Open Mind *Animals*: Insuring the quality of data openly contributed over the World Wide Web

David G. Stork and **Chuck P. Lam**
Ricoh Silicon Valley
2882 Sand Hill Road, Suite 115
Menlo Park CA 94025-7022
`{stork,lam}@rsv.ricoh.com`

Abstract

We describe the Open Mind Initiative, a framework for building intelligent systems collaboratively over the internet, and focus on one of its simpler component projects, Open Mind *Animals*. The Initiative extends traditional open source development methods by allowing non-expert netizens to contribute informal data over the internet. Such data is used to train classifiers or guide automatic inference systems, and thus it is important that only data of high accuracy and consistency be accepted. We identify a number of possible sources of poor data in *Animals* — several of which are generic and applicable to a range of open data collection projects — and implement a system of software modules for automatically and semi-automatically preventing poor data from being accepted. Our system, tested in a controlled laboratory intranet, filters faulty data through a variety of mechanisms and leads to accurate decision tree classifiers. Our reusable modules can be employed in our planned large-scale internet deployment of *Animals* and other Open Mind projects.

Introduction

Unlike many small demonstration and toy systems in artificial intelligence, virtually all real-world systems require large sets of knowledge or training data (Shortliffe 1976; Lenat 1995). Pattern recognition systems in particular rely quite heavily on large corpora of training data (Ho & Baird 1997; Jelinek 1998). In many cases such data can be collected from non-experts, for instance speech data or the labels of handwritten characters. The central hypothesis underlying the Open Mind Initiative is that for some classes of problems, such data can be collected from large number of non-expert "netizens" over the internet. Our focus in this paper is on a key problem in Open Mind: how to ensure that the data contributed in this way has an acceptably small number of errors and represents consensus of the participating netizen population.

The Open Mind Initiative

Briefly stated, the Open Mind Initiative is an internet based collaborative framework for developing intelligent software such as speech and handwriting recognizers, common sense reasoning systems, smart spam filters, and so on (Stork 1999a; 1999b; Valin & Stork 1999). The Initiative relies on three main forms of contributions: 1) *domain experts* provide fundamental algorithms such as learning algorithms for character recognition, 2) *infrastructure/tool developers* provide software infrastructure such as for capturing raw data and rewarding netizens, and 3) non-expert *netizens* contribute raw data over the internet.[1] There are many incentives for netizens to contribute, including public acknowledgement on the Open Mind website, altruism and inherent interest in artificial intelligence and particular project domains, pleasure from playing games (serving as interfaces), lotteries, gifts and coupons donated by corporations, and more (Stork 1999a).

In contradistinction with traditional data mining techniques (Fayyad *et al.* 1996), Open Mind affords interactive learning, such as learning with queries (Angluin 1988) in which the *most informative* or most useful information (given the state of the system) is requested from contributors. Thus learning is recast as a problem in decision theory in which each action (query) has an expected cost/benefit, and the optimal strategy is to present the query which when answered is expected to improve the classifier's accuracy the most (Pratt, Raiffa, & Schlaifer 1995; Thrun 1995). Learning with queries is generally faster than non-interactive learning based on randomly sampled data. Consider, for instance, the use of interactive learning for developing a recognizer of handwritten digits. The current classifier identifies the region in pattern space where images are ambiguous (e.g., the boundary between category "I" and "1"), and interactively queries contributors to provide labels of such ambiguous patterns. In this way the system "focusses in" on difficult or most informative patterns and increases the opportunity to resolve ambiguities in the data.

The software from separate Open Mind projects can be integrated, for instance by using a language identification or topic identification system to provide con-

[1] `www.OpenMind.org`

straints for handwritten OCR. All the resulting data and software are then available through an open source license and experts can propose changes.

The Initiative arose from a deep appreciation of the following facts and recent trends:

- The success and increasing acceptance of open source development methods and resulting software such as *Linux*, *emacs*, *Mozilla* and *Apache*.
- The realization that many problems in pattern recognition and intelligent systems require very large data sets that can be provided by non-experts.
- The refinement of well developed techniques in pattern recognition, machine learning, grammatical inference, data mining and closely related core disciplines.
- The increase in collaboration over the internet, and the improvement of tools for facilitating such collaboration, both among experts and non-experts.
- The growth in the participation of non-specialists (e.g., non-programmers) in group projects over the internet. Some of these allow participants to donate temporarily unused computer resources, as the Search for Extraterrestrial Intelligence,[2] Great Internet Mersenne Prime Search,[3] prime factorization,[4] and others.[5] In other collaborative projects, netizens contribute "informal" or non-expert knowledge, as in the Newhoo! open web directory[6] and MovieLens movie recommendation database.[7]
- The massive expansion of the web itself and particularly the growing number of non-specialist netizens on the web.

Table 1 summarizes some of the differences between traditional open source and Open Mind.

Current Open Mind projects

There are currently three full Open Mind projects under development. Open Mind speech recognition is initially addressing the recognition of isolated spoken *Linux* commands and speaker identification (Valin & Stork 1999). Open Mind common sense builds common sense ontologies and reasoning mechanisms; netizens contribute simple assertions (e.g., "a mother is older than her children"), ontology information ("all rabbits are animals"), as well as abstract inferencing rules. Open Mind handwriting is focussing on recognizing handwritten English words (Schomaker & Stork 2000). Figure 1 shows the web interface for netizen contributions in this project. Here the netizen's task is to segment individual characters in pixel images of unlabelled handwritten words collected previously with a dynamic pen tablet in a semi-controlled environment. The segmention information and character labels provided by netizens in this way are used to train multiclassifier based letter and full word recognizers. The existing data labelling system, written in Java, has been used successfully by experienced segmenters over an intranet and is being made more robust and user friendly for deployment on the World Wide Web.

[2]setiathome.ssl.berkeley.edu
[3]www.mersenne.org
[4]www.rsasecurity.com
[5]distributed.net
[6]www.dmoz.org
[7]www.movielens.umn.edu

Open Source	Open Mind
no netizens	netizens crucial
expert knowledge	informal knowledge
no machine learning	machine learning used
adaptive techniques used to facilitate navigation of contributed data (e.g., movie recommendations)	machine learning used to build a single classifier or AI system (e.g., speech recognizer)
most work is directly on the final software	most work is *not* on the final software
hacker/programmer culture ($\approx 10^5$ contributors to *Linux*)	netizen/business culture ($\approx 10^8$ members)
separate functions contributed (e.g., *Linux* device drivers)	single function goal (e.g., high OCR accuracy)

Table 1: Comparison of traditional open source and Open Mind approaches.

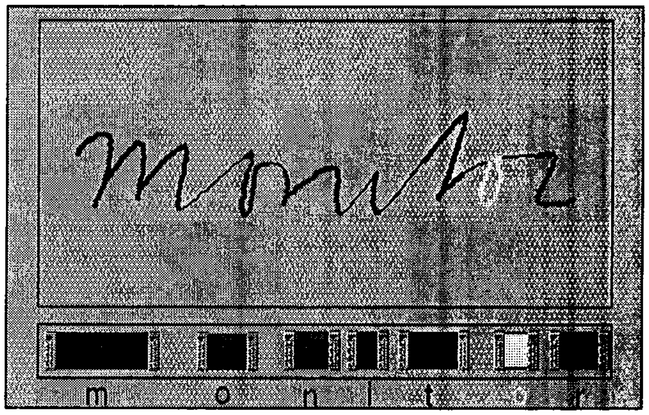

Figure 1: In an Open Mind project for recognizing handwriting words, netizens segment letters within a pixel image of a connected handwritten word from an unlabelled database of scanned or dynamically sampled handwritten words. Netizens use a mouse to indicate the beginning and end of each letter, and type the letter's label, all through a browser interface.

Open Mind *Animals*

To better understand general problems in infrastructure and data quality assurance in the Open Mind framework we have implemented a simple "20 questions" AI game called *Animals* (Shapiro 1982). During each session (game) a decision tree based on animal attributes is grown in the background. The game proceeds as follows. The netizen player thinks of a target animal and the system tries to guess this animal based on the player's answers to a sequence of yes/no questions; this sequence corresponds to a path through a binary decision tree (Fig. 2). The system ventures a guess of the target animal's identity according to the label at the leaf node reached this way. If this guess is correct, then the game is over and the player is encouraged to play the game again. If instead the guess is incorrect, then the system asks the player for the identity of the target animal and to submit a single yes/no question that distinguishes this target animal from the animal guessed by the system. The user is awarded a point for helping the system learn that new animal, as shown in the sample user session. In the actual implementation, answers to yes/no questions are entered by clicking on one of two buttons; animal names are entered by keyboard in an HTML form. The more the game is played by players having different knowledge, the larger will be the resulting tree data structure.

Sample *Animals* user session

Animals: Welcome to *Animals*, an Open Mind project. Think of an animal and I'll try guessing it by asking you a series of yes/no questions.
Does it have four legs?

Player: No

Animals: Does it have two legs?

Player: Yes

Animals: Does it lay eggs?

Player: No

Animals: Are you thinking of a human?

Player: No

Animals: Well, I guessed wrong. Please help me learn a new animal. Please give me the name of the animal you are thinking of.

Player: Gorilla

Animals: Please give me a yes/no question that will differentiate between a human and a gorilla.

Player: Does it have fur?

Animals: For a gorilla, what is the (yes or no) answer to your question above?

Player: Yes

Animals: Thanks. I'll remember that! You have been awarded an extra point. Would you like to play again?

The basic *Animals* procedure is shown below.

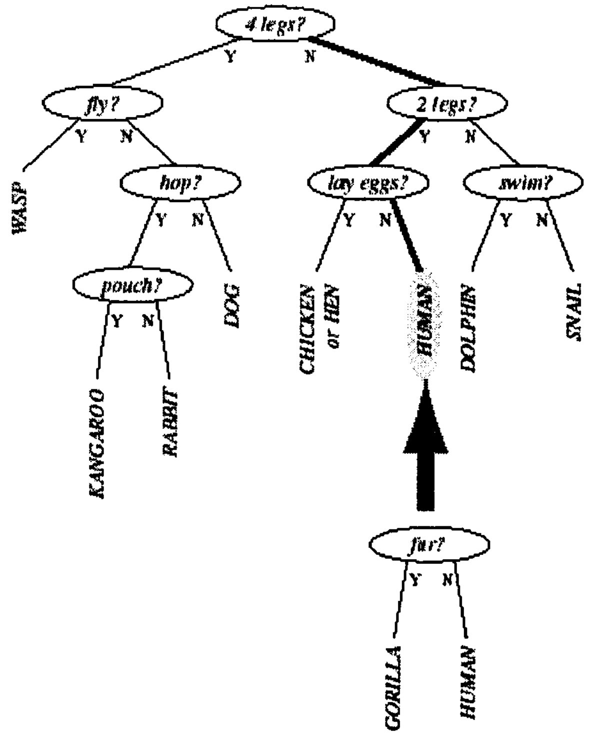

Figure 2: The current tree encodes eight leaf nodes. The player is thinking of the target animal *GORILLA*, and answers the questions corresponding to the heavy path; the system then guesses the animal at the associated leaf node, i.e., *HUMAN*. As this is an incorrect guess, the player is then asked to contribute a new question that distinguishes her target animal from the one guessed by the system. This new question, here *does it have fur?*, replaces the previous leaf node and the decision tree is expanded. Note that one of the leaf nodes contains synonyms (*CHICKEN* and *HEN*).

```
Procedure: Basic Animals
1. Initialize X to be the root node.
2. If X is a leaf node, then skip to step 6.
3. Ask the question at X.
4. If the player answers "yes," then assign the "left" child node to be the new X; otherwise assign the "right" child node to be the new X.
5. Go to step 2.
6. Ask player if she is thinking of the animal represented at X.
7. If she answers yes, then stop or else restart the game by going to step 1.
8. Ask player to give a question that distinguishes her target animal from the one guessed by the system; expand the tree using this information.
9. Award points to the user. Stop or else restart the game by going to step 1.
```

Ensuring data quality

The Basic *Animals* procedure above leads to accurate trees if and only if the data contributed by players is accurate. In the general Open Mind approach there are many netizen contributors, each having different expertise and reliability. Thus we now consider sources of data error and corresponding algorithms, implemented as software modules, for reducing the amount of faulty data accepted by the system. It should be stressed that the modules described here seek to prevent faulty data from being accepted, as is particularly appropriate when growing a hierarchical tree data structure. A number of traditional techniques for ensuring database integrity and accuracy can be used after data collection which can be applied to additional database structures (Parsaye & Chignell 1993).

Below we list sources of poor data in *Animals* and modular algorithmic approaches to reducing their effects on trees. Some of these problems were unanticipated and became apparent only in preliminary deployments and user tests.

invalid animal name A player might misspell an animal, e.g., *GORILA*. For that reason, every proposed animal (leaf node) is checked to see if it occurs in a lexicon of animals compiled semi-automatically from the web. (This "flat" lexicon did not include animal synonyms or hierarchical category information, see below.) If the proposed animal cannot be found in the lexicon, a warning message is displayed and the system requests the user to retype the animal. If the second submission also fails in this way, the game is halted and no data is accepted into the system.

incorrect item Suppose after a series of questions the system guesses an animal that the player feels does not correspond to her reply to the one of the questions. For instance, suppose the player answers "yes" to the question *four legs*? and the system nevertheless guesses *HUMAN*. The current player can, through a simple dialog interface, bring this questionable leaf node to the attention of the domain expert — someone who can independently check the quality of the data. The questionable animal name at the leaf node, and the list of questions and coded answers leading to it, are sent to the domain expert (or independent player), who serves as final arbiter. In the meantime, the questionable node is locked and cannot be modified.

inconsistency/non-uniqueness A special case of the previous source of error occurs when a player seeks to add an animal that occurs at another leaf, i.e., elsewhere in the tree. For example, suppose a player tries to contribute *DOG* at the node currently labelled *KANGAROO* in Fig. 2. This submission is inconsistent with the information about *DOG* previously submitted. To filter such inconsistent information, our module automatically brings to the attention of current netizen the lowest query node antecedent to both the nodes that would represent *DOG*. The question in this node, here *hop?*, was answered differently by two players. Our module automatically sends a warning message to the player seeking to add the conflicting information. If that player still seeks to submit this information, such information is accepted, but both conflicting leaf nodes are locked and a message is sent to a domain expert or independent player who acts as final arbiter.

submission collisions In deployments accessible by large numbers of netizens (such as on the World Wide Web), it becomes probable that two players will try to contribute different animals at a particular tree node simultaneously. To accommodate such colliding submissions, we have implemented a module that allows the first netizen to modify the tree node and locks the node from the second netizen for 30 seconds. During this time a warning notice is displayed to the second netizen.
We also implemented an alternate strategy in which the system recognizes that the first player has changed the leaf node between the time the second player is asked to add a new animal. In this case the system can trace down the modified tree to find the appropriate place to add the new animal. Suppose for example the first player has entered the information to differentiate cats from dogs and the second player has entered information to differentiate cows from dogs, and the first player's modifications reached the system first and has taken effect. Recognizing the situation when processing the second player's input, the system asks the second player the new questions in the decision tree (e.g., "Please tell me more about a cow. Does it bark?") until it reaches a leaf node again. The second player is then given the opportunity to teach the system about cows. If yet another collision occurs, the above process is repeated. While we expected this alternate strategy would lead to slightly faster data collection and larger trees, it proved burdensome and confusing to players.

synonyms Several animals have two or more names and we would like the tree to represent such information. Suppose the player's target animal is *HEN*, and the system guesses *CHICKEN*. This is, technically speaking, a correct guess. Nevertheless have implemented a module that allows the player to add a synonym to a leaf node, without splitting the leaf node, as shown in Fig. 2.

subset animal Suppose the player's target animal is *MANX* and that after answering questions in the tree, the system guesses *CAT*. Technically speaking, this answer is correct. Nevertheless, we would like to capture the information associated with the subcategory *MANX* the netizen has in mind. We have implemented a module that in this case the player can enter a question that distinguishes her target animal, *MANX*, from all other members of the category *CAT*. The resulting tree does not represent the fact that a *MANX* is a subset of *CAT*, however.

Software implementation

Our ultimate goal is to deploy all Open Mind projects, including *Animals*, on the world wide web, accessible to all netizens, regardless of browser type and with minimal burdens of plug-ins and Java. In part for this reason, processing for *Animals* is done on the server side using Java servelets. The *Animals* servelet interacts with each netizen by dynamically generating HTML pages. Since the HTTP protocol is inherently stateless, the servelet embeds state information within the FORM tags of each HTML page. The HTML code generated is simple, and does not rely on client side Javascript, Java, Macromedia Flash or other related technologies. The input to the *Animals* servelet is just the state information for the user and the player's action. Given that input, the *Animals* servelet will calculate the next state, updating its database of animals if necessary, and present to the user a new dynamically generated HTML page reflecting her new state.

A guiding principle for Open Mind projects is to make it easy for netizens to contribute. For this reason, players reply to questions by means of large buttons (yes or no). Moreover, the current sequence questions and the player's answers are updated and left on the screen. This makes it simpler for netizens to catch errors, as described in **incorrect item** and **inconsistency/non-uniqueness** items in the previous section.

Other server side software, for acknowledging netizen contributions, is written in Java under Windows NT, and is quite straightforward. Roughly 10% of our total code is for the basic *Animals* procedure, 10% for ensuring data quality, and 50% for I/O and display, and 20% for parsing commands, setting states, selecting modules and other infrastructure. While this distribution of code is skewed toward ensuring data quality because of the simplicity of the basic *Animals* procedure, we expect that the effort for ensuring data quality will represent a significant portion of the coding work in all such open data acquisition projects.

Results

We implemented *Animals* in a corporate research intranet of 22 users, representing a roughly equal mix of Netscape Navigator and Internet Explorer browsers. No effort was made to suggest browser settings such as fonts or styles and for that reason we developed a simple text-based interface. While there were no "hostile" players, each of the sources of data fault described above were present (sometimes by deliberate choice). In every case, our modules eliminated the faulty data according to the techniques described above. Manual inspection of resulting trees (containing over 100 animals) showed no faulty data. Informal interviews afterward revealed that participants found the interface and the game itself easy and fun.

Conclusions and future directions

Other components of *Animals* are being built as reusable codes for future Open Mind projects. We plan to expand the user identification component to include password protection and standardize it such that users can access all Open Mind projects using just one user name and password. We will also write more lively dialogs for the system in the near future.

There are several additional techniques for increasing the proportion of high-quality data contributed in open data aquisition. One method is to require prospective contributors to read an on-line tutorial, and to pass an on-line test before he or she can contribute. Not only would such a tutorial improve the general quality of the information provided, but the results of the on-line test would indicate the prospective contributor's reliability or level of expertise. The results of such tests could therefore control the level of difficulty or problems posed to each contributor, and could be used to automatically arbitrate between disagreeing contributors of differing reliabilities.

We stress that we chose *Animals* for the simplicity of the core tree growing method so as to allow us to focus on problems of infrastructure and ensuring data quality. Our core program and modules for ensuring data quality can be applied to many other domains, such as fruit, furniture, automobiles, and so on. Nevertheless, our ultimate challenge is to refine and expand these techniques to more sophisticated systems such as handwriting, common sense reasoning and so forth, and thus lessons learned from *Animals* will be invaluable first steps.

We feel that a number of trends, particularly the expansion of the internet and participation of non-experts in web projects, imply that the general area of open data aquisition will be increasingly important in the development of artificial intelligence and pattern recognition systems. Indeed, for problems such as very large common sense reasoning systems, it is hard to imagine a cost effective alternative to the Open Mind approach. These trends highlight the need for further research in algorithms for filtering and open data acquisition.

Acknowledgements

The authors extend warm thanks to David Bine for donating infrastructure code and to Yann Le Cun for suggesting *Animals* as a good arena for open data collection.

References

Angluin, D. 1988. Queries and concept learning. *Machine Learning* 2(4):319–342.

Fayyad, U. M.; Piatetsky-Shapiro, G.; Smyth, P.; and Uthurusamy, R., eds. 1996. *Advances in knowledge discovery and data mining.* Cambridge, MA: MIT Press.

Ho, T. K., and Baird, H. S. 1997. Large-scale simulation studies in pattern recognition. *IEEE Transactions* *on Pattern Analysis and Machine Intelligence* PAMI-19(10):1067–1079.

Jelinek, F. 1998. *Statistical Methods for Speech Recognition.* Cambridge, MA: MIT Press.

Lenat, D. B. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM* 38(11):33–38.

Parsaye, K., and Chignell, M. 1993. *Intelligent Database Tools and Applications: Hyperinformation access, data quality, visualization, automatic discovery.* New York, NY: Wiley.

Pratt, J. W.; Raiffa, H.; and Schlaifer, R. 1995. *Introduction to Statistical Decision Theory.* Cambridge, MA: MIT Press.

Schomaker, L., and Stork, D. G. 2000. Open Mind handwriting recognition. to be submitted.

Shapiro, S. C. 1982. Programming Project 1: Animal Program (20 Questions), "Introduction to Artificial Intelligence," Department of Computer Science, State University of New York at Buffalo.

Shortliffe, E. H. 1976. *Computer-Based Medical Consultations: MYCIN.* New York, NY: Elsevier/North-Holland.

Stork, D. G. 1999a. Document and character research in the Open Mind Initiative. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR99)*, 1–12.

Stork, D. G. 1999b. The Open Mind Initiative. *IEEE Intelligent Systems & their applications* 14(3):19–20.

Thrun, S. 1995. Exploration in active learning. In Arbib, M., ed., *The Handbook of Brain Theory and Neural Networks*, 381–384. Cambridge, MA: MIT Press.

Valin, J., and Stork, D. G. 1999. Open Mind speech recognition. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop (ASRU99).*