Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Natural Language Dialogue for Building and Learning Models and Structures

Ian Perera,[1] James F. Allen,[1,2] Lucian Galescu,[1] Choh Man Teng,[1]
Mark Burstein,[3] Scott Friedman,[3] David McDonald,[3], Jeffrey Rye[3]
[1]Institute for Human and Machine Cognition (IHMC), 40 S. Alcaniz, Pensacola, FL 32502
[2]Department of Computer Science, University of Rochester, 500 Joseph C. Wilson Blvd., Rochester, NY 14627
[3]SIFT, LLC, 319 1st Ave North, Suite 400, Minneapolis, MN 55401

Abstract

We demonstrate an integrated system for building and learning models and structures in both a real and virtual environment. The system combines natural language understanding, planning, and methods for composition of basic concepts into more complicated concepts. The user and the system interact via natural language to jointly plan and execute tasks involving building structures, with clarifications and demonstrations to teach the system along the way.

We use the same architecture for building and simulating models of biology, demonstrating the general-purpose nature of the system where domain-specific knowledge is concentrated in sub-modules with the basic interaction remaining domain-independent. These capabilities are supported by our work on semantic parsing, which generates knowledge structures to be grounded in a physical representation, and composed with existing knowledge to create a dynamic plan for completing goals. Prior work on learning from natural language demonstrations enables learning of models from very few demonstrations, and features are extracted from definitions in natural language. We believe this architecture for interaction opens up a wide possibility of human-computer interaction and knowledge transfer through natural language.

While great advances are being made in the ability of AI systems to process large amounts of data and learn complex decision processes for analysis, interaction with such systems is mostly limited to users with years of experience building such systems. AI-Human collaboration with domain experts, opening a wide variety of opportunities for groundbreaking research, is limited by the lack of a natural interface to convey concepts and work alongside such systems rather than simply using them as analytics tools.

While this system architecture is also used in a separate domain, Biocuration (where the user interacts with the system to build biological models), we demonstrate our system in the Blocks World domain to show ease of interaction without prior knowledge, operating in both a physical implementation and a virtual 2D implementation. The user can interact with the system to teach the system new structures, query its knowledge, and build a structure with constraints in a mixed-initiative dialogue. Maintaining dialogue and problem solving state allows the user to switch between these interactions effortlessly towards completing a shared goal, while also allowing the system to recover from errors or obstacles towards completing goals. Our emphasis on robust dialogue management and modular integration of domain-dependent and domain-independent components is one of the first steps towards better collaboration with AI systems.

Related Work

Our work has similarities to SHRDLU (Winograd 1971), especially in the domain and use of natural language as an interface. However, we go beyond reference resolution and handle dialogues as a progression towards a goal, not as a series of question-answer pairs. Our system also learns from interaction and can guide the user to carry out various tasks. The Playmate system (Skocaj et al. 2011) and SALL-E (Perera and Allen 2013) also incorporate learning through dialogue and demonstration, but do not place as much emphasis on the turn taking behavior needed as the robotic systems developed by Chao and Thomaz (2011).

Interface and Hardware

Our system consists of two possible operating modes, a virtual mode and a physical mode. In the first, the user interacts with the system in a web interface with a 2D representation of blocks on a table. This virtual environment is imbued with basic physics, but there is no rotation. Each block is a single color, with multiple blocks available in each color.

In the physical mode, the system consists of a TV monitor on a table, two Kinect 2.0 cameras for RGB+depth and audio recording, a central server, a smaller computer for Kinect processing, a Microsoft Surface as an interface for the ”proxy” (a person to act in the computer’s stead to move blocks), and a router for communication between the two computers and the Surface. The Kinect cameras are mounted on either side of the table and calibrated to generate a single point field for the table environment. Blocks are 5-inch cubes with a logo on each face and a different color on each side. The logo is used to generate a unique identifier as well as provide a means for the user to reference certain blocks. The blocks can be placed and stacked on the table in any orientation. The system currently supports more than 12 blocks on the table at once, although often occlusion limits the number of blocks that can be practically recognized.

Feedback to the user in the physical apparatus is provided through multiple modalities. First, the apparatus displays an avatar to the user, which is capable of pointing, gesturing, making facial expressions, and speaking to the user. The system can also "move" blocks by displaying desired block positions on the Surface for the proxy. The system can speak using MaryTTS, an open-source text-to-speech library that supports annotation for inflection and other modifications to prosody (Schröder and Trouvain 2003).

System Architecture

Natural language parsing, dialogue management, goal management, learning, and planning are each developed as independent modules to facilitate independent work and allow for the various needs of domain specificity at different levels of the system. Message passing via KQML (Finin et al. 1994) provides a standard means of communication between modules. This modular structure allows our system to handle both the Blocks World and Biocuration domain with little change to the system architecture and many components are identical across the two domains.

The natural language parsing is handled by the TRIPS parser (Allen, Swift, and de Beaumont 2008), which generates a semantic representation structured around events and their various arguments. This parser is integrated with a general ontology that is augmented with domain-specific knowledge – types of structures and components in the Blocks World domain, and biological knowledge from various databases in the Biocuration domain. The resulting logical form is then used to determine an appropriate problem-solving act, progressing the dialogue and task state.

Dialogue and Collaborative State Management

Rather than using a global set of goal states, we use the notion of a *collaborative state* to represent the interplay between the user and the system as they each express their own goals, actions, and knowledge. Changes in collaborative problem state occur via collaborative problem solving acts, such as *propose*, *accept*, *reject*, *etc.*, which refer to state-change acts applied to goals, such as "build a tower", or domain-specific actions, such as "add relation to the biological model". For example, the user might propose a goal for which there are not enough blocks of the correct type, for example. The system can then propose changes to the goal, which may be accepted or rejected by the user. Maintaining these states as a graph allows the system to return to the relevant plan or action when subproblems in the collaboration process have been resolved.

After parsing an utterance, we use a set of domain-independent rules for determining shifts in collaborative state via collaborative problem solving acts, such as proposing goals or informing the other participant of goal failures. This new state is passed to domain-specific modules to evaluate the problem in relation to the environment or model. The system's response to a change in problem solving state is then passed to a natural language generation module which provides a response to the user about the success or failure of its last action and next steps to be taken.

Model Learning and Planning

The system can develop constraint models from natural language descriptions provided that the system can ground the constraints in generated features of structures. For example, if we take a definition of a tower to be "a structure taller than its diameter", we can generate the features of height and diameter for any given arrangement of blocks, create a constraint between these features, and query the system as to whether a structure satisfies these constraints.

When building a known structure, the system uses the SHOP2 planner to generate a graph of plan states representing the possible, potentially underspecified states of the world towards the goal states as well as the actions (*i.e.*, placing or moving blocks) that form transitions between states. The states contain the relations between blocks, but not specific numerical coordinates, allowing the same plans to be used in the 2D or 3D case and allowing multiple real-world block arrangements to satisfy each state.

Rather than re-plan when the user deviates from the prescribed action towards the next state, we have the flexibility to evaluate the world against its plan and localize the world within the context of the plan. This allows the system to recognize that the world may be further along in the plan than expected, that a few simple actions can be taken to return into the plan graph, or that progress toward the goal(s) has regressed over time, all without having the system erase its planning context.

Acknowledgements

This work is supported by the DARPA CwC program. Special thanks to SRI for their work in developing the physical apparatus, including block detection and avatar software.

References

Allen, J.; Swift, M.; and de Beaumont, W. 2008. Deep Semantic Analysis of Text. In *Symposium on Semantics in Systems for Text Processing (STEP)*, 343–354. Morristown, NJ, USA: Association for Computational Linguistics.

Chao, C., and Thomaz, A. L. 2011. Timing in Multimodal Turn-Taking Interactions : Control and Analysis Using Timed Petri Nets. *Journal of Human-Robot Interaction* 1(1):1–16.

Finin, T.; Fritzson, R.; Mckay, D.; and Mcentire, R. 1994. KQML as an Agent Communication Language. In *Proceedings of the Third International Conference on Information and Knowledge Management*. ACM Press.

Perera, I., and Allen, J. 2013. SALL-E: Situated Agent for Language Learning. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Schröder, M., and Trouvain, J. 2003. The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching. *International Journal of Speech Technology* 6:365–377.

Skocaj, D.; Kristan, M.; Vrecko, A.; Mahnic, M.; Janicek, M.; Kruijff, G.-J. M.; Hanheide, M.; Hawes, N.; Keller, T.; Zillich, M.; and Zhou, K. 2011. A system for interactive learning in dialogue with a tutor. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3387–3394. IEEE.

Winograd, T. 1971. Procedures as a Representation for Data in a Computer for Understanding Natural Language. Technical report, Massachusetts Institute of Technology Artificial Intelligence.