16-610: The KDD Lab
Assistive Intelligent Environments

Daniel Wilson
dwilson+@cs

March 12, 2001

Abstract

About 1 in 5 Americans have some kind of disability, and 1 in 10 have a severe disability [12]. With an aging population and a greater likelihood of disability with age, the presence of disabilities can only be expected to grow. The burgeoning field of intelligent environments can augment lost abilities, bringing many people closer to independent, increasingly productive lives. This technology can only succeed if it is usable through a natural interface. The most important interface is speech recognition. Current speech recognition systems are not accurate enough to fulfill the needs of people with disabilities. We propose using reinforcement learning to allow our speech recognition package to apply environmental information to automatically tailor its listening to specific users. An environment will be instrumented with motion detectors. Several language models will be generated, based upon differing tasks. The data from the detectors will be used by our system to decide which language model to use. We predict that over time the system will come to choose the best language model based upon sensor readings, and thereby achieve a higher recognition accuracy than is possible using a general language model. The speech recognition system will adapt itself to user habits and anticipate user needs, helping to make intelligent environments a realistic choice that could help millions of individuals with disabilities live more comfortably and independently.

1 Introduction

Although the most efficient form of interpersonal communication is often speech, it is usually impractical for use in a human-computer interface. Limitations in speech technology hurt the capabilities of intelligent environments, necessitating the use of unnatural and awkward speaking methods [25]. In order to bring the benefits of intelligent environments to those who most need them, the physically disabled, it is vital to improve this interaction.

Intelligent environments go by many names, including ubiquitous computing, pervasive computing, and smart environments. The idea is to use information from sensors embedded in the environment to augment human physical and cognitive abilities. The environment could perform physical tasks such as opening doors, manipulating lights, operating a telephone, reading a book, etc., It could also perform cognitive tasks such as reminding the user to take medicine or answering queries, i.e., "Is the front door locked?".

Our goal is to use the sensors in our intelligent environment to improve speech recognition in a manner that automatically adapts to user speech patterns. One such sensor is a pyroelectric motion detector. We suggest that by using a network of these detectors placed in each room of a house it will be possible for the system to associate certain detectors with corresponding speech patterns, i.e. spoken commands or queries. Our speech recognition software uses a language model to improve recognition accuracy. A language model is a statistical description of which words are said, and which words are said together. Recognition accuracy increases as the language model becomes more domain specific. We will use a reinforcement learning algorithm, using motion detector readings for state information and a selection of language models as actions.

2 Prior Work

Over the last several years much effort has been put into improving the construction of language models over simple trigram models, including higher-order n-grams, skipping [6, 7, 8], modified Kneser-Ney smoothing [8, 10], clustering [8, 9], and combinations of these techniques [5]. Intelligent environments have been applied to several domains, including automatic control of utilities such as heating [11], automatic control of mixed media such as DVD movies based upon user presence [13], facilitation of group meetings [14], and accident detection [22]. However, none of these environments have been used to improve language model selection.

Though intelligent environments remain an unexplored resource for automatic language model selection, there have been past efforts to use multimodal information to adaptively resolve language model selection. For instance, at the CMU Interactive Systems Laboratories the information gained from gaze tracking has been used to help a Janus III spoken language system decide when to swap language models [15]. Unfortunately, in order to benefit this approach requires a camera in every room and considerable computational power.

The problem has also been approached from a theoretical standpoint. Popovici and Baggia have attempted to instantiate automatic language model selection based upon dialogue predictions [16]. Also, IBM researchers have attempted to adapt trigram language models to different situations based upon the observed topic of conversation [19]. In the spontaneous, unstructured conversation of the home environment, however, these methods face serious challenges.

Many researchers have noted the significance of this problem and attempted to grapple with it. In the past intelligent environments have been criticized as solutions looking for problems [17, 18], but they offer concrete solutions to the problems faced by those with physical disabilities. By applying the information inherent in an intelligent environment to solving the problem of automatic language model selection, we can improve speech recognition accuracy and help solve problems that are clearly definable and clearly significant both to individuals and to America as a whole.

3 Past Work & Background

A preliminary survey was conducted during the fall of 2000 [26]. Twenty five surveys were collected through personal interviews with individuals suffering from paraplegia (paralyzed from the waist down), quadroplegia (paralyzed from the neck down), and disabilities related to advanced age. The survey revealed a trend. Speech recognition systems connected to home automation devices existed in the homes of three surveyed quadroplegic individuals. Speech recognition on home computers only existed in the homes of eight surveyed individuals. Speech recognition software was in common use in none of these homes. The survey results reflected a pessimistic view towards the real world utility of speech recognition systems. When prompted about this discrepancy participants responded with varying takes on the same idea: continuous speech recognition is frustratingly inaccurate, and discrete recognition (speaking one word at a time) is too unnatural.

A home environment has been outfitted with an X10 home automation system [28]. X10 is a communications language that allows compatible devices to communicate via existing 110V electrical wiring. These devices may also be accessed by wireless RF signals. Such devices include light switches, motion detectors, thermostats, wall sockets, contact switches, cameras, etc., Many of these devices have 2-way communication, i.e. they can both accept commands and report status. These devices are well tested, widely available, and inexpensive. Currently the environment uses only automated light switches and lamp switches. These devices are accessed through a serial connection, using a transmitter module controlled by Java code. Our X10 system interfaces through Java code to a Sphinx II speech recognition system [3]. Sphinx is a speaker independent continuous speech recognition system that manipulates our X10 system in response to verbal commands through a wireless microphone.

Statistical language models (LM) improve the performance of speech recognition systems by providing estimates of a priori probabilities of word sequences. Our Sphinx tri-gram language models are built automatically using LMTool [20], from a sentence corpus that contains examples of what might be said. For instance, the CMU Communicator airline reservation system language model was trained using a corpus of dialogue between travel agents and customers [4].

Our Sphinx system uses a dialogue management module also developed by the CMU Speech group [4]. This dialogue manager classifies each utterance into different categories based upon a pre-written grammar. The grammar is a collection of possible word combinations and their classification. For instance, the following might be a grammar for a "turn on the lights" command/classification:

please TURN on *the LIGHT *please
TURN: turn, flip, switch
LIGHT: light, lights, lamp

The asterix denotes that the word is optional. A capital word denotes a word that has synonyms which are later enumerated. This small grammar could understand, "Turn on light", and "Please switch on the lights," and everything in between. In this manner utterances are categorized and the commands or queries may be answered appropriately by the system. These grammars must be hand-written, but the notation is flexible and powerful, and our domain limited enough to make this feasible.

Table 1: Command Categories & Sample Effects

Light Manipulation
Turn on and off lights
Dim or Brighten lights
Turn on lights when motion detectors trigger
Infra-Red Manipulation
Switch television channels
Turn volume up or down
Play music or watch videos
Query/Scheduling
Set reminders for medicine, bedtimes, meetings, etc.,
Answer questions such as current time, date, weather, etc.,
Read online text sources such as newspapers, etc.,

4 Approach

We will write a grammar that categorizes spoken commands pertaining to three main categories: light manipulation, infra-red commands, and query/scheduling commands. The commands will be structured in the most natural manner possible, based upon observed interactions between our target audience and their assistants and care givers [26]. Some possible command domains are included in table 1.

We will require a sentence corpus in order to generate language models. The structure of our grammar defines which spoken commands can be understood. Language models will be generated using sentence corpora generated directly from the structure of our grammar [23]. Future work will include gathering utterance data from the instrumented home environment to be used in upgrades of the sentence corpora. This method will be used to generate many language models. We will begin by training three intermediate language models using only information (sentences) from the three categories (lights, IR, & query/scheduling). These language models are useless as final products, because they will not recognize commands outside their specific category. However, by interpolating these language models we may assign different weights to each category [20]. We will begin by creating a general LM from all three initial LMs, with equal weights. Then we will create three LMs which have more weight upon each of the three categories. Finally, we will create three more LMs with equal weight between two categories and one category of lesser weight. Exact weight distributions will be determined through experimentation. We will ultimately have seven language models to choose from, each with a unique distribution.

We need to supply the Sphinx system with user location information. We will install six motion detectors, located in the kitchen, living room, study, bath-

room, bedroom, and hallway. The wireless motion detectors require both heat and motion to trigger and will therefore be less susceptible to false detections. Once triggered the detectors pause for ten seconds before taking new readings. Motion detector readings will be logged automatically by a Java interface.

At this point we have shown how to generate our language models and how to retrieve our motion detector information. We will now explain how to use these as actions and states (respectively) in a reinforcement learning problem.

Reinforcement learning is an unsupervised learning technique that explores how to map from situations to actions while maximizing a scalar reward. In our task the learner is initially ignorant and is never given a 'correct answer'. The learner must discover through exploration which actions yield the highest reward. The 'situations' that we speak of are described to the system by many contexts, primarily the status of many motion detectors and the time. Any combination of all or none of the motion detectors may be triggered at any time. The possible actions to take correspond to the language models which may be active at any time.

In our situation the actions have no impact upon the state transitions. Our agent's problem then is to choose actions that maximize immediate reward as a function of the current state. This class of problems is called *associative* reinforcement learning [RLasurvey]. Also, in our problem the state space is large and there are relatively few rewards. Therefore, we must have some way of modeling the world, or it will take an impossible amount of real time to learn anything significant.

We will start by defining our state space. Our information is derived from motion detector information, however, we would like to limit our state size while maximizing the amount of relevant information. Therefore we will model our state as three sets of six motion detector counts. The first set will be a direct counting of the number of times each motion detector triggered in the last minute. The next set will count for the last three minutes. The last set will count for the last five minutes. Formally:
$s \in S$, where $S = \{(one_1 ... one_6), (two_1 ... two_6), (three_1 ... three_6)\}$
In this manner we are able to capture and evaluate occurrences with priority given to the most recent. This method also allows use of a simple euclidean metric to determine the difference between two states.

Actions will be stored as a collection of probabilities that sum to 1. Each

action will initialize to $\frac{1}{|A|}$. An action is rewarded by gaining a percentage subtracted from other actions. Punishment redistributes a percentage taken from the offending action to other actions. These redistributions will be weighted by existing probabilities. In this manner redistributed probability from a punished action will be added mostly to actions with higher probabilities. Exploration will happen implicitly as actions are chosen randomly, but weighted by probabilities.

When an untraversed state is encountered (which should happen often) we must have a mechanism of generalizing from previous experience. Our model will be generated using a k-means algorithm. When a new state is encountered the action probabilities will be initialized as a combination of the probabilities of other previously encountered states, weighted by euclidean distance. When this new state is rewarded/punished, the results will be used to update existing states. These states and their probabilities will be stored in a lookup table. This is feasible in our situation because learning depends upon real-time utterances, which occur relatively infrequently.

5 Evaluation

When the X10 system and the learner are fully constructed we will begin gathering results. The system will exist in an authentic home environment containing two male college students and a pet cat. The students will use the capabilities of the environment on a regular basis through wireless and stationary microphones. The Sphinx system automatically logs all information included in every interaction. After one month we will review the logs to determine results. We will use two measures to gauge our success. First, it is important to see how often the RL algorithm chooses the correct language model, despite any increases in actual accuracy. This is the most important result, because if the RL algorithm can make this decision successfully, then actual accuracy results may be improved with more work on creating accurate language models. Second, we must determine any percentage increase in recognition accuracy, provided by the more accurate language models. If the system is listening with the most specific language model possible, then recognition accuracy should rise correspondingly.

6 Work Plan

This project must be complete by the start of May 2001. During February we instrument the home environment with X10 motion detectors. We will also add functionality for these detectors to the current Java interface. We will create a

grammar and use it to create several corpora. We will create initial language models and use interpolation to create final language models. Work on the reinforcement learner will begin in March and continue for the entire month. Starting in April one month will be spent logging results. At the end of April and beginning of May the results will be collected and included in a paper.

7 Conclusion

A significant number of people with disabilities could be helped by intelligent environments. This will require continuous speech recognition at a more accurate level than is currently available. We propose a method of language model selection that could improve speech recognition accuracy. A reinforcement learner will use information gathered from motion detectors to determine which language model to activate. We suggest that over time the learner will come to select the most appropriate language model based upon user location. We hope to use the multimodal capabilities inherent in our intelligent environment to improve speech recognition accuracy and help make intelligent environments more feasible for those with disabilities.

8 References

[1] Ronald Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. Computer Speech and Language, 10:187-228, 1996.

[2] Y. Gotoh and S. Renals. Variable word rate N-grams. IEEE, 1591-1594, 04/2000.

[3] Ward, W. and Issar, S. Recent improvements in the CMU spoken language understanding system. In Proceedings of the ARPA Human Language Technology Workshop, March 1994, 213-216.

[4] Rudnicky, A.I., Thayer, E., Constantinides, P., Tchou, C., Shern, R., Lenzo, K., Xu W., Oh, A. Creating natural dialogs in the Carnegie Mellon Communicator system. Proceedings of Eurospeech, 1999, 4, 1531-1534.

[5] Goodman, J. Putting it all together: language model combination. IEEE, 1647-1650, 04/2000.

[6] R. Rosenfeld. Adaptive statistical language modeling: a maximum entropy approach. PhD thesis, Carnegie Mellon University, April 1994.

[7] X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, K.-F. Lee, and R. Rosenfeld. The SPHINX-II speech recognition system: An overview. Computer, Speech, and Language, 2:137-148, 1993.

[8] H. Ney, U.Essen, and R. Kneser. On structuring probabilistic dependences in stochastic language modeling. Computer, Speech, and Language, 8:1-38, 1994.

[9] P. F. Brown, V. J. DellaPietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. Computational Linguistics, 18(4):467-479, December 1992.

[10] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Computer Speech and Language, 13:359-394, October 1999.

[11] Mozer, M. C. The neural network house: An environment that adapts to its inhabitants. In M. Coen (Ed.), Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments (pp. 110-114). Menlo, Park, CA: AAAI Press. 1998.

[12] McNeil, J. Disabilities affect one-fifth of all Americans: Proportion could increase in coming decades. U.S. Census Brief. 1997.

[13] Brumitt, B., Meyers, B., Krumm, J., Kern, A., and Shafer, S. EasyLiving: Technologies for Intelligent Environments. Handheld and Ubiquitous Computing, September 2000.

[14] Cohen, M. The future of human-computer interaction or how I learned to stop worrying and love my intelligent room. IEEE Intelligent Systems. March/April 1999.

[15] Stiefelhagen, R. and Yang, J. Gaze tracking for multimodal human-computer interaction. IEEE. 2617-2620. 1997.

[16] Popovici, C. and Baggia, P. Specialized Language Models Using Dialogue Predictions. IEEE. 815-818. 1997.

[17] Huang, A., Ling, B., Ponekanti, S., and Fox, A. Pervasive Computing: What Is It Good For? Communications of the ACM. 84-91. 5/1999.

[18] Araya, A. Questioning Ubiquitous Computing. ACM Transactions on Computer-Human Interaction. 230-237. 5/1995.

[19] Rao, P., Monkowski, M., and Roukos, S. Language Model Adaptation Via Minimum Discrimination Information. IEEE. 161-164. 5/1995.

[20] Clarkson, P. and Rosenfeld, R. Statistical Language Modeling Using the CMU-Cambridge Toolkit. From Proceedings ESCA Eurospeech. 1997.

[21] Song, F. and Croft, B. A General Language Model for Information Retreival. ACM. 1999.

[22] Abowd, G., Atkeson, C., Bobick, A., Essa, I., MacIntyre, B., Mynatt, E., and Starner, T. Living Laboratories: The Future Computing Environments Group at the Georgia Institute of Technology. Proceedings of the 2000 Conference on Human Factors in Computing Systems (CHI 2000), The Hague, Netherlands, April 1-6, 2000.

[23] Kellner, A. Initial Language Models for Spoken Dialogue Systems. IEEE. 6/1998

[24] Sutton, R. Reinforcement Learning. Kluwer Academic Publishers, Boston/Dordrecht/London, 1992.

[25] Abowd, G. and Mynatt, E. Charting Past, Present, and Future Research in Ubiquitous Computing. ACM Transactions on Computer-Human Interaction, Vol. 7, No. 1, Pages 29-58, March 2000.

[26] Wilson, D. Preliminary Survey of Murdoch Villa. Carnegie Mellon University Assistive Intelligent Environments Project. 2000.

[27] Kaelbling, L. P. and Littman M. L. and Moore A.W. Reinforcement learning: A survey. Journal of Artificial Intelligence Research, 4:237–285, 1996.

[28] SmartHome. Networked home products using X10 technology. http://www.smarthome.com/.