Corpus-based interpretation of instructions in virtual environments

Luciana Benotti[1] **Martín Villalba**[1] **Tessa Lau**[2] **Julián Cerruti**[3]
[1] FaMAF, Medina Allende s/n, Universidad Nacional de Córdoba, Córdoba, Argentina
[2]IBM Research – Almaden, 650 Harry Road, San Jose, CA 95120 USA
[3]IBM Argentina, Ing. Butty 275, C1001AFA, Buenos Aires, Argentina
{benotti,villalba}@famaf.unc.edu.ar, tessalau@us.ibm.com, jcerruti@ar.ibm.com

Abstract

Previous approaches to instruction interpretation have required either extensive domain adaptation or manually annotated corpora. This paper presents a novel approach to instruction interpretation that leverages a large amount of unannotated, easy-to-collect data from humans interacting with a virtual world. We compare several algorithms for automatically segmenting and discretizing this data into (utterance, reaction) pairs and training a classifier to predict reactions given the next utterance. Our empirical analysis shows that the best algorithm achieves 70% accuracy on this task, with no manual annotation required.

1 Introduction and motivation

Mapping instructions into automatically executable actions would enable the creation of natural language interfaces to many applications (Lau et al., 2009; Branavan et al., 2009; Orkin and Roy, 2009). In this paper, we focus on the task of navigation and manipulation of a virtual environment (Vogel and Jurafsky, 2010; Chen and Mooney, 2011).

Current symbolic approaches to the problem are brittle to the natural language variation present in instructions and require intensive rule authoring to be fit for a new task (Dzikovska et al., 2008). Current statistical approaches require extensive manual annotations of the corpora used for training (MacMahon et al., 2006; Matuszek et al., 2010; Gorniak and Roy, 2007; Rieser and Lemon, 2010). Manual annotation and rule authoring by natural language engineering experts are bottlenecks for developing conversational systems for new domains.

This paper proposes a fully automated approach to interpreting natural language instructions to complete a task in a virtual world based on unsupervised recordings of human-human interactions performing that task in that virtual world. Given unannotated corpora collected from humans following other humans' instructions, our system automatically segments the corpus into labeled training data for a classification algorithm. Our interpretation algorithm is based on the observation that similar instructions uttered in similar contexts should lead to similar actions being taken in the virtual world. Given a previously unseen instruction, our system outputs actions that can be directly executed in the virtual world, based on what humans did when given similar instructions in the past.

2 Corpora situated in virtual worlds

Our environment consists of six virtual worlds designed for the natural language generation shared task known as the GIVE Challenge (Koller et al., 2010), where a pair of partners must collaborate to solve a task in a 3D space (Figure 1). The "instruction follower" (IF) can move around in the virtual world, but has no knowledge of the task. The "instruction giver" (IG) types instructions to the IF in order to guide him to accomplish the task. Each corpus contains the IF's actions and position recorded every 200 milliseconds, as well as the IG's instructions with their timestamps.

We used two corpora for our experiments. The C_m corpus (Gargett et al., 2010) contains instructions given by multiple people, consisting of 37 games spanning 2163 instructions over 8:17 hs. The

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 181–186,
Jeju, Republic of Korea, 8-14 July 2012.

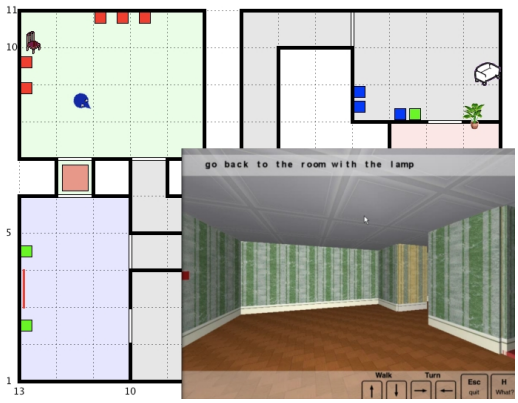

Figure 1: A screenshot of a virtual world. The world consists of interconnecting hallways, rooms and objects

C_s corpus (Benotti and Denis, 2011), gathered using a single IG, is composed of 63 games and 3417 instructions, and was recorded in a span of 6:09 hs. It took less than 15 hours to collect the corpora through the web and the subjects reported that the experiment was fun.

While the environment is restricted, people describe the same route and the same objects in extremely different ways. Below are some examples of instructions from our corpus all given for the same route shown in Figure 1.

1) out
2) walk down the passage
3) nowgo [sic] *to the pink room*
4) back to the room with the plant
5) Go through the door on the left
6) go through opening with yellow wall paper

People describe routes using landmarks (4) or specific actions (2). They may describe the same object differently (5 vs 6). Instructions also differ in their scope (3 vs 1). Thus, even ignoring spelling and grammatical errors, navigation instructions contain considerable variation which makes interpreting them a challenging problem.

3 Learning from previous interpretations

Our algorithm consists of two phases: annotation and interpretation. *Annotation* is performed only once and consists of automatically associating each IG instruction to an IF reaction. *Interpretation* is performed every time the system receives an instruction and consists of predicting an appropriate reaction given reactions observed in the corpus.

Our method is based on the assumption that a reaction captures the semantics of the instruction that caused it. Therefore, if two utterances result in the same reaction, they are paraphrases of each other, and similar utterances should generate the same reaction. This approach enables us to predict reactions for previously-unseen instructions.

3.1 Annotation phase

The key challenge in learning from massive amounts of easily-collected data is to automatically annotate an unannotated corpus. Our annotation method consists of two parts: first, *segmenting* a low-level interaction trace into utterances and corresponding reactions, and second, *discretizing* those reactions into canonical action sequences.

Segmentation enables our algorithm to learn from traces of IFs interacting directly with a virtual world. Since the IF can move freely in the virtual world, his actions are a stream of continuous behavior. Segmentation divides these traces into reactions that follow from each utterance of the IG. Consider the following example starting at the situation shown in Figure 1:

IG(1): go through the yellow opening
IF(2): [walks out of the room]
IF(3): [turns left at the intersection]
IF(4): [enters the room with the sofa]
IG(5): stop

It is not clear whether the IF is doing $\langle 3, 4 \rangle$ because he is reacting to 1 or because he is being proactive. While one could manually annotate this data to remove extraneous actions, our goal is to develop automated solutions that enable learning from massive amounts of data.

We decided to approach this problem by experimenting with two alternative formal definitions: 1) a strict definition that considers the maximum reaction according to the IF *behavior*, and 2) a loose definition based on the empirical observation that, in situated interaction, most instructions are constrained by the current *visually* perceived affordances (Gibson, 1979; Stoia et al., 2006).

We formally define *behavior segmentation* (Bhv) as follows. A reaction r_k to an instruction u_k begins

right after the instruction u_k is uttered and ends right before the next instruction u_{k+1} is uttered. In the example, instruction 1 corresponds to $\langle 2, 3, 4 \rangle$. We formally define *visibility segmentation* (Vis) as follows. A reaction r_k to an instruction u_k begins right after the instruction u_k is uttered and ends right before the next instruction u_{k+1} is uttered or right after the IF leaves the area visible at 360° from where u_k was uttered. In the example, instruction 1's reaction would be limited to $\langle 2 \rangle$ because the intersection is not visible from where the instruction was uttered.

The Bhv and Vis methods define how to segment an interaction trace into utterances and their corresponding reactions. However, users frequently perform noisy behavior that is irrelevant to the goal of the task. For example, after hearing an instruction, an IF might go into the wrong room, realize the error, and leave the room. A reaction should not include such irrelevant actions. In addition, IFs may accomplish the same goal using different behaviors: two different IFs may interpret "go to the pink room" by following different paths to the same destination. We would like to be able to generalize both reactions into one canonical reaction.

As a result, our approach *discretizes* reactions into higher-level action sequences with less noise and less variation. Our discretization algorithm uses an *automated planner* and a *planning representation* of the task. This planning representation includes: (1) the task goal, (2) the actions which can be taken in the virtual world, and (3) the current state of the virtual world. Using the planning representation, the planner calculates an optimal path between the starting and ending states of the reaction, eliminating all unnecessary actions. While we use the classical planner FF (Hoffmann, 2003), our technique could also work with *classical planning* (Nau et al., 2004) or other techniques such as *probabilistic planning* (Bonet and Geffner, 2005). It is also not dependent on a particular discretization of the world in terms of actions.

Now we are ready to define *canonical reaction* c_k formally. Let S_k be the state of the virtual world when instruction u_k was uttered, S_{k+1} be the state of the world where the reaction ends (as defined by Bhv or Vis segmentation), and D be the planning domain representation of the virtual world. The *canonical reaction* to u_k is defined as the sequence of actions returned by the planner with S_k as initial state, S_{k+1} as goal state and D as planning domain.

3.2 Interpretation phase

The annotation phase results in a collection of (u_k, c_k) pairs. The interpretation phase uses these pairs to interpret new utterances in three steps. First, we *filter* the set of pairs into those whose reactions can be directly executed from the current IF position. Second, we *group* the filtered pairs according to their reactions. Third, we *select* the group with utterances most similar to the new utterance, and output that group's reaction. Figure 2 shows the output of the first two steps: three groups of pairs whose reactions can all be executed from the IF's current position.

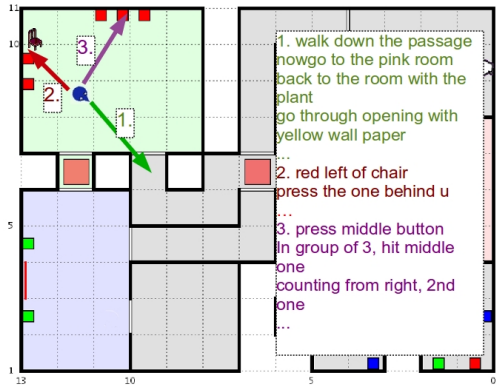

Figure 2: Utterance groups for this situation. Colored arrows show the reaction associated with each group.

We treat the third step, selecting the most similar group for a new utterance, as a classification problem. We compare three different classification methods. One method uses nearest-neighbor classification with three different similarity metrics: Jaccard and Overlap coefficients (both of which measure the degree of overlap between two sets, differing only in the normalization of the final value (Nikravesh et al., 2005)), and Levenshtein Distance (a string metric for measuring the amount of differences between two sequences of words (Levenshtein, 1966)). Our second classification method employs a strategy in which we considered each group as a set of possible machine translations of our utterance, using the BLEU measure (Papineni et al., 2002) to select which group could be considered the best translation of our utterance. Finally, we trained an SVM classifier (Cortes and Vapnik, 1995) using the unigrams

	Corpus C_m		Corpus C_s	
Algorithm	**Bhv**	**Vis**	**Bhv**	**Vis**
Jaccard	47%	54%	54%	70%
Overlap	43%	53%	45%	60%
BLEU	44%	52%	54%	50%
SVM	33%	29%	45%	29%
Levenshtein	21%	20%	8%	17%

Table 1: Accuracy comparison between C_m and C_s for Bhv and Vis segmentation

of each paraphrase and the position of the IF as features, and setting their group as the output class using a libSVM wrapper (Chang and Lin, 2011).

When the system misinterprets an instruction we use a similar approach to what people do in order to overcome misunderstandings. If the system executes an incorrect reaction, the IG can tell the system to cancel its current interpretation and try again using a paraphrase, selecting a different reaction.

4 Evaluation

For the evaluation phase, we annotated both the C_m and C_s corpora entirely, and then we split them in an 80/20 proportion; the first 80% of data collected in each virtual world was used for training, while the remaining 20% was used for testing. For each pair (u_k, c_k) in the testing set, we used our algorithm to predict the reaction to the selected utterance, and then compared this result against the automatically annotated reaction. Table 1 shows the results.

Comparing the Bhv and Vis segmentation strategies, Vis tends to obtain better results than Bhv. In addition, accuracy on the C_s corpus was generally higher than C_m. Given that C_s contained only one IG, we believe this led to less variability in the instructions and less noise in the training data.

We evaluated the impact of user corrections by simulating them using the existing corpus. In case of a wrong response, the algorithm receives a second utterance with the same reaction (a paraphrase of the previous one). Then the new utterance is tested over the same set of possible groups, except for the one which was returned before. If the correct reaction is not predicted after four tries, or there are no utterances with the same reaction, the predictions are registered as wrong. To measure the effects of user corrections vs. without, we used a different evaluation process for this algorithm: first, we split the corpus in a 50/50 proportion, and then we moved correctly predicted utterances from the testing set towards training, until either there was nothing more to learn or the training set reached 80% of the entire corpus size.

As expected, user corrections significantly improve accuracy, as shown in Figure 3. The worst algorithm's results improve linearly with each try, while the best ones behave asymptotically, barely improving after the second try. The best algorithm reaches 92% with just one correction from the IG.

5 Discussion and future work

We presented an approach to instruction interpretation which learns from non-annotated logs of human behavior. Our empirical analysis shows that our best algorithm achieves 70% accuracy on this task, with no manual annotation required. When corrections are added, accuracy goes up to 92% for just one correction. We consider our results promising since state of the art semi-unsupervised approaches to instruction interpretation (Chen and Mooney, 2011) reports a 55% accuracy on manually segmented data.

We plan to compare our system's performance against human performance in comparable situations. Our informal observations of the GIVE corpus indicate that humans often follow instructions incorrectly, so our automated system's performance may be on par with human performance.

Although we have presented our approach in the context of 3D virtual worlds, we believe our technique is also applicable to other domains such as the web, video games, or Human Robot Interaction.

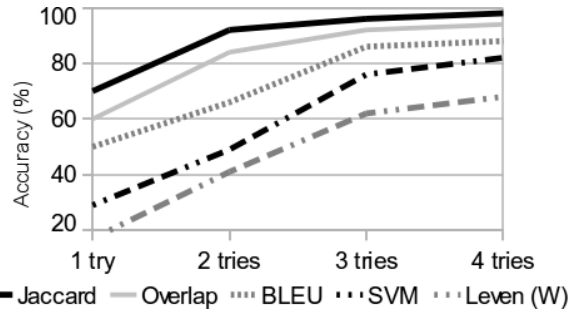

Figure 3: Accuracy values with corrections over C_s

References

Luciana Benotti and Alexandre Denis. 2011. CL system: Giving instructions by corpus based selection. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 296–301, Nancy, France, September. Association for Computational Linguistics.

Blai Bonet and Héctor Geffner. 2005. mGPT: a probabilistic planner based on heuristic search. *Journal of Artificial Intelligence Research*, 24:933–944.

S.R.K. Branavan, Harr Chen, Luke Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 82–90, Suntec, Singapore, August. Association for Computational Linguistics.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011)*, pages 859–865, August.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20:273–297.

Myroslava O. Dzikovska, James F. Allen, and Mary D. Swift. 2008. Linking semantic and knowledge representations in a multi-domain dialogue system. *Journal of Logic and Computation*, 18:405–430, June.

Andrew Gargett, Konstantina Garoufi, Alexander Koller, and Kristina Striegnitz. 2010. The GIVE-2 corpus of giving instructions in virtual environments. In *Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC)*, Malta.

James J. Gibson. 1979. *The Ecological Approach to Visual Perception*, volume 40. Houghton Mifflin.

Peter Gorniak and Deb Roy. 2007. Situated language understanding as filtering perceived affordances. *Cognitive Science*, 31(2):197–231.

Jörg Hoffmann. 2003. The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. *Journal of Artificial Intelligence Research (JAIR)*, 20:291–341.

Alexander Koller, Kristina Striegnitz, Andrew Gargett, Donna Byron, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. 2010. Report on the second challenge on generating instructions in virtual environments (GIVE-2). In *Proceedings of the 6th International Natural Language Generation Conference (INLG)*, Dublin.

Tessa Lau, Clemens Drews, and Jeffrey Nichols. 2009. Interpreting written how-to instructions. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1433–1438, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: connecting language, knowledge, and action in route instructions. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, pages 1475–1482. AAAI Press.

Cynthia Matuszek, Dieter Fox, and Karl Koscher. 2010. Following directions using statistical machine translation. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, HRI '10, pages 251–258, New York, NY, USA. ACM.

Dana Nau, Malik Ghallab, and Paolo Traverso. 2004. *Automated Planning: Theory & Practice*. Morgan Kaufmann Publishers Inc., California, USA.

Masoud Nikravesh, Tomohiro Takagi, Masanori Tajima, Akiyoshi Shinmura, Ryosuke Ohgaya, Koji Taniguchi, Kazuyosi Kawahara, Kouta Fukano, and Akiko Aizawa. 2005. Soft computing for perception-based decision processing and analysis: Web-based BISC-DSS. In Masoud Nikravesh, Lotfi Zadeh, and Janusz Kacprzyk, editors, *Soft Computing for Information Processing and Analysis*, volume 164 of *Studies in Fuzziness and Soft Computing*, chapter 4, pages 93–188. Springer Berlin / Heidelberg.

Jeff Orkin and Deb Roy. 2009. Automatic learning and generation of social behavior from collective human gameplay. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent SystemsVolume 1*, volume 1, pages 385–392. International Foundation for Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Verena Rieser and Oliver Lemon. 2010. Learning human multimodal dialogue strategies. *Natural Language Engineering*, 16:3–23.

Laura Stoia, Donna K. Byron, Darla Magdalene Shockley, and Eric Fosler-Lussier. 2006. Sentence planning

for realtime navigational instructions. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pages 157–160, Stroudsburg, PA, USA. Association for Computational Linguistics.

Adam Vogel and Dan Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 806–814, Stroudsburg, PA, USA. Association for Computational Linguistics.