Semantic Grounding in Dialogue for Complex Problem Solving

Xiaolong Li

Department of Computer Science
North Carolina State University
Raleigh, NC, 27695
xli30@ncsu.edu

Kristy Elizabeth Boyer

Department of Computer Science
North Carolina State University
Raleigh, NC, 27695
keboyer@ncsu.edu

Abstract

Dialogue systems that support users in complex problem solving must interpret user utterances within the context of a dynamically changing, user-created problem solving artifact. This paper presents a novel approach to semantic grounding of noun phrases within tutorial dialogue for computer programming. Our approach performs joint segmentation and labeling of the noun phrases to link them to attributes of entities within the problem-solving environment. Evaluation results on a corpus of tutorial dialogue for Java programming demonstrate that a Conditional Random Field model performs well, achieving an accuracy of 89.3% for linking semantic segments to the correct entity attributes. This work is a step toward enabling dialogue systems to support users in increasingly complex problem-solving tasks.

1 Introduction

In the dialogue systems research community, there is growing recognition that dialogue systems need to support users in increasingly complex tasks. To move in this direction, dialogue systems must perform natural language understanding within richer and richer contexts, and this understanding includes semantic interpretation of user utterances (Traum, et al., 2012, Rudnicky, et al., 1999). Previous approaches for semantic interpretation include domain-specific grammars (Lemon et al., 2001) and open-domain parsers together with a domain-specific lexicon (Rosé, 2000). However, existing techniques are not sufficient to support increasingly complex problem-solving dialogues due to several challenges. For example, domain-specific grammars become intractable when applied to more ill-formed domains, and open-domain parsers may not perform well across domains (McClosky et al., 2010).

The call for addressing these limitations is particularly strong for dialogue systems that help people learn, such as *tutorial dialogue systems*. Today's tutorial dialogue systems engage in natural language dialogue in support of tasks such as solving qualitative physics problems (VanLehn et al., 2002), understanding computer architecture and physics (Graesser et al., 2004), and predicting behavior of electrical circuits (Dzikovska et al., 2011). Although these systems differ in many ways, they have an important commonality: in order to semantically interpret user dialogue utterances, these systems ground the utterances in a fixed domain description that is an integral part of the engineered system. This characteristic is shared by most dialogue systems, which ground their dialogue in manually defined domain-specific ontologies, such as for the task of booking flights (Allen, et al., 2001), checking bus schedules (Raux, 2004), and finding restaurants (Young et al., 2007).

These task-oriented domains, though they present a rich set of research challenges, stand in stark contrast to a *complex problem-solving* domain in which the user is creating an artifact to solve a problem. Yet the psychology literature tells us that complex problem solving is an essential activity in human learning (Greiff et al., 2013; Mayer et al., 2006; Funke, 2010). In such a domain, understanding user dialogue utterances involves grounding them within an infinite set of possible user-created

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 841–850,
Denver, Colorado, May 31 – June 5, 2015.

artifacts, not within a system ontology. This paper focuses on the complex problem-solving domain of introductory computer programming. In this domain the user might say, for example, "Is `myVariable` supposed to be an `int`?" where `myVariable` refers to the name of a variable within the computer program that the user has created. The semantic interpretation task in this case is akin to situated dialogue where user utterances must be grounded within a physical environment (Liu et al., 2014, Gorniak et al., 2007). However, even these situated dialogue models typically rely on a world defined by a limited number of entities (e.g., a chair or a cup).

To address these challenges, this paper presents a step toward semantic grounding for complex problem-solving dialogues, in which the number of potential entities (e.g., a Java variable or a piece of code) is infinite. The present work focuses on the semantic understanding of *noun phrases*, which tend to bear significant semantic information for each utterance. Although noun phrases are typically small in their number of tokens, their complexity and semantics vary in important ways. For example, in the domain of computer programming, two similar noun phrases such as "the 2 dimensional array" and "the 3 dimensional array" refer to two different entities within the problem-solving artifact. Inferring the semantic structure of the noun phrases is necessary to differentiate these two references within a dialogue, to ground them in the task, and to respond to them appropriately.

This noun phrase grounding task is similar to coreference resolution, which discovers the relationship between pairs of noun phrases in a piece of natural language text (Culotta, Wick, & Mccallum, 2007; Lappin & Leass, 1994). However, different from coreference resolution, noun phrase grounding links natural language expressions to entities in a real world environment. The current approach leverages the structure of noun phrases, mapping their segments to attributes of entities to which they should be semantically linked. In order to overcome the limitation of needing to fully enumerate the entities in the environment, we represent the entities as automatically extracted vectors of attributes. We then perform joint segmentation and labeling of the noun phrases in user utterances to map them to the entity vectors (used to describe entities within the environment). This mapping of noun phrases to real-world attributes is the grounding task focused on in this work. The results show that a Conditional Random Field performs well for this task, achieving 89.3% accuracy. Moreover, even in the absence of lexical features (using only dependency parse features and parts of speech), the model achieves 71.3% accuracy, indicating that it may be tolerant to unseen words. The flexibility of this approach is due in part to the fact that it does not rely on a syntactic parser's ability to accurately segment within noun phrases, but rather includes parse features as just one type of feature among several made available to the model. Finally, in contrast to methods based on bag-of-words such as latent semantic analysis, the proposed approach models the structure of noun phrases to facilitate specific grounding within an artifact.

The remainder of this paper is structured as follows. Section 2 presents related work on semantic interpretation and on natural language interpretation for tutorial dialogue. Section 3 describes the corpus and highlights some of the characteristics of dialogue for complex problem solving. The semantic interpretation approach is introduced in Section 4, with the experiments and results presented in Section 5. Section 6 concludes with important directions for future work.

2 Related Work

The approach presented in this paper draws upon a rich foundation of research in semantic interpretation and specifically upon dialogue interpretation for tutorial dialogues. Each of these areas of related work is discussed in turn.

2.1 Semantic Interpretation

The current work is closely related to several well-established research directions within the computational linguistics literature: semantic role labeling, semantic parsing, and language grounding. Semantic tagging assigns a semantic role label to text segments in a sentence (Pradhan, et. al, 2004). The set of semantic roles are relatively coarse-grained, not mapping to specific entities within the world. In contrast, the approach used in this paper does perform semantic role labeling, but the semantic grounding of these text segments are extracted at the same time. Semantic parsing addresses a more complex problem than semantic role labeling: in-

terpreting the semantic structure of a sentence. Supervised semantic parsing requires a target logical form for each sentence, which is costly (Zettlemoyer et. al, 2012). Unsupervised methods rely on accurate dependency parsing, and the semantics learned with unsupervised methods are not directly grounded in a domain (Poon et al., 2009). Our approach does not require a logical form or accurate parse in order to train the model.

Another aspect of semantic interpretation involves language grounding, which links natural language to representations of entities in the (often physical) world directly. Matuszek et al. (2012) propose a joint language/perception model to learn attribute names in a physical environment. Barnard et al. (2003) learn interpretation of segments of images in words with a number of models. Liu et al. (2014) label the referential entities in a collaborative discourse with graph mapping. All of these approaches work in scenarios in which the number of entities is limited. This is different from the case of a complex problem-solving domain in which there could be infinitely many combinations of entities and surface forms of the problem-solving artifact. Thus, building an entity graph to model the relationships between entities would be intractable. Grounding based on semantic interpretation using our approach will address this problem since it first narrows down the category of entity for a noun phrase and then grounds within a family of factorized vectors.

2.2 Language Understanding in Tutorial Dialogue Systems

All dialogue systems employ some form of semantic interpretation. Within tutorial dialogue, some dialogue interpretation relies on a manually defined domain-specific grammar and lexicon (Lemon et al., 2001, Evens et al., 2005). CIRCSIM-Tutor (Evens et al., 2005), a tutorial dialogue system in the domain of cardiovascular physiology, uses a set of finite state transducers and a domain-specific lexicon. Such domain-specific grammars are successful within well-formed domains but become unwieldy in larger or ill-defined domains.

Another approach is to employ an open-domain parser in combination with domain-specific knowledge. CARMEL (Rosé, 2000) is a natural language understanding component that has been used in multiple dialogue systems (Zinn et al., 2000; Litman, 2004; VanLehn et al., 2002). CARMEL uses a semantic interpretation framework that performs semantic interpretation with semantic constructor functions during syntactic parsing. The semantic interpretation employs encoded domain-specific semantic knowledge and a frame-based representation.

Dzikovska et al. (2007) proposed an approach that divides the logical form representation of utterances and the knowledge representation ontology in order to make a NLU component adaptable for multiple domains. The logical form representation contains high-level word sense and semantic role labels. Then, a contextual interpreter is employed for mapping between the logical form and the domain ontology. This work still relies on an open domain parser to generate the logical forms.

Different from all of the approaches mentioned above, AutoTutor (Graesser et al., 2004) uses latent semantic analysis (LSA) to evaluate students' utterances by comparing them to a handcrafted expected answer. LSA represents semantics as a high-dimensional vector and computes similarity between pieces of text. As a bag-of-words approach, LSA does not capture the kind of semantic structure that facilitates specific language grounding in an environment.

3 Corpus of Complex Problem Solving Dialogue

Complex problem solving is defined within the psychology literature as the process of reaching a goal state by applying multiple problem solving skills, when the desired goal state cannot simply be reached by applying one from a set of existing solution patterns (Greiff et al., 2013; Mayer et al., 2006; Funke, 2010). Dialogue surrounding complex problem solving is therefore grounded within a problem-solving artifact that could have infinitely many surface forms. The complex problem-solving domain that is the focus of this paper is computer programming, specifically Java programming, and the corpus under consideration reflects textual tutorial dialogue exchanged between two humans in support of that problem solving.

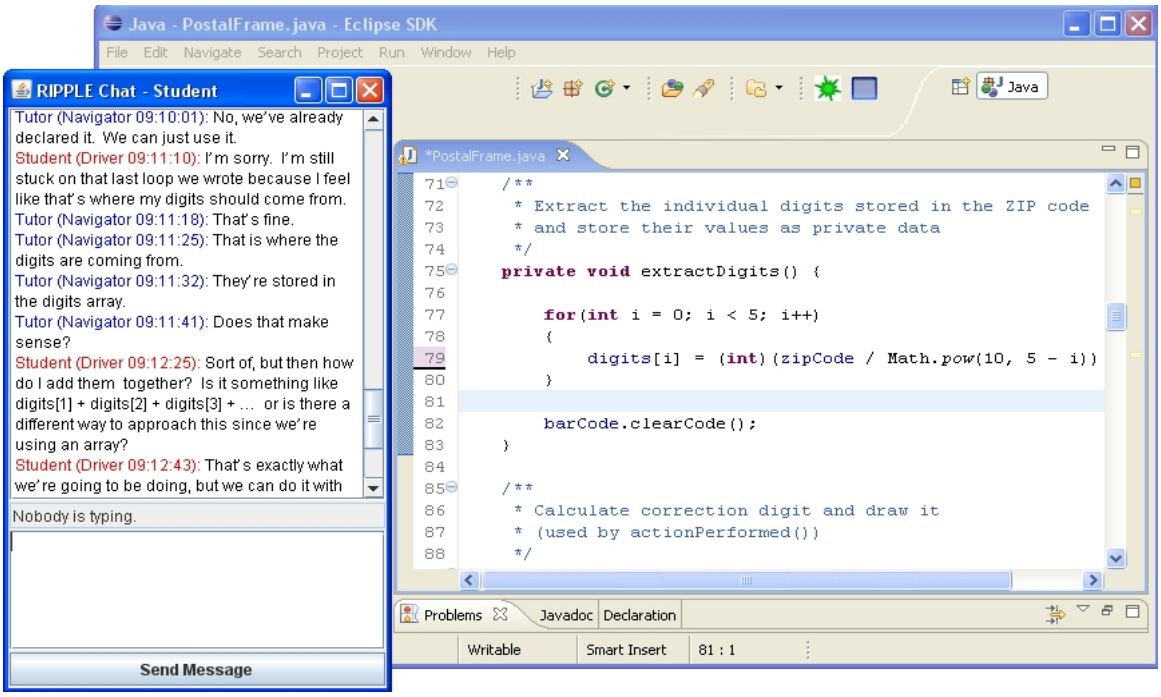

Figure 1. Tutorial dialogue interface.

The corpus was collected within a tutorial dialogue study in which human tutors and students interacted through a tutorial dialogue interface that supported remote textual communication (Boyer et al., 2011) The tutorial dialogue interface (Figure 1) consists of two windows that display interactive components: the student's Java code, the compilation or execution output associated with the code, and the textual dialogue messages between the student and tutor. All of the information in these two windows was synchronized between the student's screen and tutor's screen in real time.

The corpus contains 45 Java programming tutorial sessions from student-tutor pairs, with a total of 4857 utterances, an average of 108 utterances per session. For the current work, six of these tutorial sessions were manually annotated for their semantic grounding (as described in Section 5), a total of 758 utterances. The problem students solved during this tutorial dialogue involved creating, traversing, and modifying parallel arrays. This task was challenging for students and represented a complex problem-solving effort since the students were novices who were enrolled in an introductory computer programming class.

The dialogues within this domain are characterized by situated features that pertain to the programming task. A portion of user utterances refer to general Java knowledge, and in these cases semantic interpretation can be accomplished by mapping to a domain-specific ontology (e.g., Dzikovska et al., 2007). In contrast, many utterances refer to concrete entities within the dynamically changing, user-created programing artifact. Identifying these entities correctly is crucial for generating specific tutorial dialogue moves. A dialogue excerpt is shown in Figure 2.

4 Methodology

To ground the dialogue utterances as described in the previous section, our approach focuses first upon noun phrases, which contain rich semantic information. This section introduces the approach, based on Conditional Random Fields, to jointly segment the noun phrases and link those segments to entities within the domain.

4.1 Noun Phrases in Domain Language

A noun phrase is defined as "a phrase which has a noun (or indefinite pronoun) as its head word, or which performs the same grammatical function as such a phrase" (Crystal, 1997). The syntactic structure of a noun phrase consists of dependents which could include determiners, adjectives, prepositional phrases, or even a clause. For example, let us con-

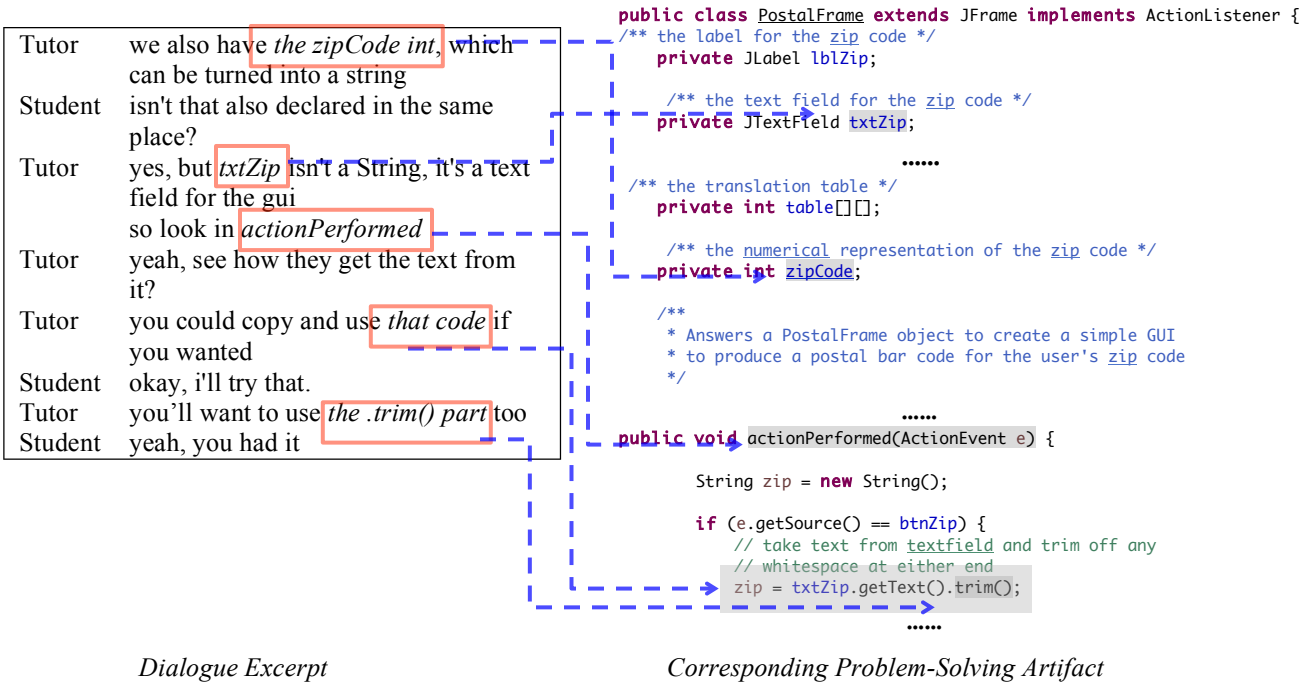

Figure 2. Dialogue excerpt from the corpus.

sider the noun phrase *"a 2 dimensional array"*. Its head is *"array"* and its dependents are *"a"* as the determiner and "2 dimensional" as an adjective phrase. In this simple case the syntactic boundaries also indicate semantic segments, as these dependents indicate one or more attributes of the head. If this relationship were always true, the semantic structure understanding task would be a labeling task that only requires assigning a semantic tag to each syntactic segment of the noun phrase. But this is not always true, in part because a syntactic parser trained on an open-domain corpus will not necessarily perform well on domain language (McClosky et al., 2010). For example, in the noun phrase *"the outer for loop,"* which also occurs in the Java programming corpus, the head of the noun phrase is "*for loop*," but the syntactic parse (generated by the Stanford parser) of this noun phrase understandably (but incorrectly) identifies this head as part of a prepositional phrase (Figure 3).

To address this challenge, this paper utilizes a joint segmentation and semantic labeling approach that does not rely on accurate syntactic parsing within noun phrases. In this approach the head and dependents of each noun phrase are each referred to as a *segment,* with exactly one segment per dependent, and one or more words per segment. Identifying these segments correctly is essential to correct assignment of semantic tags. Pipeline methods for semantic segmentation rely on stable performance of an open domain parser, but as described above, this assumption is not desirable for grounding some domain language. We therefore utilize joint segmentation and labeling and apply a Conditional Random Field approach (Lafferty, 2001), a natural choice for the sequential data segmentation and labeling problem.

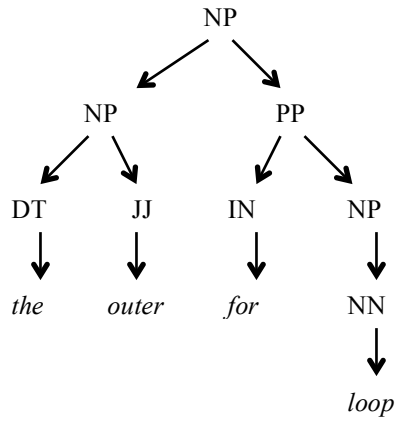

Figure 3. A parse of *"the outer for loop"* from Stanford Parser.

4.2 Description Vector

The goal is to ground each noun phrase to attributes of entities within the problem-solving artifact, which constitutes the "world" in this domain. To do this, we will link each semantic segment in a noun phrase to an attribute of an entity in the world. Because the world can contain any of an infinite set of user-created entities, representation cannot rely upon exhaustively enumerating the entities. To represent an entity in the domain, we define a description vector V which defines the attribute types for entities in the domain. Then, an entity O in the domain is represented uniquely by an instance of V. The values of each V_i indicate the value of the attribute i of O, as illustrated in Table 1. This definition of the description vector relies upon the structure of the domain by factorizing the attributes of entities.

With this representation, grounding a noun phrase involves linking each segment of the noun phrase to an attribute in the description vector. Formally, we represent a noun phrase as a series of segments:

$$NP = < s_1, s_2, ..., s_k >$$

where s_i is the i_{th} segment in this noun phrase. A noun phrase is also a sequence of words:

$$NP = < w_1, w_2, ..., w_n >$$

where each w_j is the j^{th} word in the noun phrase. Therefore each segment is a series of words:

$$s_i = < w_j, w_{j+1}, ..., w_{j+l-1} >$$

where l is the length of semantic segment i.

Given a noun phrase, the segmentation problem is thus choosing a segmentation that maximizes the following conditional probability:

$$p(< s_1, s_2, ..., s_k > \mid < w_1, w_2, ..., w_n >)$$

Complementary to the segmentation problem is the semantic linking problem, which is to link s_i to an attribute a_i, which is the label of the i^{th} attribute in the entity description vector. That is, we wish to maximize the probability of the attribute label sequence a given the segments of the noun phrase:

$$p(< a_1, a_2, ..., a_k > \mid < s_1, s_2, ..., s_k >)$$

Taking consecutive words with the same attribute label as the same semantic segment, the noun phrase segmentation and semantic linking problem is then:

$$\underset{a}{argmax}\, p(< a_1, a_2, ..., a_n > \mid < w_1, w_2, ..., w_n >)$$

In the tag sequence $<a_1, a_2, ..., a_n>$, if a_i and a_{i+1} are the same, then w_i and w_{i+1} are assigned to the same semantic segment with tag a_i. The process of segmentation and semantic linking is illustrated in Figure 4.

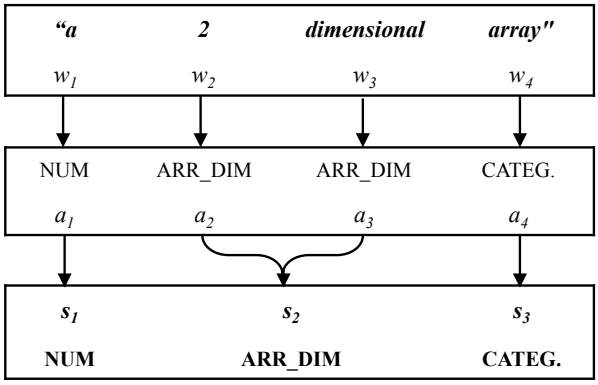

Figure 4. Segmentation and semantic linking of NP *"a 2 dimensional array."*

4.3 Joint Segmentation and Labeling

In order to perform this joint segmentation and labeling, we utilize a Conditional Random Field (CRF), which is a classic approach for sequence segmentation and labeling (Lafferty et al., 2001). Given the linear nature of our data, we employ a linear chain CRF. Specifically, given a sequence of words w, the probability of a label sequence a is defined as

$$p(a \mid w) = \frac{1}{Z(w)} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(i, w, a_i, a_{i-1})\right)$$

where $f_j(i,w,a_i,a_{i-1})$ is a feature function. The weights λ_j of this feature function are learned within the training process. The normalization function $Z(w)$ is the sum over the weighted feature function for all possible label sequences:

$$Z(w) = \sum_{a} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(i, w, a_i, a_{i-1})\right)$$

The optimal labeling $\hat{a}$ is the one that maximizes the likelihood of the training set, where K is the number of noun phrases in the corpus.

$$\hat{a} = argmax\left(\sum_{i=1}^{K} \log P(a^{(i)} \mid w^{(i)})\right)$$

4.4 Features

Next, we introduce the features used to train the CRF. The feature function $f_j(i,w,a_i,a_{i-1})$ was defined as a binary function, in which w is a feature value. We use both lexical and syntactic features. In a trained CRF model, the value of $f_i(i,w,a_i,a_{i-1})$ is known given a combination of parameters (i,w,a_i,a_{i-1}). The features used in the CRF model include words themselves, word lemmas, parts of speech, and dependency relationships from the syntactic parse. The word itself, lemmatized words and parts-of-speech have all been shown useful within segmentation and labeling tasks, so they are made available here (Xue et al., 2004). Each of these features is represented as categorical data. For example, a word is represented as its index in a list of all of the words that appeared in the corpus.

The dependency structure of natural language has also been shown to be important in semantic interpretation (Poon et al., 2009). This paper employs a dependency feature vector extracted from dependency parses. The head word of each noun phrase is the root of the dependency tree. Each dependent is a sub-tree directly under the head. We design the dependency feature as a sequence of dependency labels as follows.

Given a dependency tree, words in each semantic segment of the noun phrase are assigned a tag according to the relationship between them and the head. The relationship between each segment and head is defined by the dependency type in the dependency tree. For example, the dependency tree of *"a 2 dimensional array"* is shown in Figure 5. The dependency features are *<det,amod,amod,root>*. In this way, the dependency information from an open-domain parser is encoded as a feature to the semantic grounding model.

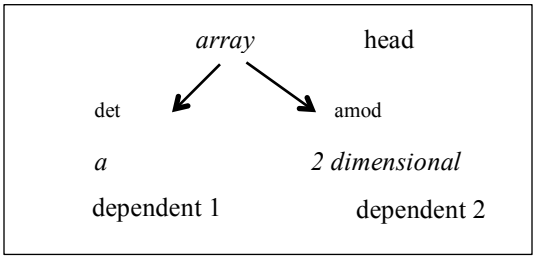

Figure 5. Dependency structure of *"a 2 dimensional array."*

5 Experiments & Results

The goal of the experiments is to determine how well the trained CRF can segment noun phrases and link these segments to the correct attribute of entities in the world. This section presents the experiments using CRFs trained and tested on the Java programming tutorial dialogue corpus. As described below, the results were evaluated by comparing with manually labeled data.

Noun phrases from the tutorial dialogues were first manually extracted and annotated as to their slots in the description vector described in Section

Attributes	**Meaning (in Java programming)**	**Example**
CATEGORY	Category of an entity	`Method, Variable, etc.`
NAME	Variable name; often user-created	`extractDigit`
VAR_TYPE	Type of variable	`int, String, etc.`
NUMBER	Number of entities	`2`
IN_CLASS	The class that contains this entity	`postalFrame`
IN_METHOD	The method that contains this entity	`actionPerformed`
DIR_PARENT	Direct parent entity	`For_Statement, Method`
LINE_NUMBER	Line number	`67`
SUPER_CLASS	Superclass of this entity	`JFrame`
MODIFIER	Access modifier	`public, private, etc.`
ARRAY_TYPE	Type of Array	`int, char, etc.`
ARRAY_DIMENSION	Dimension of array	`2, 1`
OBJ_CLASS	The class an object instantiates	`PostalBarCode`
RETURN_TYPE	Return type	`String, int, etc.`
OTHER	Other attributes	`the, extra, etc.`

Table 1. Elements of entity description vector to which noun phrases are mapped.

4.2. There were 364 grounded noun phrases extracted manually from the six tutorial dialogue sessions used in the current work. Each of these noun phrases extracted has one or multiple corresponding entities in the programming artifact. Since each word in a noun phrase is linked to an element in the description vector, the indices in this vector were used as the label for each word. Annotation of all 346 noun phrases was performed by one annotator, and 20% of the noun phrases (70 noun phrases) were doubly annotated by an independent second annotator. The percent agreement was 85.3% and the Kappa was 0.765.

To extract features, the lemmatization and syntactic parsing were performed with the Stanford CoreNLP toolkit (Manning et al., 2014). Then, a CRF was trained to predict the label for each word in a new noun phrase. The training was performed with the crfChain toolbox (Schmidt, 2008).

We use ten-fold cross-validation to evaluate the performance of the CRF in this problem. Results with different feature combinations are shown in Table 2. Manually labeled data were taken as ground truth for computing accuracy, which is defined as the percentage of segments correctly labeled.

Recall that consecutive words with the same label in a noun phrase are treated as a segment. Therefore, if a segment s_{CRF} identified by the CRF has the same boundary and the same label as a segment s_{Human} in the noun phrase containing s_{CRF}, this segment s_{CRF} will be counted as a correct segment. Otherwise, s_{CRF} will be counted as incorrect. The accuracy is then calculated as the number of correct segments identified by the CRF divided by the number of segments annotated manually. As can be seen in Table 2, all of the models perform substantially better than a minimal majority class baseline of 43%, which would result from taking each word as a segment and assigning it with the most frequent attribute label.

The results demonstrate important characteristics of the segmentation and labeling model. First, unlike most previous semantic interpretation work, our semantic interpretation of noun phrases does not rely on accurate syntactic parse within noun phrases. Rather, we use a dependency parse from an open-domain parser as only one of several types of features provided to the model. These dependency features improved the model in most feature combinations (Table 2). The feature combination of words, lemmas, and dependency parses achieved the best accuracy, which is 4.8% higher than the model that only used word features. This difference is statistically significant (Wilcoxon rank-sum test; n=10; p=0.02).

features	accuracy
word	84.5%
word + lemma	85.5%
Word + Dep	87.22%
lemma + Dep	89.1%
word + lemma + Dep	**89.3%**
word + lemma + POS	86.9%
word + lemma + POS + Dep	88.7%
POS + Dep	71.3%

Table 2. Labeling accuracy.

Notably, the combination of part-of-speech features and dependency parse features still performed at 71.3% accuracy, indicating that to some extent, the method may be tolerant to unseen words.

6 Conclusion and Future Work

This paper has presented a technique for semantic grounding of noun phrases in a complex problem-solving domain, tutorial dialogue for computer programming. By performing joint segmentation and labeling of noun phrases from user utterances, and mapping those segments to attributes of entities within the problem solving artifact, we have made a first step toward grounding complex problem-solving dialogue within a dynamically changing artifact from a potentially infinite set of surface forms. While trained on a small subset of the corpus, the high accuracy of this model indicates that it may be successfully applied to the larger corpus without extensive additional manual annotations.

Several directions of future work are very promising. In order to fully support users in complex problem-solving dialogues, the field must move toward richer grounding of natural language utterances within complex artifacts across many domains. Additionally, generating specific and tailored dialogue feedback grounded in the artifact is a complementary area of research that holds the potential to increase the effectiveness of dialogue systems for supporting problem solving. It is hoped that this line of investigation will lead to dialogue systems that smoothly support a much broader range of human endeavors.

Acknowledgments

The authors wish to thank the members of the LearnDialogue group, especially Joseph Wiggins, at North Carolina State University for their helpful input. This work is supported in part by the National Science Foundation through grants IIS-1409639 and the STARS Alliance, CNS-1042468. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the authors, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

References

Allen, J. F., Byron, D. K., Dzikovska, M., Ferguson, G., Galescu, L., & Stent, A. (2001). Toward Conversational Human-Computer Interaction. *AI Magazine*, 22(4), 27.

Barnard, K., Forsyth, D., & Jordan, M. I. (2003). Matching Words and Pictures. *Journal of Machine Learning Research*, 3, 1107–1135.

Boyer, K. E., Phillips, R., Ingram, A., Ha, E. Y., Wallis, M. D., Vouk, M. A., & Lester, J. C. (2011). Investigating the Relationship Between Dialogue Structure and Tutoring Effectiveness: A Hidden Markov Modeling Approach. *International Journal of Artificial Intelligence in Education (IJAIED)*, 21(1), 65–81.

Crystal, D. (1997). *A Dictionary of Linguistics and Phonetics* (4th ed.). Oxford University Press.

Culotta, A., Wick, M., & Mccallum, A. (2007). First-Order Probabilistic Models for Coreference Resolution. *In Proceedings of the 2007 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* (pp. 81–88).

Dzikovska, M. O., Allen, J. F., & Swift, M. D. (2007). Linking Semantic and Knowledge Representations in a Multi-Domain Dialogue System. *Journal of Logic and Computation*, 18(3), 405–430. Retrieved from http://logcom.oxfordjournals.org/cgi/doi/10.1093/logcom/exm067

Dzikovska, M. O., Isard, A., Bell, P., Moore, J. D., Steinhauser, N., & Campbell, G. (2011). BEETLE II: An Adaptable Tutorial Dialogue System. *Proceedings of the 12th Annual SIGdial Meeting on Discourse and Dialogue*, 338–340.

Evens, M., & Michael, J. (2005). *One-on-One Tutoring by Humans and Computers*. Psychology Press.

Funke, J. (2010). Complex problem solving: A case for complex cognition? *Cognitive Processing*, 11(2), 133–142.

Gorniak, P., & Roy, D. (2007). Situated Language Understanding as Filtering Perceived Affordances. *Cognitive Science,* 31(2), 197–231.

Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A., & Louwerse, M. M. (2004). AutoTutor: A Tutor with Dialogue in Natural Language. *Behavior Research Methods, Instruments, & Computers*, 36(2), 180–192.

Greiff, S., Wüstenberg, S., Holt, D. V., Goldhammer, F., & Funke, J. (2013). Computer-based Assessment of Complex Problem Solving: Concept, Implementation, and Application. *Educational Technology Research and Development*, 61(3), 407–421.

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *In Proceedings of the International Conference on Machine Learning* (pp. 282–289).

Lappin, S., & Leass, H. J. (1994). An Algorithm for Pronominal Anaphora Resolution. *Computational Linguistics*, 20(4), 535–561.

Lemon, O., Bracy, A., Gruenstein, A., & Peters, S. (2001). The WITAS Multi-Modal Dialogue System I. *In Proceedings of INTERSPEECH* (pp. 1559–1562).

Litman, D. J. (2004). ITSPOKE : An Intelligent Tutoring Spoken Dialogue System. *In Demonstration Papers at the 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)* (pp. 5–8).

Liu, C., She, L., Fang, R., & Chai, J. Y. (2014). Probabilistic Labeling for Efficient Referential Grounding Based On Collaborative Discourse. *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 13–18).

Manning, C. D., Bauer, J., Finkel, J., & Bethard, S. J. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. *In the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (pp. 55–60).

Matuszek, C., Fitzgerald, N., Zettlemoyer, L., Bo, L., & Fox, D. (2012). A Joint Model of Language and Perception for Grounded Attribute Learning. *In Proceedings of the 29th International Conference on Machine Learning*.

Mayer, R. E., & Wittrock, M. C. (2006). Problem Solving. *Handbook of Educational Psychology*, 2, 287–303.

McClosky, D., Charniak, E., & Johnson, M. (2010). Automatic Domain Adaptation for Parsing. *In Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)* (pp. 28–36).

Poon, H., & Domingos, P. (2009). Unsupervised Semantic Parsing. *In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1–10).

Pradhan, S. S., Ward, W., Hacioglu, K., Martin, J. H., & Jurafsky, D. (2004). Shallow Semantic Parsing using Support Vector Machines. *In Proceedings of the 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* (pp. 233–240).

Raux, A., & Eskenazi, M. (2004). Non-Native Users in the Let's Go!! Spoken Dialogue System: Dealing with Linguistic Mismatch. *In Proceedings of the 2004 North American Chapter of the Association for Computational Linguistics (HLT-NAACL)* (pp. 217–224).

Rosé, C. P. (2000). A Framework for Robust Semantic Interpretation. *In Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference (NAACL)* (pp. 311–318).

Rudnicky, A., Thayer, E., Constantinides, P., Tchou, C., Shern, R., Lenzo, K., … Oh, A. (1999). Creating Natural Dialogs in the Carnegie Mellon Communicator System. *In Proceedings of the Sixth European Conference on Speech Communication and Technology, (EUROSPEECH)* (Vol. 4, pp. 1531–1534).

Schmidt, M., & Swersky, K. (2008). http://www.cs.ubc.ca/~schmidtm/Software/crfChain.html.

Traum, D., Devault, D., Lee, J., Wang, Z., & Marsella, S. (2012). Incremental Dialogue Understanding and Feedback for Multiparty, Multimodal Conversation. *Intelligent Virtual Agents*, 7502, 275–288.

VanLehn, K., Jordan, P. W., Rosé, C. P., Bhembe, D., Bottner, M., Gaydos, A., … Roque, A. (2002). The Architecture of Why2-Atlas: A Coach for Qualitative Physics Essay Writing. *In Proceedings of the Sixth International Conference on Intelligent Tutoring Systems* (Vol. 2363, pp. 158–167). Springer.

Xue, N., & Palmer, M. (2004). Calibrating Features for Semantic Role Labeling. *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 88–94).

Young, S., Schatzmann, J., Weilhammer, K., & Ye, H. (2007). The Hidden Information State Approach to Dialog Management. *Acoustics, Speech and Signal Processing* (Vol. 4, pp. 149-152).

Zettlemoyer, L. S., & Collins, M. (2012). Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. *In Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence* (pp. 658–666).

Zinn, C., Moore, J. D., Core, M. G., & Varges, S. (2000). The BE&E Tutorial Learning Environment (BEETLE). *In Proceedings of the Seventh Workshop on the Semantics and Pragmatics of Dialogue*.