A smart TV system with body-gesture control, tag-based rating and context-aware recommendation

Wei-Po Lee *, Che Kaoli, Jhih-Yuan Huang

Department of Information Management, National Sun Yat-sen University, Kaohsiung, Taiwan

ABSTRACT

Smart TV, which enables viewers to conveniently access different multimedia contents and interactive services in a single platform, is currently becoming popular. The work presented here develops three new ways to enhance the performance of a smart TV. The first is to design a body control system that recognizes and interprets human gestures as machine commands to control the TV. The second is to create a new social tag-based method to recommend the most suitable multimedia contents for users. Finally, a context-aware platform is implemented to consider different environmental situations to make the best recommendations.

1. Introduction

Smart TV has quickly become popular in the market in recent years. In contrast to traditional TVs that only focus on media broadcasting, smart TV systems can deliver diverse multimedia contents from networked devices directly to the end-users, allowing them to access the contents through a user-friendly interface. Smart TVs also provide interactive Internet-based services, including media-on-demand, social networking, and online gaming. Smart TV continues to develop rapidly to offer increasingly more functions and services. In this work, we address three important issues closely related to smart TV, including the human–machine interface, content recommendations, and context awareness, and we develop the corresponding mechanisms to further enhance the performance of a smart TV system.

Traditionally, people have used device-based control to operate different consumer electronics at home. Researchers then began to implement systems that utilize personal hand-held devices to work as controllers [1,2]. Recently, different methods of human–machine interactions (such as voice-based and gesture-based control) have been proposed to provide natural control on the equipment without any remote control devices [3,4]. Gesture control has many advantages over traditional control methods, such as its high accessibility (with no need of physical contact), high flexibility (adaptable to various applications), and low cost (fixed sensor hardware cost, in contrast to touchscreen cost which increases exponentially while the screen size increases). To move towards an even more natural way for users to interact with machines, in this work we design a body control mechanism through which the smart TV can respond to the human users' actions.

On a smart TV system which can easily present different types of multimedia content to end-users, a large amount of contents can lead to information overload. Therefore, it is necessary to develop personalization techniques to recommend the most suitable content to users [5,6]. Many recommendation methods have been proposed, ranging from content-based user modeling to group-based collaboration. Generally speaking, the collaboration-based approach is considered more efficient and effective than the content-based user modeling approach [7]. The current approach toward organizing and sharing digital content through user-created metadata (i.e., social tags) suggests a potential mean to further improve collaborative recommendation by using metadata to interpret the users' preferences for any specific items. Additionally, we will employ such metadata for multimedia annotation. Also we will use the tag information to analyze the users' preferences and to exploit such information to make collaborative recommendations.

In addition to recommendation techniques that focus on multimedia items, context is an important issue to be considered in personalized recommendations, especially when mobile smart TVs become popular. These handheld devices can collect and analyze context information about their user. In general, context awareness means the ability of the computing systems to acquire and interpret the context information to adapt to the corresponding applications. In other words, context awareness enables a
system to capture a broad range of contextual attributes (such as the user’s location, activities, and surrounding environment) to better understand what the user is trying to accomplish and what content suits the user the most within that context [8–10]. By integrating context information into the application service, a recommender system can fulfill the user’s needs more efficiently and practically.

Taking the above issues into account, we develop a smart TV system with several unique features. First, we design a Kinect-based system to recognize human body gestures for TV control. Second, we compare different computational methods for making personalized recommendations on multimedia items. Following the current trend of community-based information sharing [11–15], we also integrate social tags to annotate multimedia items to improve recommendation performance. The experimental results show that the tag-based method outperforms other conventional methods. Finally, we implement a context-aware platform to present how we integrate various environmental situations into the proposed method to perform item recommendation accordingly.

2. Background

As mentioned above, due to the tremendous amount of digital multimedia items a smart TV can broadcast, a recommendation mechanism is needed to offer better services. Recommender systems have been advocated in different service domains for many years [16,17]. In general, the recommendation techniques can be categorized into three types: content-based, collaborative filtering, and hybrid methods [16]. The content-based approach predicts the user’s preferences for new items based on historical records. Therefore, the most important issue in this approach is constructing a computational model to perform the prediction. Many machine-learning approaches have been applied to construct user models. Nevertheless, the content-based approach largely relies on sufficient samples to construct the model. This approach often recommends items within a specific scope and thus loses item diversity (i.e., ignoring items of unfamiliar classes).

In contrast to the content-based method, the collaborative filtering (CF) method does not build a personal model for prediction. There are two major techniques to perform CF methods: the neighborhood methods and latent factor models [7]. The neighborhood methods recommend items to the user according to the evaluations (opinions) of other users with similar tastes (or recommends items similar to the ones with high user ratings in a similar way). In such an approach, therefore, the measurement of similarity between users is most important so that the system can employ a k-nearest neighbor method to find the most similar users to perform the recommendation. The system’s prediction of a new item for a user is thus based on a combination of the ratings of the user’s nearest neighbors. This approach has been widely used in different applications, for example [2,18].

The above similarity-based methods are very popular because they are intuitive and relatively simple to implement. They also offer useful and important properties: explicit explanation of the recommendations and easy inclusion of new ratings [7,16]. However, standard neighborhood-based methods raise some concerns: for example, the similarity function is often arbitrary; the interactions among neighbors are not taken into account; the weights for each neighbor need to be determined properly; and some users have rated only a very small number of items (i.e., the cold-start problem).

The second type of CF methods, latent factor models, provides an alternative approach by transforming both users and items to the same latent factor space. This space intends to explain the ratings on some implicit factors obtained automatically. Different algorithms have been proposed to derive these factors by minimizing the difference between the predicted ratings and the observed ones (e.g., [19,20]). Matrix factorization (i.e., singular value decomposition or SVD) is one of the most popular latent factor models, in which each user and item is assumed to be represented by a small number of unobserved features. Though latent factor models can provide somewhat more accurate results than the neighborhood methods, still they have some disadvantages. For example, optimizing the often used object function (the sum-of-square of factorization error) in the traditional matrix factorization methods cannot guarantee the consistency of the predicted ratings and the user preferences. Also, the latent factor models often employ a stochastic algorithm (such as gradient decent) to solve the optimization task. As can be observed, when the numbers of users and items increase, the corresponding optimization task will become more difficult to solve.

Some researchers have pointed out that factorization methods can lead to relatively accurate results, while neighborhood methods can have some practical advantages [7]. Therefore, several integrated methods have been suggested to have advantages from both types of methods, for example, to derive a hybrid model that can add a local perspective delivered by the neighborhood method into a global factorization method [7]. Some other techniques are also proposed to extend the capabilities of recommender systems in general, including taking more user and item information into account, incorporating contextual information, and supporting multi-criteria rating [10,16,21]. Meanwhile, to overcome the cold-start problem, the notion of trust has been introduced to improve the recommendation accuracy [22]. Trust plays a major role in exchanging relationships among people; it is often used to decide with whom we share information and from whom we accept information. It is believed that by utilizing a trust network (a social network augmented with trust ratings) in collaborative filtering methods, better results can be obtained [22,23].

In addition to measuring item similarity or user similarity to predict user preferences, we can also analyze how the user prefers specific items from different feature dimensions (i.e., in more detail). Social tags are brief descriptions of items and can be used as features to capture the semantics of the target items, for example [11,24]. These tags are freely supplied by a community of Internet users to aid access to large collections of media [12,13]. Compared with knowledge representation schemes that involve domain experts, the tagging activity shifts the task of classifying domain items from knowledge engineers to Internet users; it is thus a cost-effective alternative to the popular and precise knowledge ontology in content annotation [14,24]. Additionally, tagging is neither exclusive nor hierarchical. In some circumstances, tagging has advantages over hierarchical taxonomies. Therefore, in this work we choose to use social tags to represent item features, and we propose a new tag-based collaborative filtering (tag-CF) method for multimedia recommendation. In contrast to other works that use tags to annotate target items (e.g., [15,25]), our work uses social tags to predict a user’s preference for recommendation.

Context plays an important role in determining the relevance of an application’s service (or function) to a user’s needs, and any small contextual changes may lead the user to select a different service. Dealing with the context issue involves defining contexts relevant to the application service and identifying the key contexts in which people often use the service. Regarding different mobile applications, context factors can be defined as any information used to characterize the user situation that can influence his decision in requesting a service. There are two types of context factors: personal and environmental [8,26]. Personal context is the personal state or condition of the user himself (such as his emotional...
3. The proposed personalized smart TV system

To provide smart TV services, we develop a system framework with a client-server architecture. The client is responsible for gathering and processing user-side information in real time, and the server mainly manages user profiles and performs computations for personalization. Fig. 1 illustrates the framework.

As with other smart TV systems, the proposed system offers several popular multimedia functions, such as digital content broadcasting, social networking, web browsing and searching, e-book reading and so on. Most importantly, the system has three unique features: recognizing and interpreting a user's body gestures for TV control, collecting social opinions to perform collaborative recommendations, and using environmental information to enable context-aware services. The three kernel modules, recognition, recommendation, and context, shown in Fig. 1 are developed for providing the above features, respectively. Though the system includes a module for body-gesture control, whether this function can be launched entirely depends on the type of the displaying facility. To deliver context-aware recommendation, the system first activates the recommendation module to determine a priority list of available multimedia items (based on a tag-based user similarity measurement). Then the system uses the context module to re-rank the priority list, according to the current context information (e.g., user location, device conditions) collected from the relevant devices (e.g., the user's mobile phone). With the recommendation list, the user can easily find a preferable and suitable item. The details are described in the subsections below.

3.1. Learning and recognizing body gestures

Gestures can be regarded as a type of body language and can be used to design the mechanism of communication between humans and machines. Through the process of appropriate interpretation, the user's gestures can be recognized automatically by the system. Our system includes such a mechanism for the user to interact with a TV and control it. Usually, the gestures can be static or dynamic. In the latter case, gestures are produced continuously and each is affected by the preceding gestures. To infer the aspects of gesture, the system has to sense the user's movements and collect his position data, which is often achieved using traditional sensing devices attached to the user (such as magnetic field trackers) or using cameras with computer vision techniques. The latter offers a more natural method of human–machine interaction. Microsoft's Kinect is a popular vision-based device and able to perform real-time pose recognition. In this work, therefore, we implement a Kinect-based system to recognize human gestures for TV control. Fig. 2 illustrates the flow.

To recognize the user's gestures, the system needs to temporally segment the gestures to extract a time-varying sequence of parameters that describe the relevant body parts. Recognition of the user's gestures from these parameters then becomes a pattern recognition task that involves transforming the input into the appropriate representation and classifying it from a database of pre-defined gestures. As mentioned previously, we exploit Kinect to achieve such mappings. Kinect segments a depth image into a dense probabilistic body part labeling; it localizes the spatial modes of each part distribution and thus generates confidence-weighted proposals for the 3D locations of each skeletal joint [29]. In this work, we utilize Kinect SDK to extract body-part data of the human demonstrator and interpret the data into different gestures. A set of body gestures has been pre-defined and encoded to appropriate control commands. Table 1 lists the major mappings. In the table, the arrows describe the motions of the user’s arms, and the circles indicate the user’s forward extension of the arms. The reference data points of the body parts are also shown. This table can be expanded by creating and including more human gestures in the same way. Also, the users are allowed to define their personal mappings between gestures and control commands.

As shown in Fig. 2, in the gesture recognition procedure, the support vector machine (SVM) method is employed to train a user model. SVM has been shown to work very well with high-dimension data in many classification applications. Therefore, we use this method for the gesture recognition (classification) task here. To prepare the training dataset, the system asks the user to demonstrate each of the pre-defined gestures (as shown in Table 1), and the body-part data are tracked and encoded. In this work, we record the 3D coordinates (relative to the hip center) of the left and right hands within a certain time interval (i.e., 1.5 s) and use these coordinates as attributes to form a data record. The trained model is used for later gesture recognition (i.e. to predict which of the pre-defined gestures the user is performing), and the recognized gesture is then mapped to the command to control TV. To avoid inconsistent commands due to multiple users competing with each other to control the TV, the system can only serve one person (the first human figure recognized) at the present stage.

3.2. Personalized recommendation

To perform personalized content recommendation, the first step is to create a personal profile as a common reference point. Therefore, it is crucial to collect profile data directly (through questioning the user) or indirectly (through observing the user’s behavior) and to keep updating the user’s profile based on any changes of needs or contexts. The next step is to use the obtained information to infer the user’s preferences. Here, we adopt the collaborative filtering method to achieve personalized recommendation. As described in Section 2, several useful models and techniques can be used to improve the accuracy of collaborative filtering methods. Nevertheless, the major focus of this section is on investigating how the social tags can help to develop a recommendation mechanism to be included in a smart TV system, rather than on comparing comprehensively the use of social tags with various recommendation algorithms which is beyond the scope of this study. Therefore, in this work we mainly concentrate on developing and evaluating two collaborative filtering methods, including a standard neighborhood collaborative filtering method and a tag-based collaborative filtering. They are described in the following subsections.

![Fig. 1. The proposed system framework for smart TV.](Image)
### 3.2.1. Collaborative filtering

Collaborative recommendation performs predictions for a specific user based on the evaluations (ratings) performed by other users with similar tastes. For a user $u_a$, users with the most similar preferences or interests are selected as a set of neighbors $\text{Neig}(u_a)$, and their collective opinions on a certain item $m_{\text{recom}}$ are used to predict whether $u_a$ will like this item. That is, the rating of the preference of a specific item $m_{\text{recom}}$ is defined as:

$$R_{\text{pre}}(u_a, m_{\text{recom}}) = R_{\text{pre}}(u_a) + \frac{z_1}{C_2} \sum_{u_n \in \text{Neig}(u_a)} \text{Sim}(u_a, u_n) \cdot (R_{\text{pre}}(u_n, m_{\text{recom}}) - \bar{R}_{\text{pre}}(u_n))$$  \hspace{1cm} (1)

In the above equation, $R_{\text{pre}}(u, m)$ represents the preference of user $u$ on item $m$; $\bar{R}_{\text{pre}}(u)$ is the average preference rating of user $u$ regarding all items he has rated; $\text{Sim}(u_a, u_n)$ is the similarity between two users $u_a$ and $u_n$; and $z_1$ is the normalized factor, which can be calculated as:

$$z_1 = \frac{1}{\sum_{u_n \in \text{Neig}(u_a)} |\text{Sim}(u_a, u_n)|}$$  \hspace{1cm} (2)

There are many methods to calculate the similarity mentioned above. One common method is the Pearson correlation coefficient. Here, we use this method to measure the similarity between two users $u_a$ and $u_n$ as:

$$\text{Sim}_{\text{CF}} = \frac{\sum_{m_i \in \text{Com}(u_a, u_n)} (R_{\text{pre}}(u_a, m_i) - \bar{R}_{\text{pre}}(u_a)) \cdot (R_{\text{pre}}(u_n, m_i) - \bar{R}_{\text{pre}}(u_n))}{\sqrt{\sum_{m_i \in \text{Com}(u_a, u_n)} (R_{\text{pre}}(u_a, m_i) - \bar{R}_{\text{pre}}(u_a))^2} \cdot \sqrt{\sum_{m_i \in \text{Com}(u_a, u_n)} (R_{\text{pre}}(u_n, m_i) - \bar{R}_{\text{pre}}(u_n))^2}}$$  \hspace{1cm} (3)

#### Table 1

<table>
<thead>
<tr>
<th>Command/data points</th>
<th>Gesture</th>
<th>Command/data points</th>
<th>Gesture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start program</td>
<td>Volume control: (decrease)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right/left hand</td>
<td>Right/left hand</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right/left elbow</td>
<td>Right/left elbow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Head</td>
<td>Hip-center</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change channel: (previous)</th>
<th>Volume control: (increase)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right/left hand</td>
<td>Right/left hand</td>
</tr>
<tr>
<td>Right/left elbow</td>
<td>Right/left elbow</td>
</tr>
<tr>
<td>Right/left shoulder</td>
<td>Hip-center</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change channel: (next)</th>
<th>Volume control: (increase)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right/left hand</td>
<td>Right/left hand</td>
</tr>
<tr>
<td>Right/left shoulder</td>
<td>Right/left shoulder</td>
</tr>
<tr>
<td>Hip-center</td>
<td>Hip-center</td>
</tr>
</tbody>
</table>

**Fig. 2.** The flow of how the user’s gestures are interpreted.

![Mapping Table](image)

Table 1

<table>
<thead>
<tr>
<th>Gestures and the corresponding TV functions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command/data points</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Start program</td>
</tr>
<tr>
<td>Right/left hand</td>
</tr>
<tr>
<td>Right/left elbow</td>
</tr>
<tr>
<td>Head</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change channel: (previous)</th>
<th>Volume control: (increase)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right/left hand</td>
<td>Right/left hand</td>
</tr>
<tr>
<td>Right/left elbow</td>
<td>Right/left elbow</td>
</tr>
<tr>
<td>Right/left shoulder</td>
<td>Hip-center</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change channel: (next)</th>
<th>Volume control: (increase)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right/left hand</td>
<td>Right/left hand</td>
</tr>
<tr>
<td>Right/left shoulder</td>
<td>Right/left shoulder</td>
</tr>
<tr>
<td>Hip-center</td>
<td>Hip-center</td>
</tr>
</tbody>
</table>

![Mapping Table](image)
In Eq. (3), \( \text{Com}(u_a, u_b) \) is the set of items that both users \( u_a \) and \( u_b \) have already rated. This coefficient is between 1 (the preferences of both users are exactly the same) and -1 (their preferences are opposite each other); a value of zero means their preferences are not correlated.

### 3.2.2. Tag-based collaborative filtering

Though the above CF method is efficient for content recommendation, it does not consider the reasons behind the user’s comments in detail. Other methods that can provide such detailed analysis are thus needed. As described previously, folksonomy (i.e., the community-based method) is a very popular technique for annotating items (or content). To exploit the current trend in adding metadata to the shared contents, we develop a new approach that incorporates user-specific tags with CF to conduct item recommendation.

Fig. 3 shows the interface we develop to collect the user’s opinions. To enable users to share tags, we provide a set of popular tags (the “Default Tags” listed in the interface) for multimedia annotation. These tags are created manually with reference to online comments for movies. The user can give a value from 1 (lowest) to 5 to each tag to explicitly express his evaluation. In addition to the suggested tags, the user is allowed to define new tags with rating values through the interface (the “Other Tags”). The above tagging method can be further expanded with machine learning techniques to dynamically derive the tags from users with certain community-based strategies [30].

Similar to the collaborative filtering method described above, our Tag-CF method also measures the similarity between users and finds a neighbor set \( \text{Neig}(u_a) \) for a specific user \( u_a \) to predict his preference on a certain item \( m_{\text{recom}} \). The same equation for measuring rating preference in CF is used here to calculate \( R_{p\text{ef}}(u, m) \). However, the Tag-CF method uses tags to look for similar users rather than item preferences, as in the traditional CF method.

An example is presented here to show the advantage of using tags and how the tag-based CF differs from CF. Suppose there are three users \( (u_1, u_2, u_3) \), three tags \( (t_1, t_2, t_3) \), and five items \( (m_1, m_2, m_3, m_4, m_5) \) in the recommendation system. Figs. 4 and 5 describe the preference ratings and the corresponding tag evaluations, respectively, of the three users on the five items. In both tables, the values in the first column (i.e., the column “Pre”) for a specific item are the overall ratings of a user on this item, and the values in a tag-column mean how the user regards this item on the dimension a tag specifies. The goal of this example is to predict the rating of user \( u_m \) on item \( m_3 \), based on the information recorded in the two figures. As shown in Fig. 4, if the system uses the CF method for prediction, it will find \( u_2 \) as the most similar user and adopt his opinion. On the contrary, if tags are taken into consideration, \( u_3 \) is recognized as the most similar user instead because from Fig. 5 we can observe that \( u_3 \) takes the same perspectives as that of \( u_1 \) (i.e., they use the same tags) in rating items and giving similar evaluation results. Therefore, the system will use \( u_3 \)'s opinion to predict whether \( u_m \) likes \( m_3 \) or not. In this example, using tags for prediction is in fact more precise than using preference ratings.

Taking into account the factors of user preference and tag evaluation together, we develop a new method to calculate the similarity between two users and to use the results to work with the collaborative filtering method. The user similarity is now described as below:

\[
\text{Sim}_{\text{TagCF}} = \left[ \left( \frac{z_2 \times \sum_{m_i \in \text{ComLL}(u_a, u_b)} \text{TC}(u_a, u_b, m_i)}{2} \right)^{-1} \right] - z_4 \\
\times \left( \sum_{m_i \in \text{ComDD}(u_a, u_b)} \text{TC}(u_a, u_b, m_i) \right)
\]

if \( \text{Com}_{\text{LL}}(u_a, u_b) \neq \phi \) and \( \text{Com}_{\text{DD}}(u_a, u_b) \neq \phi \);  

(4)

![Fig. 3. Interface for collecting user’s ratings on different tags.](image)

![Fig. 4. The overall preference ratings of three users on five items.](image)

![Fig. 5. The tag evaluations of three users on five items.](image)
SimTAGCF = \( z_2 \times \sum_{m_1 \in \text{Com}_D(u_1, u_2)} TC(u_1, u_2, m_1) - z_4 \times \sum_{m_1 \in \text{Com}_D(u_1, u_2)} TC(u_1, u_2, m_1), \) if \( \text{Com}_D(u_1, u_2) \neq \phi \) \(\) (5)\\
SimTAGCF = 0, if \( \text{Com}_D(u_1, u_2) = \phi \) \(\) (6)

In the above equations, \( \text{Com}_D(u_1, u_2) \) represents the set of items that \( u_1 \) and \( u_2 \) have evaluated, and the two users both like these items; \( \text{Com}_D(u_1, u_2) \) includes the items that both users do not like; and \( \text{Com}_D(u_1, u_2) \) includes the items that one of the two users like. In addition, the normalized factors \( z_2, z_3, \) and \( z_4 \) are defined, respectively, as the following:

\[ z_2 = 1 / \sum_{m_1 \in \text{Com}_D(u_1, u_2)} |TC(u_1, u_2, m_1)| \] \(\) (7)\\

\[ z_3 = 1 / \sum_{m_1 \in \text{Com}_D(u_1, u_2)} |TC(u_1, u_2, m_1)| \] \(\) (8)\\

\[ z_4 = 1 / \sum_{m_1 \in \text{Com}_D(u_1, u_2)} |TC(u_1, u_2, m_1)| \] \(\) (9)

Here, the equations accumulate tag evaluations of the two users’ preferences. If both users have the same preference ratings, the similarity will increase; if they have different preferences, the similarity will decrease. In these equations, \( TC(u_1, u_2, m_1) \) measures the similarity between \( u_1 \)'s and \( u_2 \)'s tag evaluations about \( m_1 \). The calculation is based on the Tanimoto coefficient, which is an extended Jaccard coefficient that is often used to calculate the similarity of vectors with asymmetric binary attributes. In our work, if the tags are only used (evaluated) by one user, a default value of 3 will be automatically assigned to another user in comparison. However, if the tags are not used by any of the users, they will simply be ignored. In this way, the \( TC(u_1, u_2, m_1) \) is defined as:

\[ TC(u_1, u_2, m_1) = \frac{\sum_{t_g} R_{\text{tag}}(u_1, m_1, t_g) \cdot R_{\text{tag}}(u_2, m_1, t_g) - \sum_{t_g} R_{\text{tag}}(u_1, m_1, t_g)^2 - \sum_{t_g} R_{\text{tag}}(u_2, m_1, t_g)^2 + \sum_{t_g} R_{\text{tag}}(u_1, m_1, t_g) \cdot R_{\text{tag}}(u_2, m_1, t_g) \cdot R_{\text{tag}}(u_1, m_1, t_g) \cdot R_{\text{tag}}(u_2, m_1, t_g)}{\sum_{t_g} R_{\text{tag}}(u_1, m_1, t_g)^2 + \sum_{t_g} R_{\text{tag}}(u_2, m_1, t_g)^2 - \sum_{t_g} R_{\text{tag}}(u_1, m_1, t_g) \cdot R_{\text{tag}}(u_2, m_1, t_g) \cdot R_{\text{tag}}(u_1, m_1, t_g) \cdot R_{\text{tag}}(u_2, m_1, t_g)} \] \(\) (10)

In the above equation, \( R_{\text{tag}}(u, m, t_g) \) means the evaluation result of user \( u \) on tag \( t_g \) that is used to annotate item \( m_1 \).

The computational complexity of the tag-CF method presented above is similar to that of the standard neighborhood-based collaborative filtering method. In the traditional method, there is no need for model training, and the computational complexity for recommending an item (in the test phase) to a user is about \( O(mn) \), where \( m \) is the number of users and \( n \) is the number of items. Because the tag-CF method expands the overall rating for a specific item to multiple feature dimensions defined by the tags, its computational complexity thus becomes \( O(mnk) \) approximately. Here, \( k \) is the average number of tags used to annotate an item.

### 3.3. Context awareness

Because mobile Internet use has become increasingly popular, a variety of environmental contexts that do not occur in stationary Internet use should be taken into account in the deployment of context-aware recommendation. This section describes the contexts considered in our system, including user location, audience, mobile device, and network condition, as discussed below. These contexts were used to develop a set of rules to re-rank the recommendation list derived from the user preference.

The first context is location information. A location-aware system operates according to the particular geographical location it is situated in. One way to achieve location awareness is to use the location of a mobile user as a parameter for service provision, for example, using a handset-based positioning solution (such as a global positioning system, GPS) or a network-based positioning solution (such as the GSM cellular system). Here, we do not use the detailed location parameters; rather, we divide the location context into public and private types. We then use this information to restrict the playing of multimedia content. In accordance with the MPAA (Motion Picture Association of America) rating systems we categorize multimedia products into five levels: G (general audiences), PG (parental guidance suggested), PG-13 (parents strongly cautioned), R (restricted), and NC-17 (only people 17 and older admitted), and we map them with appropriate location types. This approach is taken to consider the impressions of the people around the user accessing the multimedia with a mobile device.

The second type of context, social context, considers the crowd (or audience) around the user. Sometimes a user may invite other people to access multimedia content (e.g., to watch a movie) together, and the system needs to consider whether the content is suitable for all of them. Similar to the situation in the above location context, the recommendation list needs to be adjusted and some contents filtered out. In this work, we use the above MPAA rating criterion to remove the content not suitable for every viewer, and we re-rank the remaining items in the list according to their popularity.

The third type of context is related to the infrastructure delivering the service. Today, it is very popular to use powerful mobile devices, such as mobile phones, slim notebooks, or portable TVs, to access multimedia content in the mobile environment. Devices used by mobile users are diverse and heterogeneous. These devices have different screen sizes, memory units, media supports, connection speeds, and, perhaps the

most important factor, computation capabilities with which to address the multimedia content. For high-resolution media content, a more powerful computational mechanism is needed to prevent the delay of video and audio in media playback. Battery power is another important issue to note because accessing multimedia through mobile devices is a very power-consuming application. Under such circumstances, some content cannot be supported by mobile devices with more hardware limits. Therefore, a context-aware system must take the device context into consideration when making recommendations.

In addition to hardware devices, communication is a critical factor that decides the quality of service directly. High-resolution online content is generally preferred, but it can require the support of high network bandwidth. Multimedia products can therefore be produced in different modes to fit in the available network bandwidth. For example, YouTube started to support high-resolution videos in 2008 when the network condition was largely improved. The website now allows the user to choose the appropriate resolution mode for the multimedia he is playing, depending on the network and device conditions. Similarly, our system considers the communication condition to recommend the most suitable content to users.
4. Implementation and experiments

Having presented our smart TV system with the corresponding strategies in human–machine interaction, social community-based recommendation, and context information processing, in this section we describe the experiments conducted. In the first phase, we presented how the user data were used to train a model by the SVM learning method and corresponding results. In the second phase, we evaluated and compared the performance of the different recommendation methods. Finally, in the third phase, we embedded the most efficient recommendation method into the proposed framework. We then illustrate the implementation details of our system.

4.1. Learning to recognize user gestures

As described in Section 3.1, we use Kinect to track the user’s trajectories and extract the body-part data to constitute a dataset to build a recognition model (classifier). Here, a trajectory means a series of image frames captured by Kinect with a frame update rate 30 fps (frame per second), and we define a trajectory to include 45 frames (i.e., tracked within 1.5 s). That is, each data record has $45 \times 6$ (i.e., the 3D coordinates of the left and right hands in the 45 consecutive frames) attributes in total. To derive a robust model with noisy tolerance ability, in the experiment we asked the user to demonstrate each specific gesture of control command ten times, and used the trajectories belonging to the same gesture to build a normal distribution to generate more (i.e., 200) data records. The eight types of gesture data were then assembled to be the dataset for model training and testing.

The online software LibSVM was adopted to learn the model for gesture recognition, and a 10-fold cross validation strategy was used for performance evaluation. The accuracy of the model reached 92.49% (value was averaged over six different trails/users). These results show that the SVM models can be successfully trained to perform gesture recognition.

4.2. Performance evaluation of recommendation

The performance evaluation of personalized recommendation can be conducted in two ways: a quantitative comparison of user preference prediction or a qualitative investigation of user satisfaction. The former focuses on the computational methods, while the latter focuses on the users’ perspectives [31]. Because our current goal is to include a more precise recommendation mechanism in the proposed smart TV system we used the first method (i.e., preference prediction) for the experiments.

A dataset was collected from 82 individual users. Each participant was asked to provide at least 40 movie items (chosen from a default list or specified by the user manually) he had viewed and evaluated previously. To avoid the data imbalance problem and to produce more objective evaluations for different methods, we asked the user to give roughly the same number of example items for each class (like or dislike). For each item, the user was required to further express his degree of preference from the common five-scale measurement (with the degree of preference decreasing from 5 to 1). These values were used to predict the user preference in the collaborative filtering method, as indicated.

In addition to their overall preferences, during the data collection process, the users were asked to arbitrarily select tags from the default list or add new tags and then give their ratings on these tags (see Fig. 3). The rating for each tag was a value from 1 (lowest) to 5 representing explicitly the user’s evaluation. This step was performed to investigate how the user commented on a specific item from different feature dimensions. To enable users to share tags and keep the annotation consistent, in the experiment reported here, we adopted the method of suggestive-tagging: only the ratings on the defaults tags (as shown in Table 2) were used. In the collected dataset, the user gave 12 tag evaluations for each movie item on average.

With the above dataset, two types of recommendation methods were used, content-based and collaborative filtering. The first series of experiments evaluated the performance of the proposed tag-based method for the situation in which a content-based strategy is adopted for recommendation. In the experiments, the traditional keyword-based and tag-based methods were used to represent the multimedia content. For the keyword-based method, the publicly available Internet Movie Database (IMDb) was used, and the keywords for each movie were extracted from the database to represent the movie accordingly. For the tag-based method, only the ratings on the defaults tags listed in Table 2 were used to build the user model for preference prediction.

To determine which of the above representations (i.e., keywords and tags) can deliver the best performance, we subjected them to three popular computationally efficient content-based methods: decision tree, support vector machine (SVM), and Naïve Bayes (NB) classifiers. Additionally, the 10-fold cross validation evaluation method was employed to obtain a more objective assessment. Fig. 6 presents the results of three popular performance measurements in classification: accuracy, precision and recall. As can be observed, in all measurements, the tag-based method outperformed the keyword-based method when the above three machine-learning techniques were used for user modeling. The result shows that the proposed representation can more pertinently capture user characteristics in movie recommendations.

In addition to the content-based strategy, the second series of experiments was based on user collaboration. This series was conducted to examine the efficiency of the tag-based method when used with the collaborative filtering strategy. In this series of experiments, we first employed the methods described in Section 3.2 to measure the similarity between users. If this measuring result exceeded a certain threshold, their preferences were considered similar. This way, the nearest neighbors of a certain user could be determined. To examine the effect of information sharing by the traditional collaborative filtering method and the proposed tag-based collaborative filtering method, we arranged three sets of experiments with different user-similarity thresholds (0.7, 0.6 and 0.5).

Fig. 7 presents the test results in which the three criteria were used for performance measurement as in the experiments using the content-based methods. As shown, the collaborative filtering methods performed better than the content-based methods. We can also observe from this figure that the tag-based method offered

| Table 2 |
The suggested tags for annotation.|

<table>
<thead>
<tr>
<th>Story</th>
<th>Climax</th>
<th>Originality</th>
<th>Profundy</th>
<th>Dialogue</th>
<th>Pace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thinking</td>
<td>Portraying</td>
<td>Role</td>
<td>Popularization</td>
<td>Horror</td>
<td>Music</td>
</tr>
<tr>
<td>Touchingness</td>
<td>Satire</td>
<td>Humor</td>
<td>Entertainment</td>
<td>Cast</td>
<td>Acting</td>
</tr>
<tr>
<td>Visual</td>
<td>Action</td>
<td>Stunt</td>
<td>Characteristic</td>
<td>Atmosphere</td>
<td>Director</td>
</tr>
</tbody>
</table>
Fig. 6. Comparison of different content-based methods.

Fig. 7. Comparison of the tag-based and traditional CF methods.
better recommendations than the traditional CF method in terms of accuracy, precision and recall for the threshold of 0.7. The results of the two methods show significant differences (t-test, $\alpha < 0.05$).

The figure shows that the tag-based method had a better performance when a similarity threshold of 0.6 was used (therefore, more user opinions were taken into consideration), and the results...
obtained from the two methods were significantly different. However, when lower thresholds (0.5 or less, meaning lower similarity) were used, the two methods showed no significant differences after statistical examination ($t$-test, $\alpha < 0.05$). These results indicate that social tags are better media for capturing user characteristics and that they can be used to precisely measure user similarity. Therefore, the method based on tag measurement can deliver better performance in preference prediction.

4.3. System implementation

The proposed smart TV system has been implemented and verified. In addition to the multimedia content broadcasting system, we integrated popular interactive software into our system, including the Google search engine, YouTube, Facebook, and a PDF reader. At the first stage of verification, we used the Kinect SDK to construct a mechanism on a PC (with a CPU of Intel Pentium Dual-Core 2.0 GHz processor and 2 GB RAM) for TV control. As described in Section 3.1, this mechanism can recognize and interpret the user’s body gestures and turn them into control commands. After the software corresponding to the command is driven, the user can operate it through his body gestures. Fig. 8 presents several examples in which the user requested a keyboard on the screen and typed a keyword to perform a Google search (Fig. 8(a)), chose a TV program from the recommendation list (Fig. 8(b)), changed the TV channel to the next (Fig. 8(c)), and lowered the volume of the media channel (Fig. 8(d)).

In addition to the above implementation, we then expanded our system to perform the personalized content recommendation in a...
context-aware environment. Fig. 9 illustrates the enhanced client-server architecture, in which the tag-based collaborative filtering method was used to develop the recommendation module, as it was shown to deliver the best performance in the above section. Since the gesture control module relies on the user’s individual environment, we now focus mainly on the context-aware recommendation. The work flow of context-aware recommendation is that the server side of the system uses the recommendation module to produce a candidate list from the currently available multimedia items. Then the system exploits the context module to re-rank the candidate items and produce a new priority list, as shown in Fig. 9. The re-ranking is achieved by taking context information (including network condition, device type, and user location in this work) collected on the client side, together with the pre-programmed rules to determine the priority. This recommendation list is send to the user for his reference regarding multimedia selection.

To identify the networking condition, the system on the server sends certain packets to the client and waits for its responses. The system can then estimate the bandwidth available for the user. Meanwhile, the system retrieves information recorded in the HTTP header received from the client to classify the operating system embedded in the most popular client devices (currently including Windows NT, Mac OS, Linux and Solaris for non-mobile devices and Windows CE, iPhone Mac OS, Palm OS, Pocket PC, EPOC, and Linux for mobile devices) and to recognize the type of the user device accordingly. Based on the above network condition and user device information, the proposed system is able to suggest the most suitable resolution to the user.

As mentioned in Section 3.3, the GPS or GSM cellular system can be used (with an electronic map) to provide detailed location context. However, because handheld devices with embedded positioning systems are not widely used in daily life (they are mostly used for tracking purposes), for practical reasons we did not integrate the positioning-system-based location information into our system in our current implementation. Instead, we asked the user to provide his environmental context to the system by making selections from pre-defined choices on the interface (as shown below).

Fig. 10 shows two screenshots of the system presented to the user. As presented, the upper-left icons provide pre-defined options of social and location contexts. The social context here tries to capture the social condition around the user. The user can tell the system whether he is alone (single user) or with other people (multiple users) through these options. Similarly, the user can tell the system whether he is in a public or private domain. The recommended movies are listed below the icons according to the contexts the user specifies. In Fig. 10, the interface windows show the recommendation results for two different situations: in the first the user chooses both “multiple users” and “public place”; in the second, the user chooses “single user” and “private place”. The frame for showing the movie is allocated in the middle area of the interface window. The information of the user device, currently including the type of operating system and the network bandwidth, is detected and provided on the right-hand side, and the resolution for viewing the movie is suggested accordingly. Fig. 11 shows the recommendation examples for considering device and network conditions. In addition, the tags used to annotate the movie are provided below the movie frame. The user can provide his feedback through the evaluation of these tags.

5. Conclusions

In this work, we developed a smart TV system with three important features. The first is a body gesture-based control system implemented to enrich the ways of operating smart TVs. This control mechanism recognizes human body gestures and turns them into machine commands. We also advocated the need for developing a personalized recommendation service to help end-users access the most suitable digital content. Considering the current trend of organizing and sharing digital content through user-created metadata, we proposed a new recommendation method that exploits social tags to annotate multimedia items. To verify the proposed method, experiments were conducted to prove that it surpasses other methods. Because the social tags can measure user preferences from different semantic dimensions, our method has a better recommendation function. In addition to the user’s preferences, we also consider different environmental conditions, including user location, audience, device, and network connection, and we extend our work to become a context-aware system.

The work presented here shows prospects for further research. We are now developing an adaptive strategy to maintain the default (suggested) tag list periodically, by extracting useful tags from the ones freely created by users and establishing the meaningful relations among these tags automatically. Meanwhile, investigations have been conducted to find new ways to construct a social trust network with varying weights. We hope to incorporate the users’ trust and the tag-based rating presented in this work to further improve the recommendation performance. We are also trying to integrate more efficient CF techniques and models to the proposed tag- CF method to extend its abilities in making recommendation. Moreover, we plan to build an enhanced recognition system for multiple users and to determine the priorities for these users in TV control.

References