Learning to Sportscast: A Test of Grounded Language Acquisition

David L. Chen   DLCC@CS.UTEXAS.EDU
Raymond J. Mooney   MOONEY@CS.UTEXAS.EDU
Dept. of Computer Sciences, The University of Texas at Austin, 1 University Station C0500, Austin TX 78712, USA

Abstract

We present a novel commentator system that learns language from sportscasts of simulated soccer games. The system learns to parse and generate commentaries without any engineered knowledge about the English language. Training is done using only ambiguous supervision in the form of textual human commentaries and simulation states of the soccer games. The system simultaneously tries to establish correspondences between the commentaries and the simulation states as well as build a translation model. We also present a novel algorithm, Iterative Generation Strategy Learning (IGSL), for deciding which events to comment on. Human evaluations of the generated commentaries indicate they are of reasonable quality compared to human commentaries.

Figure 1. Screenshot of our commentator system

1. Introduction

Children acquire language through exposure to linguistic input in the context of a rich, relevant, perceptual environment. By connecting words and phrases to objects and events in the world, the semantics of language is grounded in perceptual experience (Harnad, 1990). Ideally, a machine learning system would be able to acquire language in a similar manner without human supervision. As a step in this direction, we present a commentator system that can describe events in a simulated soccer game by learning from sample human commentaries paired with the simulation states. A screenshot of our system with generated commentaries is shown in Figure 1.

Although there has been some interesting computational work in grounded language learning (Roy, 2002; Bailey et al., 1997; Yu & Ballard, 2004), most of the focus has been on dealing with raw perceptual data and the complexity of the language involved has been very modest.

To help make progress, we study the problem in a simulated environment that retains many of the important properties of a dynamic world with multiple agents and actions while avoiding many of the complexities of robotics and vision. Specifically, we use the Robocup simulator (Chen et al., 2003) which provides a fairly detailed physical simulation of robot soccer. While several groups have constructed Robocup commentator systems (André et al., 2000) that provide a textual natural-language (NL) transcript of the simulated game, their systems use manually-developed templates and are incapable of learning.

Our commentator learns to semantically interpret and generate language in the Robocup soccer domain by observing an on-going commentary of the game paired with the dynamic simulator state. By exploiting existing techniques for abstracting a symbolic description of the activity on the field from the detailed state of the physical simulator (André et al., 2000), we obtain a pairing of natural language with a symbolic description of the perceptual context in which it was uttered. However, such training data is highly ambiguous because each comment usually co-occurs with several events in the game. We integrate and enhance existing methods for learning semantic parsers and NL gen-

erators (Kate & Mooney, 2007; Wong & Mooney, 2007a) in order to learn to understand and produce grounded language from such ambiguous training data.

2. Background

Systems for learning semantic parsers induce a function that maps NL sentences to *meaning representations* (MRs) in some formal logical language. Existing work has focused on learning from a supervised corpus in which each sentence is manually annotated with its correct MR (Mooney, 2007). Such human annotated corpora are expensive and difficult to produce, limiting the utility of this approach. The systems described below assume they have access to a formal context-free grammar, called the *meaning representation grammar* (MRG), that defines the MR language (MRL).

2.1. KRISP and KRISPER

KRISP (Kate & Mooney, 2006) uses SVMs with string kernels (Lodhi et al., 2002) to learn semantic parsers. For each production in the MRG, the system learns an SVM string classifier that recognizes the associated NL words or phrases. The resulting suite of classifiers is then used to construct the most probable MR for a complete NL sentence. Given the partial matching provided by string kernels and the over-fitting prevention provided by SVMs, KRISP has been experimentally shown to be robust to noisy training data.

KRISPER (Kate & Mooney, 2007) is an extension to KRISP that handles ambiguous training data, in which each sentence is annotated only with a *set* of potential MRs, only one of which is correct. It employs an iterative approach analogous to EM that improves upon the selection of the correct NL–MR pairs in each iteration. In the first iteration, it assumes that all of the MRs paired with a sentence are correct and trains KRISP with the resulting noisy supervision. In subsequent iterations, KRISPER uses the currently trained parser to score each potential NL–MR pair, selects the most likely MR for each sentence, and retrains the parser. In this manner, KRISPER is able to learn from the type of weak supervision expected for a grounded language learner exposed only to sentences in ambiguous contexts. However, the system has previously only been tested on artificially corrupted or generated data.

2.2. WASP

WASP learns semantic parsers using *statistical machine translation* (SMT) techniques (we use the Wong & Mooney (2007b) version). It induces a *probabilistic synchronous context-free grammar* (PSCFG) (Wu, 1997) to translate NL sentences in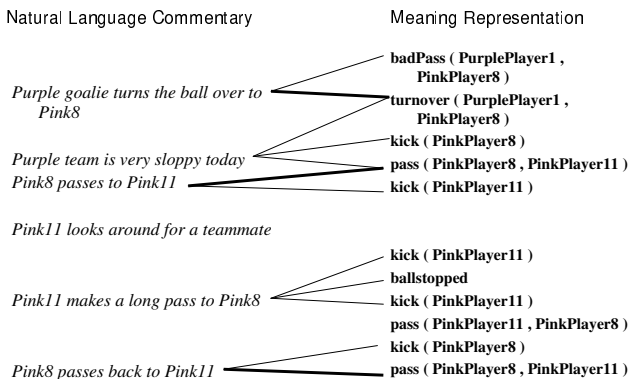to logical MRs using a modification of recent methods in syntax-based SMT (Chiang, 2005). Since a PSCFG is symmetric with respect to input/output, the same learned model can also be used to *generate* NL sentences from formal MRs. Thus, WASP learns a PSCFG that supports both semantic parsing and natural language generation. Since it does not have a formal grammar for the NL, the generator also learns an n-gram language model for the NL and uses it to choose the overall most probable NL translation of a given MR using a noisy-channel model (Wong & Mooney, 2007a).

Figure 2. Sample trace of ambiguous training data

3. Sportscasting Data

To train and test our system, we assembled human-commentated soccer games from the Robocup simulation league (www.robocup.org). Since our focus is language learning not computer vision, we chose to use simulated games instead of real game video to simplify the extraction of perceptual information. Symbolic representations of game events were automatically extracted from the simulator traces by a rule-based system. The extracted events mainly involve actions with the ball, such as kicking and passing, but also include other game information such as whether the current playmode is kickoff, offside, or corner kick. The events are represented as atomic formulas in predicate logic with timestamps. These logical facts constitute the requisite MRs, and we manually developed a simple MRG for this formal semantic language.

For the NL portion of the data, we had humans commentate games while watching them on the simulator. The commentators typed their comments into a text box, which were recorded with a timestamp. To construct the final ambiguous training data, we paired each comment with all of the events that occurred five seconds or less before the comment was made. A sample set of ambiguous training data is shown in Figure 2. Note that the use of English words for predicates and constants in the MR is for human read-

	Number of events	Number of comments			Events per comment		
		Total	Have MRs	Have Correct MR	Max	Average	Std. Dev.
2001 final	3992	722	671	520	9	2.235	1.641
2002 final	2125	514	458	376	10	2.403	1.653
2003 final	2112	410	397	320	12	2.849	2.051
2004 final	2223	390	342	323	9	2.729	1.697

Table 1. Statistics about the dataset

ability only, the system treats these as arbitrary conceptual tokens and must learn their connection to English words.

We annotated a total of four games, namely, the finals for the Robocup simulation league for each year from 2001 to 2004. Summary statistics about the data are shown in Table 1. The 2001 final has almost twice the number of events as the other games because it went into double overtime. For evaluation purposes only, a gold-standard matching was produced by examining each comment manually and selecting the correct MR if it exists. The bold lines in Figure 2 indicate the correct matches. Notice some sentences do not have correct matches (about one fifth of our data). For example, the sentence "Purple team is very sloppy today" cannot be represented in our MRL and consequently does not have a corresponding correct MR. On the other hand, in the case of the sentence "Pink11 makes a long pass to Pink8", the correct MR falls outside the 5-second window. For each game, Table 1 shows the total number of NL sentences, the number of these that have at least one recent extracted event to which it *could* refer, and the number of these that actually *do* refer to one of these recent extracted events. The maximum, average, and standard deviation for the number of recent events paired with each comment is also given.

4. New Algorithms

While existing systems are capable of solving parts of the sportscasting problem, none of them are able to perform the whole task on their own. We introduce three new end-to-end systems below which are able to learn from the ambiguous supervision in our training data and generate commentaries on unseen games.

4.1. WASPER

Since our primary goal is to learn a sportscaster rather than a parser, we use WASP to learn a system that can also generate NL from MRs produced by the perceptual system. However, WASP requires unambiguous training data which is not available for our domain. Therefore, we extend WASP using EM-like retraining similar to KRISPER to handle ambiguously annotated data, resulting in a system we call WASPER. In general, any system that learns semantic parsers can be extended to handle ambiguous data

as long as it can produce confidence levels for given NL–MR pairs.

4.2. KRISPER-WASP

KRISP has been shown to be superior to WASP at handling noisy training data (Kate & Mooney, 2006). Consequently, we can expect KRISPER's parser to outperform WASPER's because EM-like training on ambiguous data initially creates a lot of noisy, incorrect supervision. Even if the average number of possible MRs per sentence is only 2, it still results in at least 50% noise in the training data in the first iteration. However, KRISPER cannot learn a language generator, which is necessary for our sportscasting task. As a result, we create a new system called KRISPER-WASP that is both good at disambiguating the training data and capable of generation. We first use KRISPER to train on the ambiguous data and produce a disambiguated training set by using its prediction for the most likely MR for each sentence. This unambiguous training set is then used to train WASP to produce both a parser and a generator.

4.3. WASPER-GEN

In both KRISPER and WASPER, the criterion for selecting the best NL–MR pairs during retraining is based on maximizing the probability of parsing a sentence into a particular MR. However, since WASPER is capable of both parsing and generation, we could alternatively select the best NL–MR pairs by evaluating how likely it is to *generate* the sentence from a particular MR. Thus, we built another version of WASPER (WASPER-GEN) that disambiguates the training data in order to maximize the performance of *generation* rather than parsing. It uses a generation-based score rather than a parsing-based score to select the best NL–MR pairs. Specifically, an NL–MR pair (n, m) is scored by using the current trained generator to generate an NL sentence for m and then comparing the generated sentence to n to compute the NIST score. NIST score is a machine translation (MT) metric that measures the precision of a translation in terms of the proportion of n-grams it shares with a human translation (Doddington, 2002). It is also used to evaluate NL generation. Another popular MT metric is BLEU score (Papineni et al., 2002) but we found it inadequate for our domain because it overly penalizes

translations shorter than the target sentences. Most of our generated commentaries are shorter than the human commentaries due to the fact that humans are more verbose and many details of the human descriptions are not represented by our MRL.

4.4. Learning for Strategic Generation

A language generator alone is not enough to produce a sportscast. In addition to knowing *how* to say something, one must also know *what* to say. A sportscaster must also choose which events to describe. In NLP, deciding what to say is called *strategic generation*.

We developed a simple method for learning which events to describe. For each event type (i.e. for each predicate like `pass`, or `goal`), the system uses the training data to estimate a probability that it is mentioned by the sportscaster. Given the gold-standard NL–MR matches, this probability is easy to estimate; however, the learner does not know the correct matching. Instead, the system must estimate the probabilities from the ambiguous training data. We compare two basic methods for estimating these probabilities.

The first method uses the *inferred* NL–MR matching produced by the language-learning system. The probability of commenting on each event type, E_i, is estimated as the percentage of events of type E_i that have been matched to *some* NL sentence.

The second method, which we call Iterative Generation Strategy Learning (IGSL), uses a variant of EM, treating the matching assignments as hidden variables, initializing each match with a prior probability, and iterating to improve the probability estimates of commenting on each event type. Unlike the first method, IGSL uses MRs not associated with any sentences explicitly in training. Algorithm 1 shows the pseudocode. Each sentence accounts for at most one occurrence of an event being commented (some comments do not correspond to any MRs), so we enforce that the counts associated with a sentence add up to exactly one. In the initial iteration, every possible match gets assigned a weight inversely proportional to its amount of ambiguity. Thus, a sentence associated with five possible MRs will assign each match a weight of $\frac{1}{5}$. In the subsequent iterations, we use the learned estimates for each event type to assign weights to the edges, again normalizing to make sure that the weights of the edges coming out of each sentence sum to one.

To generate a sportscast, we first use the learned probabilities to determine which events to describe. For each time step, we only consider commenting on the event with the highest probability. The system then generates a comment for this event stochastically based on the estimated probability for its event type.

Algorithm 1 Iterative Generation Strategy Learning

input event types $E = \{E_1, ..., E_i, ..., E_n\}$, the number of occurrences of each event type $totalCount(E_i)$, sentences S and their associated sets of meaning representations $MR(s)$,

output probabilities of commenting on each event type $Pr(E_i)$

  for event type $E_i \in E$ **do**
    Initialize $count = 0$
    for sentence $s \in S$ and $E_i \in MR(s)$ **do**
      $count = count + \frac{1}{|(MR(s))|}$
    end for
    $Pr(E_i) = \frac{count}{totalCount(E_i)}$
  end for

  repeat
    for event type $E_i \in E$ **do**
      Initialize $count = 0$
      for sentence $s \in S$ and $E_i \in MR(s)$ **do**
        $totalProb = 0$
        for event $E_j \in MR(s)$ **do**
          $totalProb = totalProb + Pr(E_j)$
        end for
        $count = count + \frac{Pr(E_i)}{totalProb}$
      end for
      $Pr(E_i) = \frac{count}{totalCount(E_i)}$
    end for
  until Convergence or MAX_ITER reached

---

5. Experimental Evaluation

This section presents experimental results on the Robocup data for four systems: KRISPER, WASPER, KRISPER-WASP, and WASPER-GEN. To better gauge the effect of accurate ambiguity resolution, we also include results of unmodified WASP. Since WASP requires unambiguous training data, we randomly pick a meaning for each sentence from its set of potential MRs. Finally, we also include the result of WASP trained using *gold matching* which consists of the correct NL–MR pairs annotated by a human. This represents an upper-bound on what our systems could achieve if they disambiguated the training data perfectly.

We evaluate each system on three tasks: matching, parsing, and generation. The matching task measures how well the systems can disambiguate the training data. The parsing and generation tasks measure how well the systems can translate from NL to MR, and from MR to NL, respectively.

Since there are four games in total, we trained using all possible combinations of one to three games, and in each case, tested on the games not used for training. Results were averaged over all train/test combinations. We evalu-
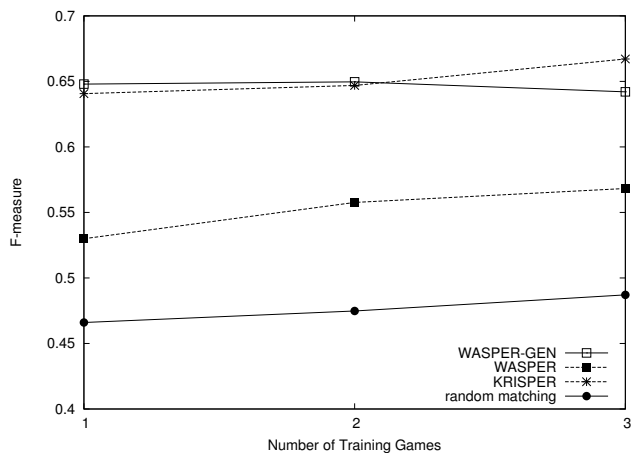
Figure 3. Matching Results

Figure 4. Semantic Parsing Results

Figure 5. Generation results

ated matching and parsing using F-measure, the harmonic mean of recall and precision. Precision is the fraction of the system's annotations that are correct. Recall is the fraction of the annotations from the gold-standard that the system correctly produces. Generation is evaluated using NIST scores which roughly estimates how well the produced sentences match with the target sentences.

5.1. Matching NL and MR

Since handling ambiguous training data is an important aspect of grounded language learning, we first evaluate how well the various systems pick the correct NL–MR pairs. Figure 3 shows the F-measure for identifying the correct set of pairs for the various systems. WASPER does better than random matching, but worse than the other two systems. While we expected KRISPER to perform better since it is more adept at handling noisy data, it is somewhat surprising that WASPER-GEN does about the same. A potential explanation is that WASPER-GEN avoids making certain systematic errors typical of the other systems. This is discussed further in section 5.3.

5.2. Semantic Parsing

Next, we present results on the accuracy of the learned semantic parsers. Each trained system is used to parse and produce an MR for each sentence in the test set that has a correct MR in the gold-standard matching. A parse is considered correct if and only if it matches the gold standard exactly. Parsing is a fairly difficult task because there is usually more than one way to describe the same event. For example, "Player1 passes to player2" can refer to the same event as "Player1 kicks to player2." Thus, accurate parsing requires learning all the different ways people describe an event. Synonymy is not limited to verbs. In our data, "Pink1", "PinkG" and "pink goalie" all refer to player1 on
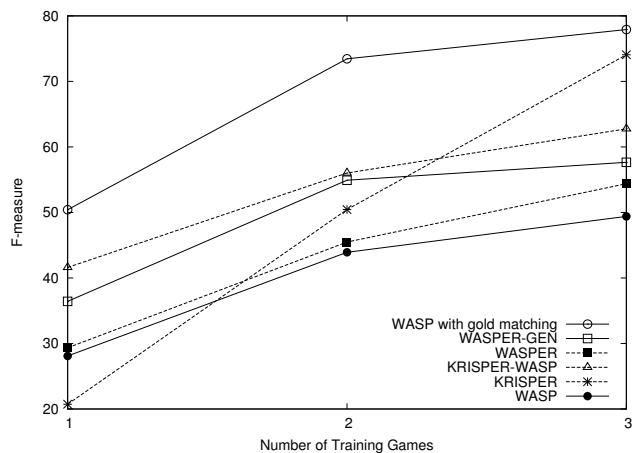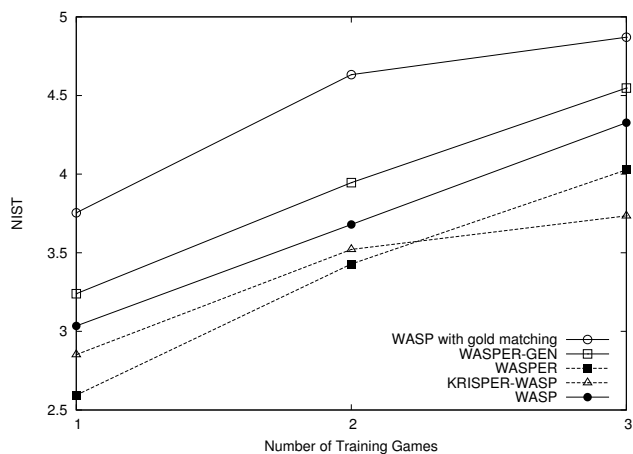
the pink team. Since we are not providing the systems with any prior knowledge, parsers have to learn all these different ways of referring to the same entity.

Results are shown in Figure 4, and, as expected, follow the matching results. Systems that did better at disambiguating the training data also did better on parsing since their supervised training data is less noisy. When trained on 3 games, KRISPER does the best since it is most effective at handling the noise in the final supervised data. However, it tends to do worse than the other systems when given less training data.

5.3. Generation

The third evaluation task is generation. All of the WASP-based systems are given each MR in the test set that has a gold-standard matching NL sentence and asked to generate an NL description. The quality of the generated sentence is measured by comparing it to the gold-standard using NIST scoring.

This task is easier than parsing because the system only needs to learn *one way* to accurately describe an event. This property is reflected in the results, shown in Figure 5, where even the baseline system WASP does fairly well, outperforming WASPER and KRISPER-WASP. As the number of event types is fairly small, only a relatively small number of correct matchings is required to perform this task well as long as each event type is associated with a correct sentence pattern more often than any other sentence patterns. Consequently, it is far more costly to make systematic errors as is the case for WASPER and KRISPER-WASP.

Even though systems such as WASPER and KRISPER-WASP do fairly well at disambiguating the training data, the mistakes they make in selecting the NL–MR pairs often repeat the same basic error. For example, a bad pass event is often followed by a turnover event. If initially the system incorrectly determines that the comment "Player1 turns the ball over to the other team" refers to a bad pass, it will parse the sentence "Player2 turns the ball over to the other team" as a bad pass as well since it just reinforced that connection. Even if the system trains on a correct example where a bad pass is paired with the linguistic input "Player1 made a bad pass", it does not affect the parsing of the first two sentences and does not correct the mistakes. As a result, a bad pass becomes incorrectly associated with the sentence pattern "Someone turns the ball over to the other team."

On the other hand, WASPER-GEN does the best due to the imbalance between the variability of natural language comments and the MRs. While the same MR will typically occur many times in a game, the exact same comments are almost never uttered again. This leads to two performance advantages for WASPER-GEN.

WASPER-GEN avoids making the same kind of systematic mistakes as WASPER and KRISPER-WASP. Following the previous example, when WASPER-GEN encounters the correct matching for bad pass, it learns to associate bad passes with the correct sentence pattern. When it goes back to those first two incorrect pairings, it will likely correct its mistakes. This is because the same MR bad pass is present in all three examples. Thus, it will slowly move away from the incorrect connections. Of course, parsing and generation are symmetrical processes, so using generation to disambiguate data has its own problems. Namely, it is possible to converge to a point where many events generates the same natural language description. However, since there is much more variability in natural language, it is very unlikely that the same sentence pattern will occur repeatedly, each time associated with different events.

Another performance advantage of WASPER-GEN can be found by looking at the objective differences. Systems such as WASPER and KRISPER-WASP which use parsing
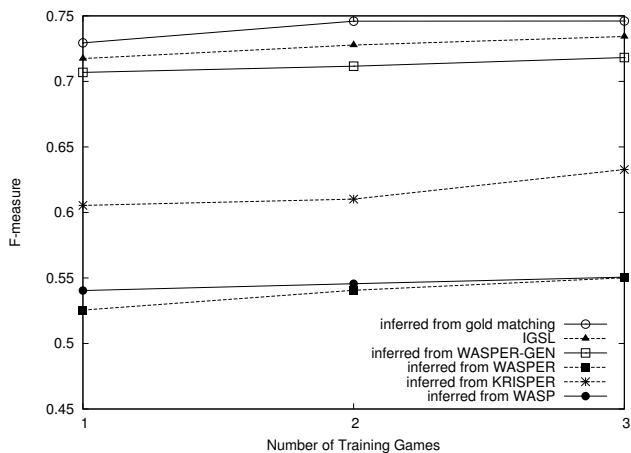

Figure 6. Strategic Generation Results

IGSL		WASPER-GEN	
corner_kick	1	pass	1
pass	0.983	badPass	0.708
badPass	0.970	corner_kick	0.438
goal	0.970	block	0.429
block	0.955	turnover	0.377

Table 2. Top scoring predicates with their estimated probabilities of being described

scores, try to learn a good translation model for each sentence pattern. On the other hand, WASPER-GEN only tries to learn a good translation model for each MR pattern. Thus, WASPER-GEN is more likely to converge on a good model as there are fewer MR patterns than sentence patterns. However, it can be argued that learning good translation models for each sentence pattern will help in producing more varied commentaries, a quality that is not captured by the NIST score.

5.4. Strategic Generation

The different methods for learning strategic generation are evaluated based on how often the events they describe coincide with those the human decided to describe in the test data. For the first method, results from using the inferred matchings produced by KRISPER, WASPER, KRISPER-WASP, and WASPER-GEN as well as the gold and random matching for establishing baselines are all presented in Figure 6. From the graph, it is clear that IGSL outperforms learning from the inferred matchings and actually performs at a level close to using the gold matching. However, it is important to note that we are limiting the potential of learning from the gold matching by using only the predicates to decide whether to talk about an event.

The top scoring predicates from IGSL as well as the best result from using inferred matchings, WASPER-GEN, are

	English Fluency	Semantic Correctness	Sportscasting Ability
Human	3.938	4.25	3.625
Machine	3.438	3.563	2.938

Table 3. Human evaluation of overall sportscast

shown in Table 2. While both systems learn to talk about frequent events such as passing, WASPER-GEN does poorly on rare, but significant events such as goal scoring. This is because WASPER-GEN saw those events very rarely in training and did not learn to correctly match them to sentences. It is worth noting that IGSL learns a higher probability for events in general. This improves its recall and hurts its precision. However, since many of its top-ranked events such as goals are rare, the overall quality is maintained without becoming overly verbose. Therefore, we used IGSL for the human evaluations below.

5.5. Human Evaluation

Automatic evaluation of generation is an imperfect approximation of human assessment at best. Moreover, automatically evaluating the quality of an entire generated sportscast is even more difficult. Consequently, we recruited four fluent English speakers with no previous experience with Robocup or any of our systems to serve as human judges. We compared their subjective evaluations of human and machine generated sportscasts. Each judge was given 8 clips of simulated game video along with subtitled commentaries. The 8 clips use 4 game segments of 2 minutes each, one from each of the four games. Each of the 4 game segments is shown twice, once with human commentary and once with generated commentary. We use IGSL to determine the events to comment on and use WASPER-GEN (our best performing system for generation) to produce the commentaries. The system was always trained on three games, leaving out the one from which the test segment was extracted. The videos are shown in random order with the human and machine commentaries of a segment flipped between judges to ensure no consistent bias toward segments being shown earlier or later. We asked the judges to score the commentaries using the following metrics:

Score	English Fluency	Semantic Correctness	Sportscasting Ability
5	Flawless	Always	Excellent
4	Good	Usually	Good
3	Non-native	Sometimes	Average
2	Disfluent	Rarely	Bad
1	Gibberish	Never	Terrible

Fluency and semantic correctness, or adequacy, are standard metrics in human evaluations of NL translations and generations. Fluency measures how well the commentaries are structured, including syntax and grammar. Semantic

correctness indicates whether the commentaries actually describe what is happening in the game. Finally, sportscasting ability measures the overall quality of the sportscast. This includes whether the sportscasts are interesting and flow well. The scores are averaged over all four games and across all the judges. Table 3 shows the results.

While human commentaries are clearly superior to the machine's, the largest difference between the average scores is only 0.7. Moreover, the judges indicated that they were able to understand and follow the generated commentaries without trouble. Part of the reason for the lower scores actually result from our impoverished MRL. Semantic correctness scores were deducted when the machine misses commenting on certain facts not represented in our MRL such as the location of the ball and the players. The lack of temporal or locality information also results in dry and repetitive comments which hurt the sportscasting score. This is an important point that is not captured by the NIST score. In our NIST score evaluation, each sentence is treated separately and no attempt was made at measuring how well the individual comments fit together. However, it is clear from the human evaluations that variability of sentence pattern is vital to a good sportscast. The machine can correctly comment on all the factual events in a game and still produce a bad sportscast that no one wants to listen to.

6. Related Work

Robotics and vision researchers have worked on inferring a grounded meaning of individual words or short referring expressions from visual perceptual context, e.g. (Roy, 2002; Bailey et al., 1997; Barnard et al., 2003; Yu & Ballard, 2004). However, the complexity of the natural language used in this existing work is very restrictive, many of the systems use pre-coded knowledge of the language, and almost all use static images to learn language describing objects and their relations, and cannot use dynamic video to learn language describing actions. Some recent work on video retrieval has focused on learning to recognize events in sports videos and connect them to English words (Fleischman & Roy, 2007). There has also been recent work on grounded language learning in simulated computer-game environments (Gorniak & Roy, 2005). However, none of this prior work makes use of modern statistical-NLP parsing techniques, learns to build formal meaning representations for complete sentences, or learns to generate natural language.

There has been some recent work on learning generation strategies using reinforcement learning (Zaragoza & Li, 2005). In contrast, our domain does not include interaction with the users and no feedback is available.

7. Future Work

The current system is limited by its simple MRL. For example, the location of players or the ball is not represented. Moreover, we do not keep contextual information which makes it difficult to generate interesting, non-repetitive sportscasts. Contextual information would also help us provide comments not directly induced by the events happening now, such as the current score. Finally, it is clear that we need a more hierarchical representation that captures the relationships between events in order to avoid making systematic matching errors on frequently co-occurring events.

With respect to algorithms, using learned strategic-generation knowledge (information about what events are likely to illicit comments) could improve the resolution of ambiguities. We would also like to eventually apply our methods to real captioned video input using the latest methods in computer vision.

8. Conclusion

We have presented an end-to-end system that learns from sample commentaries and generates sportscasts for novel games. Dealing with the ambiguity inherent in the training environment is a critical issue in learning language from perceptual context. We have evaluated various methods for disambiguating the training data in order to build a language generator. Using a generation evaluation metric as the criterion for selecting the best NL–MR pairs produced the best results overall. Our system also learns a simple model of strategic generation from the ambiguous training data by estimating the probability that each event type invokes a comment. Experimental evaluation verified that the system learns to accurately parse and generate comments and to generate sportscasts that are competitive with those produced by humans.

Acknowledgement

References

André, E., Binsted, K., Tanaka-Ishii, K., Luke, S., Herzog, G., & Rist, T. (2000). Three RoboCup simulation league commentator systems. *AI Magazine*, *21*, 57–66.

Bailey, D., Feldman, J., Narayanan, S., & Lakoff, G. (1997). Modeling embodied lexical development. *COGSC-97*.

Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D. M., & Jordan, M. I. (2003). Matching words and pictures. *Journal of Machine Learning Research*, *3*, 1107–1135.

Chen, M., Foroughi, E., Heintz, F., Kapetanakis, S., Kostiadis, K., Kummeneje, J., Noda, I., Obst, O., Riley, P., Steffens, T., Wang, Y., & Yin, X. (2003). Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at `http://sourceforge.net/projects/sserver/`.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. *ACL-05* (pp. 263–270). Ann Arbor, MI.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. *Proc. of ARPA Workshop on Human Language Technology* (pp. 128–132). San Diego, CA.

Fleischman, M., & Roy, D. (2007). Situated models of meaning for sports video retrieval. *NAACL-HLT-07*. Rochester, NY.

Gorniak, P., & Roy, D. (2005). Speaking with your sidekick: Understanding situated speech in computer role playing games. *AIIDE-05*. Stanford, CA.

Harnad, S. (1990). The symbol grounding problem. *Physica D*, *42*, 335–346.

Kate, R. J., & Mooney, R. J. (2006). Using string-kernels for learning semantic parsers. *ACL-06* (pp. 913–920). Sydney, Australia.

Kate, R. J., & Mooney, R. J. (2007). Learning language semantics from ambiguous supervision. *AAAI-2007* (pp. 895–900).

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, 2, 419–444.

Mooney, R. J. (2007). Learning for semantic parsing. In A. Gelbukh (Ed.), *Computational linguistics and intelligent text processing: Proc. of the 8th Intl. Conference, CICLing 2007, Mexico City*, 311–324. Berlin: Springer Verlag.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. *ACL-02* (pp. 311–318). Philadelphia, PA.

Roy, D. (2002). Learning visually grounded words and syntax for a scene description task. *Computer Speech and Language*, *16*, 353–385.

Wong, Y. W., & Mooney, R. (2007a). Generation by inverting a semantic parser that uses statistical machine translation. *NAACL-HLT-07* (pp. 172–179).

Wong, Y. W., & Mooney, R. J. (2007b). Learning synchronous grammars for semantic parsing with lambda calculus. *ACL-07* (pp. 960–967).

Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, *23*, 377–403.

Yu, C., & Ballard, D. H. (2004). On the integration of grounding language and learning objects. *AAAI-2004* (pp. 488–493).

Zaragoza, H., & Li, C.-H. (2005). Learning what to talk about in descriptive games. *HLT/EMNLP-05* (pp. 291–298). Vancouver, Canada.