Training an Integrated Sentence Planner on User Dialogue

Brian McMahan
Computer Science
Rutgers University
brian.mcmahan@rutgers.edu

Matthew Stone
Computer Science
Rutgers University
matthew.stone@rutgers.edu

Abstract

An appealing methodology for natural language generation in dialogue systems is to train the system to match a target corpus. We show how users can provide such a corpus as a natural side effect of interacting with a prototype system, when the system uses mixed-initiative interaction and a reversible architecture to cover a domain familiar to users. We experiment with integrated problems of sentence planning and realization in a referential communication task. Our model learns general and context-sensitive patterns to choose descriptive content, vocabulary, syntax and function words, and improves string match with user utterances to 85.8% from a handcrafted baseline of 54.4%.

1 Introduction

Natural language generation (NLG) in dialogue involves a complex array of choices. It's appealing to scale up NLG by training systems to make these choices with models derived from empirical data. Sometimes, these choices have a measurable effect on the flow of the interaction. Systems can plan such choices with a model of dialogue dynamics that predicts which utterances will fulfill communicative goals successfully and efficiently (Lemon, 2011; Janarthanam et al., 2011; Garoufi and Koller, 2011).

Other times, a wide variety of utterances work well (Belz and Gatt, 2008). In these cases, systems can instead be designed simply to choose those utterances that most closely resemble specified target behavior. This paper describes and evaluates a new data-driven methodology for training sentence planning and realization in interactive dialogue systems this way. Our work is particularly inspired by Walker et al. (2002), who train a dialogue sentence planner by annotating its possible outputs for quality; and Jordan and Walker (2005), who train a referring expression generator to match annotated human–human dialogue.

In text generation, researchers have been able to exploit automatic analysis of existing resources on such tasks as ordering words more naturally (Langkilde and Knight, 1998) and identifying named entities in line with attested mentions (Siddharthan and Copestake, 2004). However, previous work on training dialogue generation has involved the acquisition or annotation of relevant data *ad hoc*, for example by collecting human–human dialogue, running Wizard of Oz experiments, or rating system outputs. Our work is different: we use a bootstrapping approach that automatically mines interactions with a running prototype to adapt NLG to match users.

As described in Section 2, our work builds on the COREF system of DeVault and Stone (2009). COREF and its users chat together to identify simple objects in a visual scene. COREF is designed with reversible models of language and dialogue—it tracks users' utterances and its own utterances with the same data structures and represents them as updating the conversational state in parallel ways. Because of this symmetry, COREF's understanding of each user utterance determines an input–output pair that the system could take as a target for NLG. We explain the significance of learning from such data in Section 3. However, we argue in Sections 4 and 5 that this learning will yield significant results only if system and user do in fact turn out to make similar contributions to dialogue.

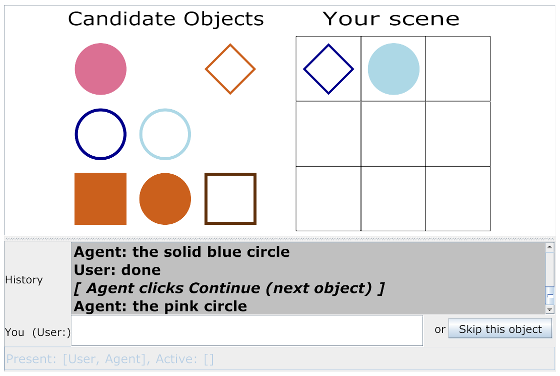
Our main experiment therefore uses data collected with a new version of COREF with more flexible strategies for taking initiative, as described in Section 6. We use the system's understanding of user utterances in the experiment, along with its productive capacity to generate alternative paraphrases of those utterances, to build an automatically labeled training set of good and bad NLG examples. We learn a model of the difference and evaluate its use in choosing novel utterances. As documented in Section 7, the learned model leads to improvements in naturalness over COREF's handcrafted baseline generator; our experiments document these improvements qualitatively and quantitatively.

Our work suggests new ways to design dialogue systems to adhere to formal models with guaranteed behavior (Paek and Pieraccini, 2008) while reaping the benefits of data-driven approaches (Rieser and Lemon, 2011) by improving themselves through ongoing interactions with users. Our experiments suggest that engaging with user expertise is a key factor in enabling such new design strategies. Our technique crucially exploits synergies in our domain between the architecture of the dialogue system, the specific dialogue policy that the system implements, and users' abilities to contribute to domain problem solving.

2 Background

COREF, short for "collaborative reference", communicates with users through a text-chat window for human–computer dialogue. A graphical interface provides task context and realizes domain actions; it orchestrates a basic referential communication task like those studied by Clark and Wilkes-Gibbs (1986) or Brennan and Clark (1996). In each round of interaction, the participants in the conversation are presented with a set of simple geometric shapes that they must talk about; the shapes are displayed on screen to human users and described as a knowledge base to the COREF agent. As the dialogue proceeds, one participant, assigned to work as the director, gets an indication of which object to describe next. The other participant, assigned to work as the matcher, must move this target object to its final disposition. Figure 1 is a snapshot of the interface in a session where the user works as matcher. Experimental sessions normally involve multiple rounds where participants alternate serving as director and as matcher.

COREF's architecture factors its reasoning into three integrative problem-solving modules, as shown in Figure 2. The modules use different algorithms and control flow, but are linked together by common representations and knowledge bases. One shared resource is COREF's prob-



Agent: the solid blue circle
User: done
[Agent clicks Continue (next object)]
Agent: the pink circle

Present: [User, Agent], Active: []

Figure 1: User's view of the chat interface in an interaction with COREF acting as director.

abilistic context model, which tracks the likely state of ongoing activity, maintains a linguistic context describing what has probably been said and what should be salient as a result, and represents the information available through the interface as grounded in interlocutors' perception. Another shared resource is COREF's tree-adjoining grammar (TAG; Joshi and Schabes (1997)), which assigns syntactic structures and semantic representations to utterances, and predicts what utterances will refer to in context and what dialogue moves they will contribute. Finally, both understanding and generation use a common representation of the interpretation of utterances, *utterance plans*, which associate specific strings of words with the updates that they are predicted to achieve via grammar and context.

The dialogue manager handles interaction with the user, coordinates understanding and generation, tracks updates to the context, and selects updates that COREF should contribute to the conversation. In case of ambiguity, the dialogue manager propagates uncertainty forward in time and works to resolve it through interaction. (COREF has general mechanisms for engaging in clarification subdialogues.) In fact, by the time each object has been identified, COREF has committed retrospectively, in light of what has happened, to a single most likely interpretation for everything the user has said about it. COREF has evidence that other interpretations it originally entertained were not what the user intended. This links each user utterance with a corresponding utterance plan that can be used for subsequent learning (DeVault and Stone, 2009).

The understanding module parses utterances using the grammar and resolves them using the con-

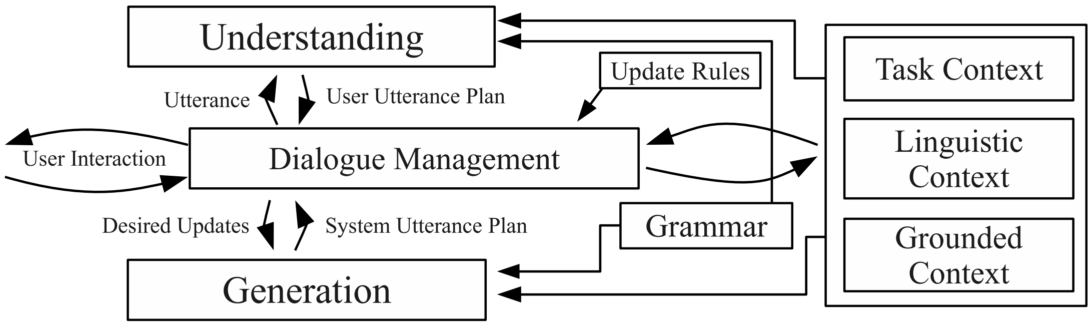


Figure 2: COREF system architecture, showing representations and knowledge shared across modules: utterance plans show how each agent's contributions follow from the system's representations of grammar and context; update rules map out consistent contextual effects for each agent's contributions.

text model to recognize the possible utterance plans behind them. The generator, meanwhile, uses the grammar and the context model to synthesize an utterance plan for a grammatical expression that is predicted to achieve some desired updates unambiguously, as in SPUD (Stone et al., 2003). A range of choices are folded together by this integrated problem-solving process. For example, the grammar specifies alternative realizations involving different syntactic frames and functional items, as in the paraphrases *'the target is a square'*, *'a square'* and *'square'*. The grammar also specifies lexical paraphrases, as in the equivalents *'dark blue'* and *'navy blue'* or *'beige'* and *'tan'*. SPUD's problem solving also creates choices about how much descriptive content to include in a reference, as *'the square'* versus *'the blue square'*, and what kind of descriptive content to include, as in *'the blue square'* versus *'the solid square'*. Full utterances involve all these choices, potentially in overlapping combinations, as in *'the target is the light brown object'* versus *'the solid square'*. See the Appendix for examples of NLG search, and DeVault (2008) for full details about COREF's design and implementation.

COREF's handcrafted NLG search heuristics draw on ideas from Stone et al. (2003) and Dale and Reiter (1995) to prioritize efficient, specific utterances which use preferred descriptive attributes and respect built-in preferences for certain words and constructions. When we implemented these heuristics, we had no intention of revising the model using learning. However, COREF's strategy never generates human-like overspecification, its lexical and syntactic choices are determined by hand-coded logical constraints, and it offers few tools to discriminate among comparable paraphrases. In principle, a system like COREF ought to be able to find out how people tend to make such choices in interacting with it, and learn to speak the same way. This is the central problem we address in this paper.

3 Related Work

Our key contribution is demonstrating that a dialogue system can bootstrap an integrated NLG strategy from interactions with a prototype system by training a model to imitate user utterances. This complements DeVault and Stone (2009), who train an interpretation model in a similar way. Bootstrapping NLG for dialogue requires new insights, and require us to synthesize of a number of trends in dialogue, in NLG and in social learning.

A number of researchers have trained generators for dialogue based on human specifications of desired output. For example, Walker et al. (2002) and Stent et al. (2004) optimize sentence plans based on expert ratings of candidate output utterances. Jordan and Walker (2005) learn rules for predicting the content of referring expressions to match patterns found in corpora of human descriptions in context. Garoufi and Koller (2011) tune the referential strategies of a general-purpose sentence planner based on metrics of utterance effectiveness mined from human–human interactions. Our work involves a new domain and for the first time involves integrated training of all these dimensions of NLG, but we draw closely on the architectures, features and learning techniques developed by these researchers. The key difference that they use data collected, and to some degree hand-annotated, specifically to train NLG.

At the same time, a range of research has explored the way existing data sets can improve NLG results. For example, Langkilde and Knight (1998) *n*-gram statistics to bias a non-deterministic realization system towards frequent utterances. Siddharthan and Copestake (2004) use references in corpora to bootstrap a generator for named entities in text. Such methods, however, have generally focused on offline text generation applications. Our research shows that specific infrastructure must be in place to tune NLG to a dialogue system's own experience.

In addition, our work finds echoes in work across AI on learning by imitation. Interactive robots can learn in new ways by modeling their behavior on competent humans (Breazeal et al., 2005). Other domains require agents to develop cooperative relationships and elicit meaningful behavior from one another before they can learn to act effectively together (Zinkevich et al., 2011). Our work helps to establish the connections of these ideas to dialogue.

Finally, we note that our work is orthogonal to a range of other research that aims to extend and improve NLG in dialogue through learning. Given specified target utterances, knowledge acquisition techniques can be used to induce new resources that describe those utterances for NLG as well as to optimize the use of those resources to match the corpus (Higashinaka et al., 2006; DeVault et al., 2008). Moreover, given a model of the differential effects of utterances on the conversation, reinforcement learning can be used to identify utterances with the best outcomes (Lemon, 2011; Janarthanam et al., 2011). We see no reason not to combine these techniques with imitation learning in the development of future systems.

4 Training COREF

Our method for mining COREF's dialogue experience involves three steps. First, we compile training data: positive instances are derived from user utterances and negative instances are derived from the generator's alternative realizations of communicative goals inferred from user utterances. Next, we build a machine learning model to distinguish positive from negative instances, using features describing the utterance itself, the current state of the conversation and relevant facts from the dialogue history. Finally, we apply the learned model on new NLG problems by collecting candidate paraphrases and finding the one rated most likely to be natural by the learned model.

4.1 Data Analysis

Each user utterance in COREF's interaction logs is associated with a particular state of the dialogue and with the utterance plan ultimately identified as its best interpretation. Our method extracts the task moves in the utterance plan as candidate communicative goals for the utterance. It swaps the role of the user and the system, so as to realize an NLG problem instance to plan a contribution with the utterance's inferred communicative goals, given the user's role in the dialogue and their reconstructed dialogue state. It then calls a revised version of the generator that's non-deterministic and accumulates a range of plausible solutions.[1]

This process automatically creates a representation of the NLG problem faced by the user and the set of possible solutions to that problem implicitly determined by COREF's models of language in context. Our method partitions the training instances based on how the user chose to solve the NLG problem. If the NLG output string matches what the user actually said here, it becomes a positive training example. If it differs from what the user actually said, it becomes a negative one.

4.2 Machine Learning

We can now build a machine learning model of this data set. Given an unlabeled candidate solution to an NLG problem, we want to build a model of the probability that the solution is representative of human behavior in our transcripts. We train a maximum entropy model (Berger et al., 1996) to make the prediction, using the MALLET software package (McCallum, 2002). Given that the generator ultimately wants to choose the best utterance, we could explore approaches to learn rankings directly, such as RankBoost (Freund et al., 2003).

Formally, the machine learning model characterizes an input–output pair for NLG with a set of features that would be available to a generator in assessing a candidate output. Each training example pairs an inventory of features with an observed value indicating whether the instance does or does not match the utterance produced by the human user. Given a training set, MALLET selects a set

[1]Our specific approach was to capture all the successful utterances that differ from the preferred NLG path by any three derivation steps of the lexicalized generation grammar. This heuristic was easy to implement with COREF's existing infrastructure for look-ahead search, and we found empirically that more comprehensive search was expensive to carry out and tended primarily to add unnaturally verbose and redundant utterances. See the Appendix for examples.

of features to use and fits numerical weights for the features for logistic regression by maximum entropy. That is, the features determine the predicted probability that candidate output j for problem t (utterance $u_{t,j}$) is good (a match with a hypothetical user utterance), as a logistic function of the sum of the feature weights describing the instance—formally,

$$P(u_{t,j} = Good \,|\, features(u_{t,j})) = \\ 1/(1 + \exp(-w_0 - \textstyle\sum_i features(u_{t,j})_i * w_i))$$

This model can then be applied to unlabeled instances with features derived from novel NLG problem instances and candidate outputs.

The features we use in our experiments are described in full in Tables 4 and 5 in the Appendix. Most are from DeVault and Stone (2009). We have features describing the form of the output utterance: what phrase structure it has and what lexical items are used. We have features describing what task moves are achieved by the utterance and what links the utterance has to context. For completeness, we also add DeVault and Stone's features describing the context itself, including the conversational tasks underway, the facts on the conversational record, and the properties relevant to ongoing problem solving.[2]

In designing features for learning, we also draw on the experience of Jordan and Walker (2005) in predicting the form of referring expressions. Many of their features closely align with those we inherit from DeVault and Stone (2009). One kind that doesn't is Jordan and Walker's conceptual pacts feature set. These features are designed to capture utterance choices that are contingent on other participants' previous choices in interaction—entrainment (Brennan and Clark, 1996). We make it possible for the learner to detect entrainment by introducing a new set of *history features*, which list the presuppositions of recent utterances.

We do not need Jordan and Walker's distractor features, however. Unlike them, we do not try to learn the difference between distinguishing descriptions and ambiguous ones. Our architecture, like that of Garoufi and Koller (2011), doesn't even consider a candidate utterance unless it's unambiguous on a standard reference model (Dale and Reiter, 1995). Garoufi and Koller (2011) provide evidence for the effectiveness of this kind of factorization of modeling and learning.

[2]If these context features were shared across all outputs for a given input, they would not affect what option for NLG was best. But this is not always the case in COREF, because contexts can be uncertain and because COREF can trigger accommodation that changes the context as part of NLG. Moreover, including these features might allow us to capture possible variability in NLG, since the model can then predict that otherwise marked utterances work naturally in some contexts.

4.3 Assessing the Model

To use the trained model, we start from the NLG problem of generating an utterance to achieve specified communicative goals in context. Our NLG model constructs its space of candidate utterances. Each candidate input–output pair is analyzed in terms of its features, and then the learned model assigns it a probability score. We pick our output via the candidate with the highest score.

In evaluating how well this works, we are interested in how well the learned model predicts the utterances of new subjects given data from other subjects. We assess this by reporting cross-validation results, predicting the choices of one, held-out subject given a model trained on the data from all other users in an experiment. We report an exact match error measure. In a more complex generation task, we could measure error based on edit distance to give partial credit to NLG results that are closer to user utterances. As a baseline, we report comparable measures for COREF's original NLG implementation.

5 Pilot: The Need for Reciprocity

We applied our NLG training methodology to the data set reported by DeVault and Stone (2009) with 20 subjects interacting with COREF. The results were not compelling.

Analysis of this data set transforms human subjects' utterances into 889 problem instances for NLG. In 247 of these instances, the user's utterance is not in the NLG search space, usually because it is interpreted by robust methods rather than COREF's grammar. Of the remaining 642 utterances, our baseline generator already matches the user utterance 308 times (48%); it differs on the other 334 instances (52%). After learning, a model-based generator trained on the other 19 users' data now matches the utterance of a held-out user on 546 instances (85%) across cross-validation runs. This sounds promising, but in fact almost all of the model successes (534 instances) are due to just five utterance types that fulfill simple dialogue-management functions: *'yes'*, *'no'*, *'click continue'*, *'done'* and *'ok'*.

There is in fact quite little evidence in this data about how COREF should make its typical generation decisions. Looking under the hood, the problem is that COREF's dialogue management policy did not exploit the symmetry and reciprocity of its dialogue models and NL representations. COREF took the initiative in object-identification dialogues when it was the director, offering descriptions of the target object, but it also took the initiative when it was the matcher, asking the user to confirm or reject its suggestions about the identity and properties of the target objects.

System builders often make such design choices to foster task success. Giving the system the initiative generally means that user utterances are understood more reliably, which helps keep the dialogue on track. However, in settings where the system can potentially improve its behavior, we may have to design the system to take more risks so it can acquire the data it needs; we may even want to sacrifice short-term task success to enable long-term improvement. Such trade-offs of exploration and exploitation are endemic in reinforcement learning, but learning by imitation gives the problem a distinctively social dimension: getting the right data may mean not only trying new actions in new situations, but actively creating the right relationship with the user.

6 Collecting Mixed-initiative Data

We revised COREF's dialogue strategy to better reflect users' interactive competence using simple statistics about dialogue outcomes. For each class of dialogue move by the agent in DeVault and Stone's evaluation data, we tabulated the number of subsequent utterances required to identify the object. These measures give COREF's planned utterance an empirical score quantifying its anticipated effect in dialogue. For example, after asking if a particular object was the target, the subdialogue finished in 6.0 more turns on average. Analogous measures give a comparable score to the most effective kind of contribution that's potentially available to the user at each point in the dialogue. For example, after saying that a particular object was the target, the subdialogue finished in 3.2 more turns on average. Our new dialogue policy compares COREF's planned move with the user's best option. COREF proceeds with its utterance if its score is better but waits for the user if its score is worse. This analysis gives our revised version of COREF an empirical threshold for taking initiative in the dialogue based on the strengths of the contributions COREF and the user could make next in context. In practice, the revised strategy lets user directors drive the dialogue much more often than DeVault and Stone's original handcrafted policy. For example, COREF now waits for the user to propose a description rather than asking about a candidate object.

We had 42 subjects interact with the revised COREF in a protocol of 29 object identification tasks, grouped in blocks of 4, 9 and 16 as in DeVault and Stone (2009). Subjects were recruited by advertisement and word of mouth from our institution and were paid for their participation. The data was collected as part of an independently-motivated assessment of COREF's trade-offs between asking for clarification and proceeding under uncertainty with its best interpretation, so COREF varied these choices across the dialogues.

Analysis of our new data set induces 2006 NLG problem instances corresponding to human utterances, including 1382 cases where the user's utterance is (1) completely described by COREF's grammar, (2) found in the NLG search space, and (3) represented as unambiguous by the underlying NLG model. To confirm the diversity of utterances in this set, we automatically partitioned the utterances into four classes based on surface form and communicative goals achieved: acknowledgments that coordinate on the current state of the dialogue (569 instances), task instructions (23 instances), yes/no answers (434 instances) and other dialogue contributions with explicit descriptive content (356 instances). Thus, this data set contains substantial evidence about human strategies in COREF's domain. We continue to perform analyses of utterances by category to document the results of our learning experiment.

7 Results

Table 1 compares the aggregate performance of the learned NLG module in comparison to COREF's baseline generator across all cross-validation runs (training on 41 users and testing on data from one held-out user). Except in the small category of task instructions, where the baseline is already good, the learned model offers a substantial improvement in rate of exact match to user utterance across all categories. These differences in

Table 1: Comparison of learned model and baseline generator.

System	Descriptive	Acknowledgments	Yes/No	Instructions	Total
Baseline	$\frac{170}{356} = 47.8\%$	$\frac{349}{569} = 61.3\%$	$\frac{210}{434} = 48.4\%$	$\frac{23}{23} = 100\%$	$\frac{752}{1382} = 54.4\%$
Model	$\frac{259}{356} = 72.8\%$	$\frac{477}{569} = 83.8\%$	$\frac{427}{434} = 98.4\%$	$\frac{23}{23} = 100\%$	$\frac{1186}{1382} = 85.8\%$

Evaluation of exact match to user utterances across hold-one-user-out cross-validation runs. We report number of matching instances out of number of instances with the user utterance in the NLG search space, along with percentage match, broken down by form and communicative goal of the utterance.

Table 2: Comparison of accuracy by item.

		Baseline	
		Match	Mismatch
Model	Match	720	466
	Mismatch	32	164

(a) Counts of NLG problem instances of all types, comparing matches in the baseline generator against matches in the learned model.

		Baseline	
		Match	Mismatch
Model	Match	152	107
	Mismatch	18	79

(b) Counts of NLG problem instances with substantive contributions and explicit descriptive material, comparing matches in the baseline generator against matches in the learned model.

rates are all statistically significant ($p < .005$ by Fisher's exact test).

Table 2 breaks down overall results (Table 2a) and results on descriptive utterances (Table 2b), to explore associations between the performance of the baseline generator and the performance of the learned model on individual items. We find a clear link between the two methods: when the model gets an utterance wrong, the baseline method is much more likely to have gotten the utterance wrong as well ($p < .001$ by Fisher's exact test). We conclude that the model is not just improving on the baseline generator in aggregate, but has learned to correct specific choices in the baseline system that are not representative of user behavior.

The breakdown in Table 1 gives a sense of the range of cases covered by the learned model. The 'yes/no' cases mostly involve training COREF to say *'yes'* rather than *'yeah'*. The acknowledgment cases involve understanding the subtle ways that people trade off alternatives such as *'ok'*, *'done'* and *'I added it'*—a difficult problem but one where we have little choice but to trust machine learning results.

Descriptive utterances are more substantial. To understand these cases better, we built an overall model with data from all 42 users and looked at the features selected by MALLET and the weights fit for them in the maximum entropy model. Table 3 shows a sample of the MALLET output. We think of these features as establishing a network of prioritized defaults; lower-weighted features must conspire together to override higher-weighted ones. Syntax is the strongest effect; for example, the contrast between $[_S$ DET N$]$ and $[_S$ NP IS DET N$]$ gives a preference of 1.27 to the simpler structure. Lexical features encode more natural items (*'brown'* versus *'beige'*) but also implicitly encode natural descriptive patterns (as with the color modifier *'light'*). Presupposition features, meanwhile, help ensure that words have their most natural meanings. On this analysis, the model contents corroborate our hypothesis that user data gives evidence to refine a wide variety of NLG choices.

8 Discussion

In this paper, we show how users' utterances can give a dialogue system consistent and reliable indicators not only of how to solve its NLU problems, as in DeVault and Stone (2009), but also how to solve its NLG problems. Thus, we can now design dialogue systems to learn to imitate their human users in certain cases. To do so, the system needs to work in a domain where users are prepared to offer the same kind of contributions

Table 3: Sample features used to identify user tuples and their weights in an overall model.

Syntax Features:	
Fits [$_S$ DET N]	2.29
Fits [$_S$ COLOR N]	2.09
Fits [$_S$ DET COLOR N]	1.86
Fits [$_S$ NP IS DET N]	1.12
Lexical Features:	
Includes word *light*	0.87
Includes word *dark*	0.60
Includes word *brown*	0.22
Includes word *beige*	0.005
Presupposition Features:	
Uses *square* for square object	2.05
Uses *diamond* for rhombus	2.09
Uses *pink* for pale red-purple	1.70
Describes light blue as *light*	0.92

as the system, the system needs to represent those contributions symmetrically, and the system needs to be able to actually elicit, analyze and learn from relevant user utterances.

Our approach, like that of Garoufi and Koller (2011), is to combine a symbolic account of utterance interpretation with a learned model of utterance quality. Thus, on our approach, system utterances always come with formal guarantees that they fulfill specified communicative goals and have a unique interpretation in context. That may help underwrite the guarantees that Paek and Pieraccini (2008) emphasize, that data-driven systems must respect the coherence of dialogue and must continue to do so even as they learn to improve dialogue efficiency and naturalness.

Our work suggests some natural followups. It would be interesting to refine the NLG model based on the disambiguation strategy learned in DeVault and Stone (2009). If the system discovers that utterances are not as ambiguous as the initial model suggests, it opens up new possibilities for tuning NLG to match what users say. Scaling up the ideas, meanwhile, invites us to build factored models that describe NLG decisions in a more compositional way, as well as finding more powerful and generalizable features.

Further work is also required to use these techniques in a broader range of settings. Our technique requires the system to give users the opportunity to say the same kinds of things it says, so it is most appropriate for collaborative problem solving. Further research is required to use the methodology for asymmetric situations such as information seeking. Use in spoken dialogue systems, meanwhile, would challenge the limits of mixed-initiative interaction and would require techniques to discount users' errors and disfluencies. Although these limitations make our techniques difficult to use in many current applications, we are optimistic that our methods will apply quite naturally to emerging open-domain settings such as human–robot interaction, where users and systems meet on a more equal footing.

Acknowledgments

This authors were supported by NSF DGE 0549115 (IGERT) and IIS 1017811. Thanks to the reviewers, and to David DeVault and Lara Martin for discussion and assistance.

References

Anja Belz and Albert Gatt. 2008. Intrinsic vs. extrinsic evaluation measures for referring expression generation. In *Proceedings of ACL*, pages 197–200.

Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Cynthia Breazeal, Daphna Buchsbaum, Jesse Gray, and Bruce Blumberg. 2005. Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots. *Artificial Life*, 11(1–2):1–32.

Susan E. Brennan and Herbert H. Clark. 1996. Conceptual pacts and lexical choice in conversation. *J. Experimental Psychology*, 22(6):1482–1493.

Herbert H. Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1–39.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 18:233–263.

David DeVault and Matthew Stone. 2009. Learning to interpret utterances using dialogue history. In *Proceedings of EACL*, pages 184–192.

David DeVault, David Traum, and Ron Artstein. 2008. Practical grammar-based NLG from examples. In *Proceedings of INLG*, pages 78–85.

David DeVault. 2008. *Contribution Tracking: Participating in Task-Oriented Dialogue under Uncertainty*. Ph.D. thesis, Rutgers.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *J. Machine Learning Research*, 4:933–969.

Konstantina Garoufi and Alexander Koller. 2011. Combining symbolic and corpus-based approaches for the generation of successful referring expressions. In *Proceedings of EWNLG*, pages 121–131.

Ryuichiro Higashinaka, Rashmi Prasad, and Marilyn A. Walker. 2006. Learning to generate naturalistic utterances using reviews in spoken dialogue systems. In *Proceedings of ICCL–ACL*, pages 265–272.

Srinivasan Janarthanam, Helen Hastie, Oliver Lemon, and Xingkun Liu. 2011. "The day after the day after tomorrow?" a machine learning approach to adaptive temporal expression generation: training and evaluation with real users. In *Proceedings of SIGDIAL*, pages 142–151.

Pamela W. Jordan and Marilyn A. Walker. 2005. Learning content selection rules for generating object descriptions in dialogue. *J. Artif. Intell. Res. (JAIR)*, 24:157–194.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, pages 69–123. Springer.

Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of COLING–ACL*, pages 704–710.

Oliver Lemon. 2011. Learning what to say and how to say it: Joint optimisation of spoken dialogue management and natural language generation. *Computer Speech & Language*, 25(2):210–221.

Andrew McCallum. 2002. MALLET: A MAchine learning for LanguagE toolkit. http://mallet.cs.umass.edu.

Tim Paek and Roberto Pieraccini. 2008. Automating spoken dialogue management design using machine learning: An industry perspective. *Speech Communication*, 50(8–9):716–729.

Verena Rieser and Oliver Lemon. 2011. *Reinforcement Learning for Adaptive Dialogue Systems: A Data-driven Methodology for Dialogue Management and Natural Language Generation*. Springer.

Advaith Siddharthan and Ann A. Copestake. 2004. Generating referring expressions in open domains. In *Proceedings of ACL*, pages 407–414.

Amanda Stent, Rashmi Prasad, and Marilyn A. Walker. 2004. Trainable sentence planning for complex information presentations in spoken dialog systems. In *Proceedings of ACL*, pages 79–86.

Matthew Stone, Christine Doran, Bonnie Webber, Tonia Bleam, and Martha Palmer. 2003. Microplanning with communicative intentions: the SPUD system. *Computational Intelligence*, 19(4):314–381.

Marilyn A. Walker, Owen Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech & Language*, 16(3–4):409–433.

Martin Zinkevich, Michael H. Bowling, and Michael Wunder. 2011. The lemonade stand game competition: solving unsolvable games. *SIGecom Exchanges*, 10(1):35–38.

Appendix: NLG Search and Features

User utterance	*pink square*
Goal(s) found	1. Target is pink 2. Target is square, *or* 3. Target is both pink and square
Baseline	1. the target is pink 2. the target is square 3. pink square
Model	1. pink square 2. square 3. pink square
Candidates	a box, a fuschia box, a fuschia fuschia box, a fuschia fuschia square, a fuschia pink box, a fuschia pink square, a fuschia purple box, a fuschia purple square, a fuschia square, a like fuschia box, a like fuschia square, a like pink box, a like pink square, a like purple box, a like purple square, a pink box, a pink fuschia box, a pink fuschia square, a pink pink box, a pink pink square, a pink purple box, a pink purple square, a pink square, a purple box, a purple fuschia box, a purple fuschia square, a purple pink box, a purple pink square, a purple purple box, a purple purple square, a purple square, a square, box, fuschia box, fuschia square, pink box, pink square, purple box, purple square, square, the target is fuschia, the target is pink, the target is purple, the target is square

Model confirms baseline vocabulary, learns to overspecify color goal (1) for more natural syntax. COREF can't spell 'fuchsia'.

Table 4: Features derived from the current state of the dialogue (s_t).

feature set	description
NumTasksUnderway	The number of tasks underway in the state s_t.
TasksUnderway	For any task that is underway in state s_t, a feature includes its name, its depth on the task stack, and its current status in its formal task network.
NumRemainingReferents	The number of targets that remain to be identified in state s_t.
TabulatedFacts	For any fact on the conversational record at state s_t there is a corresponding string feature—a formula with any unique reference symbols anonymized (e.g. *X34* becomes *some-object*).
CurrentTargetConstraints	For any positive or negative constraint on the current target in state s_t, there is a corresponding string feature.
UsefulProperties	For any property instantiated in the display in state s_t there is a corresponding feature.
History	Each assertion and presupposition on the conversational record in state s_t is represented as a string feature.

Table 5: Features derived from the proposed utterance ($u_{t,j}$).

feature set	description
Presuppositions	Each of the atomic presuppositions of the utterance $u_{t,j}$ is represented as a string feature. The string captures predicate–argument structure but anonymizes references to individuals (e.g. *target12* becomes *sometarget*).
Assertions	Each of the dialogue moves that the utterance contributes corresponds to a feature. This string also captures predicate–argument structure but anonymizes references to individuals.
Syntax	A string representation of the bracketed phrase structure, including non-terminal categories, of the utterance.
Words	We represent each word that occurs in the utterance as a feature.

User utterance	*the light blue diamond*
Goal(s) found	Target is specified object
Baseline	the blue object
Model	the light blue diamond
Candidates	the blue blue diamond, the blue blue object, the blue blue rhombus, the blue diamond, the blue diamond outline, the blue object, the blue object outline, the blue rhombus, the blue rhombus outline, the empty blue diamond, the empty blue object, the empty blue rhombus, the hollow blue diamond, the hollow blue object, the hollow blue rhombus, (continued)
Candidates	the light blue diamond, the light blue object, the light blue rhombus, the lighter blue diamond, the lighter blue object, the lighter blue rhombus, the like blue diamond, the like blue object, the like blue rhombus, the outline blue diamond, the outline blue object, the outline blue rhombus, the sky blue diamond, the sky blue object, the sky blue rhombus

Model confirms baseline pattern of color and type reference but learns to overspecify color as *light blue* and to use basic type *diamond*.