Developing an AI Player for Guess Who
By Jason Pan

Abstract

My project is to create a computerized version of the popular Milton Bradley game “‘Guess Who?” complete with an AI player. This involves two research areas: Natural Language Processing and Data Mining. Data mining is the analysis of data and the use of software techniques for finding patterns and regularities in sets of data. My project can be divided into two major phases: Developing the Game Interface and Developing the AI Player. My game interface consists of a matrix of buttons with pictures of the suspects and an input text field where questions can be entered. My AI's strategy algorithm will formulate questions that eliminate 50% of the suspects, which is the optimal percentage. If my program can beat an opponent at least half of the time, then I can deem it successful.

Data Mining

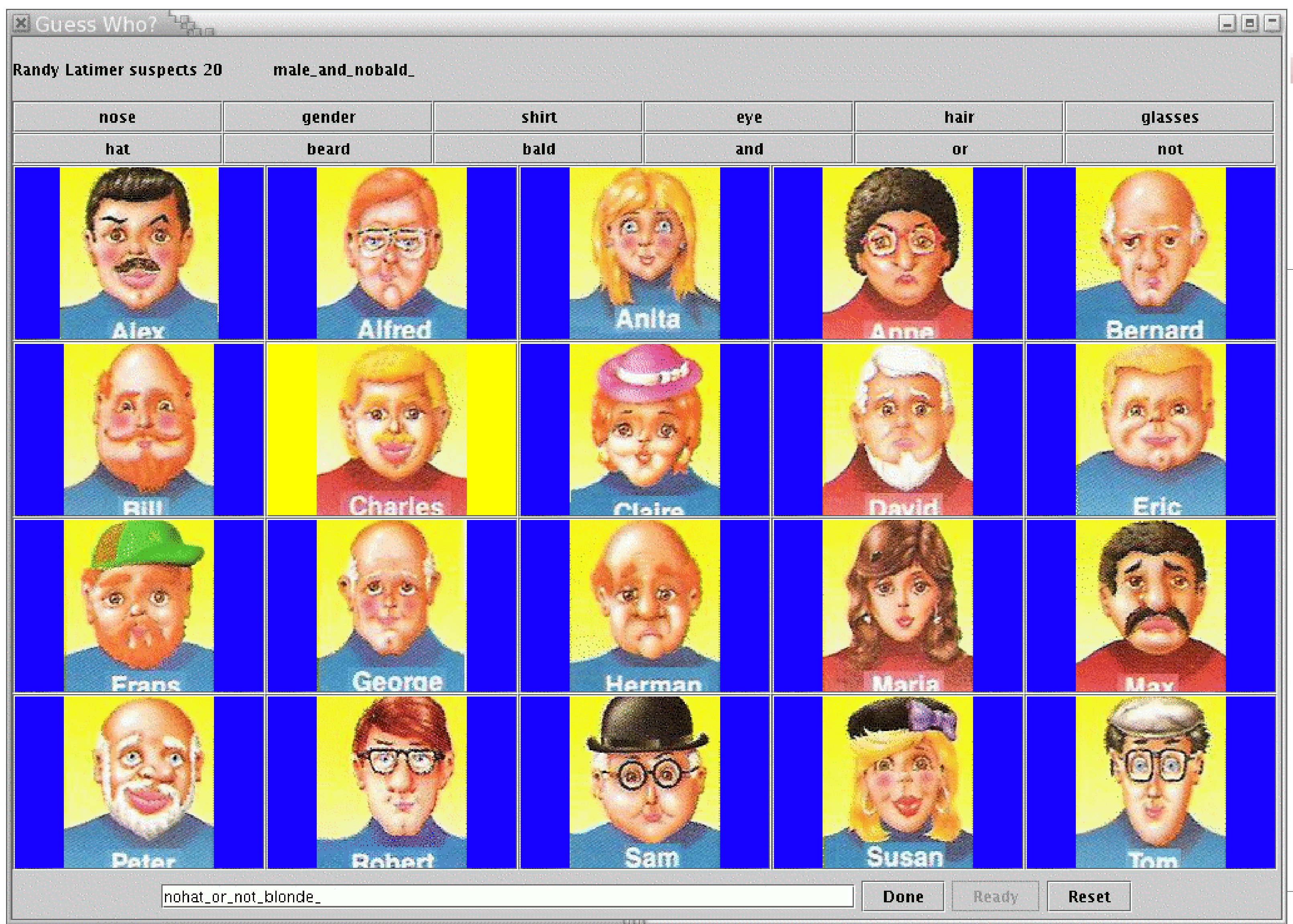
Data mining, also known as knowledge-discovery in databases (KDD), uses computational techniques like statistics and pattern recognition. Although it is usually used in relation to analysis of data, data mining, like artificial intelligence, is an umbrella term and is used with varied meaning in a wide range of contexts. Its official definition is the "nontrivial extraction of implicit, previously unknown, and potentially useful information from data." This encompasses a number of technical approaches, such as clustering, data summarization, learning classification rules, finding dependency networks, analyzing changes, and detecting anomalies. My project however implements an elementary form of KDD; one that does not require complicated equations or extremely sophisticated search algorithms. Developing data mining software often involves corporate funding in the millions and teams of programmers working in cooperation with project deadlines of years. Because I work solo and only have the length of an year, my project can at best produce only a crude but effective data mining function.

Implementation

I decided to code my game in Java because of my substantial experience with it and the language's conveniently built-in GUI. My game is very low budget and the only resources I needed were a java compiler and internet access. The research paper was written in LaTex, and this poster and the Powerpoint presentation was created using OpenOffice on Linux machines.

My project consists of four files. To store the suspects I wrote a special Person class whose fields were physical attributes. The object has a constructor, modifying methods, retrieval methods, and an equals method. GIPanel.java is the main file, and it comprises of the two-player game interface. An extended one-player version is AIPanel.java, which in addition contains the AI opponent. Both of these files are saved in JPanel form, which Driver.java then loads up and runs.

My project can be divided into three iterations, each of which required one school year quarter. The fourth quarter was spent on completing the research paper and preparing this poster and a Powerpoint presentation. First quarter I concentrated mostly on research and organization. Second quarter was focused on Game Structure and Logistics, and third quarter was spent researching and developing the AI agent.



Randy Latimer suspects 20 male_and_nobald_

nohat_or_not_blonde_

Algorithm

I have been an avid player of Guess Who for at least four years. The beauty of the game is its infiniteness of possibilities. I have never seen the exact same game play duplicated twice. In the real world, questions can be as long and complicated as desired. However constrained to the still primitive reality of today's computers, there are limitations on the player’s ability to compose creative queries. Still this is only a minor hindrance since most players can compute only up to two or three traits. Usually the safest strategy is to ask questions that for yes or no will eliminate half of the suspects. Some bold individuals dare a move whose breakdown lets say is if "yes," eliminate 70% of the suspects, and "no," eliminate 30%. This is unnecessarily dangerous, since if "no" is the response, the player will suffer a major setback. The proper strategy is to make steady gains, every turn eliminating the volume of suspects by a factor of two, and wait for the opponent to make a sloppy mistake and fall behind. It is only appropriate to gamble a risky, uneven question when the opponent has much smaller volume of suspects than you do.

The AI strategy algorithm is simple. Using a depth first search, iterate through the list of traits. If any trait is shared by 45%-55% of the current roster of suspects, that trait is formed into a question. If not, the AI moves on to plan B. The percentage each trait eliminates is recorded in an array. On a systematic basis two traits are chosen, and every possible combination is run through it. These include adding a conjunction or prefixing a "not." Each scenario is checked on whether the condition is met. For "and," the previously-calculated percentages of both traits are multiplied together. For "or," they are added up. Adding a "not" subtracts the trait’s percentage from 100. During all of this, any question whose range falls between 35%-65% is stored onto an array in sorted order. In the event that no question matches the original description, the next best question is popped out and used. Lastly if all else fails, a random question will be selected. In the event that opponent has 2/3 or less of the suspects of the AI, in order catch up the percentage constraint will be temporarily expanded to 25%-75% to allow a possible catch up.

Natural Language Processing

Although I did not plan so, my project involves elements of Natural Language Processing. The nature of Guess Who is that the user enters questions, which my program then has to break down and interpret. To achieve such a task I wrote a very primitive natural language processor. The user can only enter up to five words into the input text field. This includes a maximum of two traits and one conjunction connecting them. To prevent any input typos, I created a panel of buttons that the user presses to formulate his question. The syntax of the input must be precise, and I added a throw exception clause to catch any errors. If the format of the question is not exactly correct, the program will not accept it and asks the user to rewrite his query.

Rules

One of my favorite childhood pastimes during really long road trips was Milton Bradley’s "Guess Who?", a simple two-player game. In it each player has a board with pictures of twenty different people labeled with their names. To begin, the opponents each choose a mystery person from the list of twenty. Afterwards in subsequent turns, each player asks his opponent a yes-or-no question about the Mystery Person, and he then uses the clues to narrow down the twenty suspects into the answer. A player can only guess the opponent’s mystery person once, and if he succeeds he wins the game.