Computer learns language by playing games

July 12 2011, by Larry Hardesty

"Civilization" is a strategy game in which players build empires by, among other things, deciding where to found cities and deploy armies. Image courtesy of Sid Meier's Civilization V

Computers are great at treating words as data: Word-processing programs let you rearrange and format text however you like, and search engines can quickly find a word anywhere on the Web. But what would it mean for a computer to actually understand the meaning of a sentence written in ordinary English -- or French, or Urdu, or Mandarin?

One test might be whether the computer could analyze and follow a set of instructions for an unfamiliar task. And indeed, in the last few years, researchers at MIT's Computer Science and Artificial Intelligence Lab have begun designing artificial-intelligence systems that do exactly that, with surprisingly good results.

In 2009, at the annual meeting of the Association for Computational

Linguistics (ACL), researchers in the lab of Regina Barzilay, associate professor of computer science and electrical engineering, took the best-paper award for a system that generated scripts for installing a piece of software on a Windows computer by reviewing instructions posted on Microsoft's help site. At this year's ACL meeting, Barzilay, her graduate student S. R. K. Branavan and David Silver of University College London applied a similar approach to a more complicated problem: learning to play "Civilization," a computer game in which the player guides the development of a city into an empire across centuries of human history. When the researchers augmented an artificial-intelligence system so that it could use a player's manual to guide the development of a game-playing strategy, its rate of victory jumped from 46 percent to 79 percent.

Starting from scratch

"Games are used as a test bed for artificial-intelligence techniques simply because of their complexity," says Branavan, who was first author on both ACL papers. "Every action that you take in the game doesn't have a predetermined outcome, because the game or the opponent can randomly react to what you do. So you need a technique that can handle very complex scenarios that react in potentially random ways."

Moreover, Barzilay says, game manuals have "very open text. They don't tell you how to win. They just give you very general advice and suggestions, and you have to figure out a lot of other things on your own." Relative to an application like the software-installing program, Branavan explains, games are "another step closer to the real world."

The extraordinary thing about Barzilay and Branavan's system is that it begins with virtually no prior knowledge about the task it's intended to perform or the language in which the instructions are written. It has a list of actions it can take, like right-clicks or left-clicks, or moving the

cursor; it has access to the information displayed on-screen; and it has some way of gauging its success, like whether the software has been installed or whether it wins the game. But it doesn't know what actions correspond to what words in the instruction set, and it doesn't know what the objects in the game world represent.

So initially, its behavior is almost totally random. But as it takes various actions, different words appear on screen, and it can look for instances of those words in the instruction set. It can also search the surrounding text for associated words, and develop hypotheses about what actions those words correspond to. Hypotheses that consistently lead to good results are given greater credence, while those that consistently lead to bad results are discarded.

Proof of concept

In the case of software installation, the system was able to reproduce 80 percent of the steps that a human reading the same instructions would execute. In the case of the computer game, it won 72 percent more frequently than a version of the same system that didn't use the written instructions as input, and 27 percent more frequently than a more sophisticated system that used additional artificial-intelligence techniques to improve its performance.

"If you'd asked me beforehand if I thought we could do this yet, I'd have said no," says Eugene Charniak, University Professor of Computer Science at Brown University. "You are building something where you have very little information about the domain, but you get clues from the domain itself."

Charniak points out that when the MIT researchers presented their work at the ACL meeting, some members of the audience argued that more sophisticated machine-learning systems would have performed better

than the ones to which the researchers compared their system. But, Charniak adds, "it's not completely clear to me that that's really relevant. Who cares? The important point is that this was able to extract useful information from the manual, and that's what we care about."

Most computer games as complex as "Civilization" include algorithms that allow players to play against the computer, rather than against other people; the games' programmers have to develop the strategies for the computer to follow and write the code that executes them. Barzilay and Branavan say that, in the near term, their system could make that job much easier, automatically creating algorithms that perform better than the hand-designed ones.

But the main purpose of the project, which was supported by the National Science Foundation, was to demonstrate that computer systems that learn the meanings of words through exploratory interaction with their environments are a promising subject for further research. And indeed, Barzilay and her students have begun to adapt their meaning-inferring algorithms to work with robotic systems.

This story is republished courtesy of MIT News (web.mit.edu/newsoffice/), a popular site that covers news about MIT research, innovation and teaching.

More information: Research paper: people.csail.mit.edu/regina/my_papers/civ11.pdf

Provided by Massachusetts Institute of Technology

from https://phys.org/news/2011-07-language-games.html