida, USA, May 17-19, 2004) FLAIRS-2004, AAAI Press.