A Naive Statistics Method for Electronic Program Guide Recommendation System

Jin An Xu and Kenji Araki

Graduate School of Information Science and Technology,
Hokkaido University, Kita 14 Nishi 9, Kita-ku,
Sapporo 060-0814, Japan
{xja, araki}@media.eng.hokudai.ac.jp

Abstract. In this paper, we propose a naive statistics method for constructing a personalized recommendation system for the Electronic Program Guide (EPG). The idea is based on a primitive approach of N-gram to acquire nouns and compound nouns as prediction features, and then to combine the $tf \cdot idf$ weighting to predict user favorite programs. Our approach unified feedback process, system can incrementally update the vector of extracted features and their scores. It was proved that our system has good accuracy and dynamically adaptive capability.

1 Introduction

Today, a number of technical developments, such as satellite, cable and digital TV technology have resulted in an increasing number of available TV channels, hundreds of channels broadcast thousands of TV programs everyday. The challenge, how to offer a convenient and intelligent user interface, has become a research point.

There have been several research projects around EPG recommendation system [1,2,3,4]. In our previous work, an approach using Inductive Learning with N-gram to predict user's habits and preferences, showed good dynamically adaptive capability in small data sets [5].

In this paper, we propose a naive statistics method for constructing a personalized recommendation system for EPG. The idea is based on a primitive approach of N-gram to acquire nouns and compound nouns as prediction features, and then to combine the $tf \cdot idf$ weighting to predict user favorite programs, and then to unified feedback process. The objective is to develop a good intelligent user interface between each TV fan and his/her digital television.

This paper includes three sections as follows: Presentation of our system architecture, evaluation of the performance of the present system and a summary of this work.

2 Outline

This section describes the procedure of our proposed method as shown in Figure 1, our system consists of term extraction process, prediction process, user feedback process and IEPG extraction process.

Y. Hao et al. (Eds.): CIS 2005, Part I, LNAI 3801, pp. 434–441, 2005.

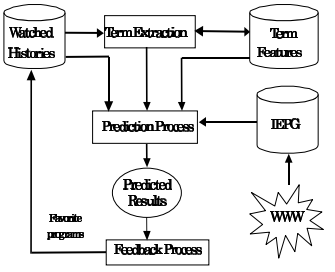

Fig. 1. Outline of the Procedure

2.1 Term Extraction

Our idea of term extraction is to use bigrams. It is based on the algorithm of Nakagawa [6] et al., which considered both the score using simple noun bigrams as components of compound nouns to calculate score of compound nouns, and the nouns independent frequency to calculate the score of each extracted term. These scoring methods content of four steps:

First, acquiring of simple noun bigrams including their frequencies shown in Figure 2 where [N M] means bigram of noun N and M.

In Figure 2, [LN_i N] (i=1,..,n) and [N RN_j] (j=1,...,m) are single-noun bigrams which constitute (parts of) compound nouns.$\#L_i$ and $\#R_j$ (i=1,..,n and j=1,..,m) mean the frequency of the bigram [LN_i N] and [N RN_j] respectively. Note that since we depict only bigrams, compound nouns like [LN_i N RN_j] which contains [LN_i N] and/or [N RN_j] as their parts might actually occur in a corpus. Again this noun trigram might be a part of longer compound nouns.

Second, inassigning the direct score of a noun, since a scoring function based on [LN_i N] or [N RN_j] could have an infinite number of variations, the following sample, yet is presentative, scoring functions are considered.

#LDN(N) and #RDN(N): The number of distinct single-nouns which directly precede or succeed N. There are exactly "n" and "m" in Figure 2.

LN(N,k) and RN(N,k): The general functions that take into account the number of occurrences of each noun bigram like [LN_i N] and [N RN_j] are defined as follows.

$$LN(N,k)=\sum_{i=1}^{\#LDN(N)}(\#Li)^k \tag{1}$$

$$RN(N,k)=\sum_{j=1}^{\#RDN(N)}(\#Rj)^k \tag{2}$$

We can find various functions by varying the parameter **k** of (1) and (2). For instance, #LDN(N) and #RDN(N) can be defined as LN(N,0) and RN(N,0).

[LN_1 N] ($\#L_1$)	[N RN_1] ($\#R_1$)
[LN_2 N] ($\#L_2$)	[N RN_2] ($\#R_2$)
...	...
[LN_n N] (L_n)	[N RN_m] ($\#R_m$)

Fig. 2. Noun bigram and their Frequency

LN(N,1) and RN(N,1) are the frequencies of nouns that directly precede or succeed N.

Third, scoring the importance of compound nouns, it means to extend the scoring functions of a single-noun to the scoring functions of a compound noun. We adopt a very simple method, namely a geometric mean. Now think about a compound noun: $CN = N_1 N_2 \ldots N_L$. Then a geometric mean: GM of CN is defined as follows.

$$GM(CN,k)=\left(\prod_{i=1}^{L}(LN(N_i,k)+1)(RN(N_i,k)+1)\right)^{1/2L} \tag{3}$$

And then, combining the score of compound nouns and frequency of nouns, if a single-noun or a compound noun occurs independently, the score of the noun is multiplied by the number of its independent occurrences. Then GM(CN,k) of the formula (3) is revised. This new GM is called FGM(CN,k) and define it as follows.

if N occurs independently then

$$FGM(CN,k)=GM(CN,k)\times f(CN), \tag{4}$$

where f(CN) means the number of independent occurrences of noun CN.

Finally, normalizing the credibility metric between 0 and 1 for each extracted term. These terms are extracted from programmes watched over time and from programmes available to the viewer, and a recommender system incrementally updates the scores of terms extracted from programmes viewed. In our system, we just focus on Japanese nouns and unknown words as extraction targets.

2.2 *tf · idf*

tf · idf ranking is used for searches. It is a way of weighting the relevance of a term to a document. For a vector search in a set of documents for example, a document vector over all known terms is calculated for every document with *tf · idf* ranking. The correlations between this vectors and a query vector are then used to weight the documents according to a query [7].

The ranking takes two ideas into account for the weighting. The Term Frequency (*tf*) in the given document and the Inverse Document Frequency (*idf*) of the term in the whole documents. The *tf* in the given document shows how important the term is in this document. The document frequency of the term shows how generaly important the term is. A high weight in a *tf · idf* ranking scheme is therefore reached by a high term frequency in the given document and a low document frequency of the term in the whole documents.

In this paper, we take each a program as a document. The *tf · idf* is defined as follows: Let d be a program, let w_i be the ith word in d. The *tf* of w_i,$tf(w_i, d)$, is the number of times w_i occurs in d. The program frequency of w_i, $df(w_i)$, is the number of program in which w_i occurs at least once. The inverse program frequency of w_i, $idf(w_i)$, is defined as follows:

$$idf(w_i) = \log \frac{|N|}{df\, w_i} \tag{5}$$

```
'ドラマ=> Drama', '映画=> Movies', 'スポーツ =>Spor
'演劇 => Drama', '音楽=>Music', 'バラエティ=>Talk
'ニュース・報道=> News', '趣味・実用=>Home/How-to'
'ドキュメンタリー・教養=>Documentary', 'キッズ=>Kids
'アニメ・特撮=> Animation', '教育=>Educational'
'情報=> Home/How-to', 'その他 => Etc.'
```

Fig. 3. Categories of Japanese TV Programs

where $|N|$ is the total number of programs. Then, the $tf \cdot idf$ of wi is defined as $tf(w_i, d) \cdot idf(w_i)$. In this paper, we use the normalize $tf \cdot idf$, shown as follows:

$$d_{ik} = \frac{tf_{ik} \times \log \frac{|N|}{df\, w_i}}{\sqrt{\sum_{j=1}^{N} (tf_{ik} \times \log \frac{|N|}{df\, w_j})^2}} \tag{6}$$

2.3 IEPG Extraction

The Internet is a globally distributed dynamic information repository that contains vast amount of digitized information, and more and more such information now available in multimedia forms.

We extract TV programs to XML format as our IEPG database from internet using XMLTV module [8], from http://www.ontvjapan.com/program/. All of the extracted IEPG data is classified automatically according to the Concurrent Versions System (CVS) of Japanese TV programs. The categories are shown in Figure 3.

IEPG data vector consists of start time, stop time, channel, title, description, Japanese category, English category, etc.. In our system, we changed some contents of IEPG, such as time, day of week and channel names and so on.

2.4 Prediction

The basic idea of prediction is to use extracted term features and $tf \cdot idf$ to evaluate each new TV program. The terms were extracted from user watched programs. These histories have the same format as the IEPG. They are classified according to its category, as shown in Figure 3. Using these categories, we can acquire classified text for each category, and then extract term features for each category. We assume acquired terms to be a vector of each category as follows:

$$F_i = (C_k, T_i, S_i, F_i), \tag{7}$$

where F_i is the ith term features vector, C_k is the kth category, T_i is the ith extracted terms, S_i is the score of the ith term, F_i means the frequency of the ith term.

The credibility of each new TV program for prediction process was calculated as follows:

(1) Acquire all categories feedback tasks from watched programs to an array as a Categories Array (CA), re-ordering them according to the frequency of our feedback strategy (see next section).

(2) For each CA, acquire the category terms vector from term features to an array as a Terms Array (TA).

(3) For each CA, extract all new programs to another array as a Programs Array (PA).

(4) For the ith element PA_i of PA, extract the element TA_i if it matchs PA_i, push TA_i to an array Q. let Q be the query of using $tf \cdot idf$, to caculate the total $tf \cdot idf$ score ($S_{PA_i}^{tfidf}$) of PA_i for query Q according to the Eq. 5 and Eq. 6, the N of the Eq. 5 is the number of PA.

(5) For the PA_i, calculate the credibility using the score of the query Q according to Eq.8, when the term of Q_k matches current evaluated PA program, reorder the evaluated PA according to the credibility.

$$S_{PA_i}^{bigram} = \sum_{k=1}^{max} Q_k^{S_k} \tag{8}$$

(6) The Score of the PA_i is caculate as follows.

$$S_{PA_i}^{Score} = (S_{PA_i}^{tfidf} \cdot S_{PA_i}^{bigram})/(S_{PA_i}^{tfidf} + S_{PA_i}^{bigram}) \tag{9}$$

(7) Acquire all of the score of new programs to an array as a Result Array(RA).

(8) Sort RA according to the score, day of week etc..

(9) Output results to user, we can give some thresholds to control the number of recommendations.

2.5 Feedback

For realizing the dynamically adaptive capability, we have taken into account two factors that cause the dynamics of personalization.

The first step is to split the watched programs by time to generate two terms vectors. For prediction of the user's favorite programs, a simple vector space modification model is used as a feedback method as follows:

$$H^{'} = H_{new} + \gamma \cdot H_{old}, \tag{10}$$

where, $H^{'}$ is the extracted terms vector for prediction, H_{old} means the extracted terms vector from user watched programs before two weeks ago. H_{new} means the extracted terms vector from the latest user watched programs of two weeks. γ is coefficient. There are three ways to determine its value: by using results of existing research, or by determining them via experiments and/or by automatically learning them within the system.

The second step is to create some tasks for each category according to user watched programs, elements of tasks are start time (including day of week), channel, title, Japanese category, English category and the frequency of watched programs of certain category as shown in Figure 4.

These tasks are extracted from the watched programs, the frequency was acquired from user watched programs. Using these frequencies, to re-order the acquired categories for improving prediction precision.

Example of Japanese Feedback Tasks

```
<programme start="火曜日 19:00" channel="HTB">
    <title lang="ja_JP">プロ野球</title>
    <category lang="ja_JP">スポーツ</category>
    <category lang="en">Sports</category>
    <frequency> 5 </frequency>
</programme>
```

English Explanation

火曜日: Tuesday, プロ野球: Professional Baseball, スポーツ: Sports.

Fig. 4. Example of Category Feedback Tasks

3 Experiment

As mentioned above, the system based on our proposed approach was developed for experimentation to investigate its validity. In our experiments, we use open data to test the performance of our system. We adopted periodic training to our system. The training data is incremented on weekly basis.

Our experiment datasets were collected based on daily life of five graduate students of engineering over a period of about three months and total number of data was 2974. The data size of every week is shown in Table 1. The values of γ was given 0.5, the optimize value γ will be investigated in future experiments.

In order to keep the starting state constant for each user, the watched programs and the term features always started from an empty initial state.

Table 1. The Data Size of Each Week

Weeks	1	2	3	4	5	6	7	8	9	10	11	12
Data Num.(A)	48	47	46	48	52	52	51	51	50	45	48	48
Data Num.(B)	52	50	48	52	50	51	48	52	50	48	49	48
Data Num.(C)	48	52	50	50	48	52	50	52	52	50	48	48
Data Num.(D)	48	54	48	56	46	48	48	52	50	48	46	50
Data Num.(E)	50	52	48	52	48	48	50	50	50	48	48	52

Table 2. Comparison of Average Precisons

User	A	B	C	D	E
bigram(%)	65.0	64.2	58.3	64.5	62.7
bigram+tfidf(%)	67.1	65.2	63.1	64.9	64.8

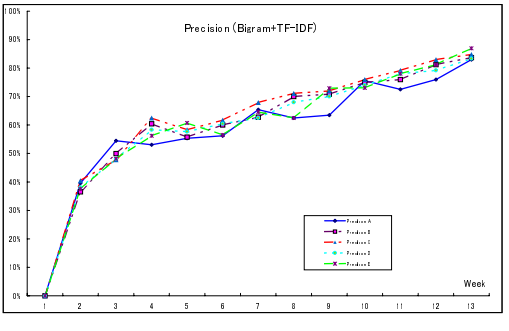

Fig. 5. Precision of bigram+$tf \cdot idf$

We use open data to evaluate prediction porfermance,the correct recommendations are judged from rank 60, and we use the next week training data as the correct results. Precision is difined as:

$$Precision = \frac{number\ of\ correct\ recommendations}{number\ of\ given\ correct\ results} * 100(\%) \qquad (11)$$

Figure 5 shows weekly precision of our proposed method.

As a comparison, we performed other experiments, in case of just using the score of extracted features based on the algorithm of Nakagawa, as the baseline of our experiments. Table 2 shows the average precison results.

4 Results and Discussion

In Figure 5 we plot the performance of our proposed method. It is indicated that the average precisions are about 67.1% (User A), 65.2% (User B), 63.1% (User C),64.9% (User D) and 62.9% (User E). Figure 5 also indicates that the highest precisions are about 83.0% (User A), 83.7% (User B), 84.8% (User C) and 83.3% (User D) and 86.9% (User E), all porfermance were improved than only using the score of extracted features based on the algorithm of Nakagawa. It is thought that the precision can be more improved, in case of enough learning datasets.

Moreover, the precision in the categories of "Movies" and "Sports" was found higher than the category of "News". It is thought that the description texts of category "News" are less descriptive than those of the categories of "Movies" and "Sports" in the IEPG data.

5 Conclusions and Future Work

In this paper, an idea for predicting users' favorite TV programs is described. The system has proven to have better performance according to our experiment results. We think that our system can provide simpler user interface with enhanced functions to TV watchers because they just need to select the recommendation programs rather than to consider any keywords for selection.

Our goal is to develop a personalized system of TV program recommendations with adaptive capability. We will try to test the performance and put our system to practical use for a computer or a digital television.

Experiments on a large scale including languages other than Japanese will be done in the near future to further verify the accuracy of the present system based on term extraction method. Other algorithms, such as reinforcement learning, SVM and neural networks will also be used for comparison.

References

1. Anna,B., John,Z., Kaushal, K.: Improving Ease-of-Use, Trust and Accuracy of a TV show Recommender. In Proc. of 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems: Workshop on Personalization in Future TV (AH), Universidad de Malaga, Spain (2002) 1-10.

2. Ardissono, L., Gena, C., Torasso, P., et al.: Personalized Recommendation of TV Programs. Lecture Notes in Artificial Intelligence n. 2829. AI*IA 2003: Advances in Artificial Intelligence, Pisa, Italy (2003) 474-486.
3. Johansson, P.: Natural language interaction in personalized EPGs. In Proc. of Workshop notes from the 3rd International Workshop on Personalization of Future TV, Johnstown, Pennsylvania, USA (2003) 27-31.
4. Setten, M. van, Veenstra, M., Nijholt,A.: Prediction strategies: Combining prediction techniques to optimize personalization. In Proc. of TV-02: 2nd Workshop on Personalisation in Future TV, Location Malaga, Spain (2002) 29-37.
5. Xu, J. A., Itoh, T., Araki K.: Action Prediction Method Using Recursive Different and Common Parts Extraction Method with N-gram. Journal of Human Interface Society, Vol.7,No.1,Japan (2005) 55-68.
6. Nakagawa, H., Mori, T.: A Simple but Powerful Automatic Term Extraction Method. In Proc.of 2nd International Workshop on Computational Terminology, COLING-2002 WORKSHOP, Taipei (2002) 29-35.
7. Joachims, T.: A Probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical Report,CMU-CS-96-118,Carnegie Mellon University, Pittsburgh, PA (1996).
8. http://membled.com/work/apps/xmltv/.