Learning Text Filtering Preferences

Anandeep S. Pannu and Katia Sycara
The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
pannu+@cs.cmu.edu, katia@cs.cmu.edu *

Abstract

We describe a reusable agent that learns a model of the user's research interests for filtering conference announcements and request for proposals (RFPs) from the Web. For this task, there is a large volume of irrelevant documents and the proportion of relevant documents is very small. It is also critical that the agent not misclassify relevant documents. Information Retrieval and Neural Network techniques were utilized to learn the model of user's preferences. Learning was boot-strapped using papers and proposals the user had written as positive examples. The agent's performance at startup is quite high. Information retrieval and Neural Nets were used to train the agent and experimental performance results were obtained and reported.

Introduction

We describe a reusable Learning Personal Agent (LPA) that learns a user's research interests and filters Web-based conference announcements and RFPs. Learning Personal Agents have been used for information filtering from the WWW (Lang 1995), (Armstrong *et al.* 1995), (Pazzani, Nguyen, & Mantik 1995). In contrast to the environments of these systems where the probability of finding relevant links or documents is relatively high, so that the agent's task is to avoid user information overload, in our environment the probability of finding relevant documents is very small and the cost of missing a relevant document is very high.

Our goal is to put in the hands of CMU faculty an LPA to retrieve conference announcements and requests for proposals based on their research interests. Each LPA uses the same learning methods and interface but learns a different user model. We have experimented (see Section 6) with the LPA of one user and we are currently setting up the LPA's of other users.

The News Information agent (Figure 1) polls the Commerce Business Daily (CBD) and conference announcement newsgroups for arrival of new items. When a new item arrives the News agent forwards it

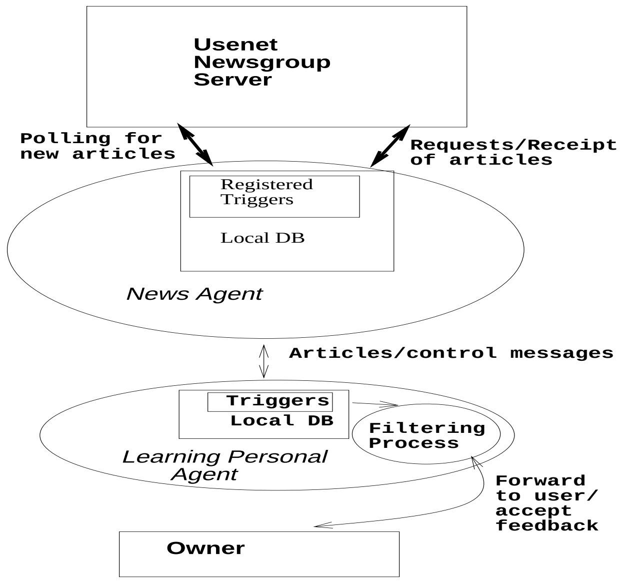

Figure 1: Learning Personal Agent and Environment

to all the LPAs (owned by different individuals) that have *registered* with it. Upon receipt of the new article, each LPA activates itself and runs the text of the article through a learned filter of its owner's research interests; it notifies its owner of relevant articles, i.e. those that pass through the filter.

In designing the agent some explicit design criteria were kept in mind. We needed to avoid excessive set up costs in terms of user time. The agent should have the capability to track evolving user preferences. Finally the agent should not be computationally expensive when functioning autonomously, delaying computationally expensive tasks for times when system load is low.

To comply with the requirements, learning was boot-strapped off-line using papers and proposals that a user has written as positive examples and documents written by other faculty with different research interests as negative examples. In this way, the agent acquired a high degree of competence at startup. This compe-

*This research was supported by ARPA contract F33615-93-1-1330 tence was increased (see section on experimental results) through additional training using user feedback during agent operation.

Using Information Retrieval Based Filtering

One of the techniques adopted for learning a user preference model was a standard well tested technique from Information Retrieval called term frequency-inverse document frequency weighting (tf-idf) (Salton 1989). This is a simple technique that is hard to beat (Lang 1994) and provides a well tested benchmark to which other learning models can be compared. Assume some dictionary vector $\vec{D}$ where each element d_i is a word. Each document then has a vector representation $\vec{V}$, where element v_i is the weight of the word d_i for that document. If the document does not contain d_i then $v_i = 0$. The term *feature* refers to an element of the vector V for a document and so is denoted by v_i. The more times a word t occurs in a document d the more likely it is that t is related to the topic of d. The number of times t occurs throughout *all* documents is called the *document frequency* of t or df_t. The larger df_t the worse t discriminates between documents. So for a given document, the relevance of the document based on a term is directly proportional to $tf_{t,d}$ (the number of times word t occurs in document d) and inversely proportional to df_t. A weight for each word encountered in a collection of documents isl allocated as follows:

$w(t,d) = tf_{t,d} log(|N|/df_t)$

where N is the entire set of documents. The classification power of a word in a document is indicated by this weight. Each element v_i of the vector $\vec{V}$ representing a document, contains the tf-idf weight calculated as shown above for the word d_i from the dictionary $\vec{D}$. The similarity or dissimilarity of two documents can be measured by using the cosine angle between two vectors representing the two documents. To learn a user profile a set of documents intended for training is converted into vectors and each document's classification noted. For each vector $\vec{V_i}$ classified in the same category each corresponding individual element v_{ij} is summed up. The average of each vector element is then taken forming a *prototype vector* for that category. When a new document is to be classified the "similarity" or "dissimilarity" of the tf-idf vector representation of the new document to each prototype vector is calculated, using the cosine angle measure.

To cut down vector dimensionality, we used two heuristics. First, we used an *information based* approach where we threw out from the dictionary vector $\vec{D}$ the k most common words and used only the n top ranking vector elements in the document vector in terms of tf-idf weight. The values of k and n for best performance were determined empirically.k values tested were 10, 20, 50, 75 and 100. n took the values 200, 1000 and 2000. Second, we used a *keyword* approach where the dictionary vector consists of 360 keywords gleaned from the Faculty Research Interests Database at CMU.

Neural Network Based Filtering

As an alternative we used a Neural Network based technique to learn a model of user preferences. The learning process used the same document vector as the tf-idf process described above. The network was trained using the same positive and negative examples as the tf-idf technique. The dimensionality of the document vectors was reduced using the keyword approach. For the Neural Network each v_i in a document vector $\vec{V}$ could either be a 1 (indicating the presence of the word in the document) or a 0 (indicating the word's absence).

Experimental Setup and Results

Filtering through Information Retrieval

We tested both variants, information based and keyword based, of the tf-idf technique. We used 178 negative and 179 positive examples for "bootstrap" off-line training. This training set was used to form the *prototype vector* for each category. The test set consisted of 50 documents from the CBD and conference announcements newsgroups. After a document was classified by the system, user feedback was solicited. The user classified the document and the document was added to the training set. Each time user feedback had been obtained from 10 additional documents, retraining was performed off-line with the augmented training set.

The classification accuracy of the technique was quite good. The initial performance of both tf-idf variants was high (around 85%) despite the fact that the initial training examples were chosen from a biased sample consisting of user supplied documents that were not drawn from the same population as the test set. This initial 85% performance increased through incorporating user feedback during system operation. So, the LPA starts with a high competence acquired from a biased training set but can rid itself of the bias as the user gives feedback during regular operation of the agent.

A surprising result was that the information based approach performed *on average* better than the keyword based approach despite the greater domain knowledge incorporated into the latter approach; however the keyword approach performed better on selecting *relevant* documents.

The Neural Network based Filtering Technique

The NN was trained using the same set of training and test sets as the IR based methods and using the same performance measures to test its performance. The document vectors in the training set were presented

Figure 2:

to the network at the input nodes after being converted to a Boolean zero-one feature vector. Each input node represent an element v_i of the document vector $\vec{V}$. The output of the network was compared to the representation of the relevance of the document called the *expected output* (1 for a relevant example and -1 for an irrelevant example). The error between the expected output for the document and the forward propagated output was then back-propagated for a number of *epochs*. In case the network output was greater than 0.5 for an expected value of 1, or less than 0.5 for an expected value of -1 the document was marked as being classified correctly. Otherwise the document was marked as being classified incorrectly. We used a *coordination set* (see (Pannu & Sycara 1996)) to determine the number of epochs to train the Neural Net for optimal performance.

The results reported are for a 360 word vector of keywords, and training time was roughly comparable to the tf-idf algorithm using the keyword based approach. We also experimented with different length vectors. The training time for tf-idf increased for longer length vectors. The vector length did not have a significant impact on the training time for the NN.

Figure 2 presents the overall experimental results. As can be seen from the figure the information based tf-idf approach has the highest overall performance. However keyword tf-idf performance is the best when we consider classification of only relevant documents. The Neural Net performed the worst, though our NN performance is considerably better than Neural Net performance reported by others (e.g. (Lang 1994)). Despite the small number of training examples used (357) compared to those reported in other literature (for instance (Lang 1994) which used 20000 examples), we were able to get performance approaching 70% correct classification. For the Cascade 2 neural network growing algorithm trained on a 1000 articles, Lang (Lang 1994) reported 25% correct classification. In our case the Neural Network has approximately 3 times the performance achieved by the Cascade 2 algorithm. This is no doubt due to the fact that our text filtering task is more specialized than the task for (Lang 1994). A trend seen in Figure 2 is that classification accuracy increases with user feedback. Though the keyword based approach shows a small decrease in *overall* classification accuracy, classification accuracy for relevant documents alone follows an increasing trend.

Conclusions and future work

We described a Learning Personal Assistant that learns a model of the user's research preferences and notifies the user when relevant conference announcements and request for proposals show up on corresponding Web newsgroups. Information Retrieval and Neural Net techniques were used for learning. The results reported in this paper are preliminary. The two approaches used here are both good candidates for the task of creating a user profile to filter documents. The Information Retrieval based approach is the most promising. The Neural Network based approach uses an extremely simple representation for a document (a Boolean zero-one vector). The results also show that it is possible to use a biased set of training examples, in order to get high initial performance and yet get acceptable performance when classifying documents that are not part of the biased set. Currently we are setting up experiments to collect data from additional users.

References

Armstrong, R.; Freitag, D.; Joachims, T.; and Mitchell, T. 1995. Webwatcher: A learning apprentice for the world wide web. In *Proceedings of AAAI Spring Symposium on Information Gathering from Heterogenous Distributed Environments.*

Lang, K. 1994. Newsweeder: An adaptive multi-user text filter. Research Summary, Carnegie Mellon University.

Lang, K. 1995. Learning to filter netnews. In *Proceedings of the Machine Learning Conference 1995.*

Pannu, A., and Sycara, K. 1996. A learning personal agent for text filtering and notification. Submitted to AAAI 96.

Pazzani, M.; Nguyen, L.; and Mantik, S. 1995. Learning from hotlists and coldlists : Towards a www information filtering and seeking agent. In *Proceedings of the Seventh International IEEE conference on Tools with AI.* IEEE Computer Press.

Salton, G. 1989. *Automatic Text Processing. The Transformation, Analysis and Retrieval of Information by Computer.* Addison-Wesley.