2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing

Machine Learning Approach to Task Ranking

Michael Yoseph Ricky[1],
Spits Warnars Harco Leslie Hendric[2], Widodo Budiharto[3], Bahtiar Saleh Abbas[4]
Bina Nusantara University[1, 2, 3, 4]
mricky@binus.edu[1], shendric@binus.edu[2], wbudiharto@binus.edu[3],
bahtiars@binus.edu[4]

Abstract – *There are variety of methods and algorithms that can be used to overcome the ranking problem. Task ranking is one of the problems that can be solved by using a machine learning algorithm ranking problem. This work focuses on finding the right approach and corresponding algorithms in the process of ranking to be able to help people in determining which jobs have a higher priority than others. Our approach is to compare several algorithms performed in the process of ranking that are Bipartite Ranking, k-partite Ranking, and Ranking by pairwise comparison. We're used questionnaires and deployment of prototype of Intelligent Personal Assistant Agent to apply the appropriate algorithm in intelligence agent in arranging task priority in daily activity that must be done by the users. After training dataset and evaluate the validation dataset using NDCG, it is found that the collaborative ranking used have a more accurate value / lower variance test evaluation because it uses a large dataset and smaller training dataset. We found that labeling for more than 2 values it is not recommended to use a bipartite ranking if there are many repetitive data, both k-partite ranking and rank by pairwise comparison are able to be used for multi-dimensional data labeling.*

Keywords: *Task Ranking, Bipartite Ranking, k-partite Ranking, Ranking by pairwise comparison, Intelligence Agent*

I. Introduction

Technology is created by human and will be used to help human activities where the technology which is derived from the human mind is created to assist humans in performing their activities more easily. It is without doubt that technology can help human in performing their daily work or even performing as an assistant who can help in reminding the plan of activities that should be done. One of the example is intelligent personal assistant agent which can assist humans in performing their daily work, such as organizing and conducting the complex tasks on the desktop office setting [2].

Obviously, in determining priority in human daily activities, we require a proper and appropriate algorithm such as machine learning to determine which activity has the most priority. One of the example is "learning to rank" machine learning technique with many methods option to determine task priority, where will training the existing models in the ranking task [1]. Learning to rank is very useful for many application in Information Retrieval, Natural Language Processing, and Data Mining [1].

Meanwhile, determination of the ranking activity priority will be subjected for each human where they have their own ranking preferences for their daily activities. Thus, it would be very important for humans where "learning to rank" algorithm can help them to rank their daily activities with the highest ranking should be the one that should be done first. The Preference Distribution Learning (PDL) method can be used to conduct multi-label ranking by rankers/users where inconsistent ranking from different people can be solved [3].

II. Problem Statement

We are interested to apply 3 of LTR methods to solve instance, label, and object ranking using bipartite, multipartite ranking (k-partite) and learning by pairwise comparison according to their quality and accuracy [4]. Furthermore, we are interested in finding the right approach and corresponding algorithms in the process of ranking to be able to help people in determining which jobs have a higher priority than the others. Effort has been done by comparing several algorithms performed in the process of ranking which are Bipartite Ranking, k-partite Ranking, and Ranking by pairwise comparison. Our goal is to apply the appropriate algorithm in intelligence agent using Collaborative Filtering in arranging a priority in daily activity that must be done by the users.

2375-527X/17 $31.00 © 2017 IEEE
DOI 10.1109/ISPAN-FCST-ISCC.2017.67

III. Various Ranking Methods

There are various methods currently used in the process of determining the ranking using machine learning. It is not defined which method is best used in determining the priority task. Each method has its own way in determining the ranking process using models and existing training data [8]. At this section we will discuss how these methods and applications will be used in the algorithm / method used in determining the task ranking.

There are some issues in the ranking problem such as the instance ranking, ranking object and label ranking [4].

In the instance ranking that describes ranking process at an instance x and y set label, where the label is determined to have a fixed order $y1 > .. > yn$ and each instance xi is associated with a label yi. Given a set of instances as training data, where the objective of this task is to find the rankings order for a new instance [5]. In object ranking the model is given a preference information from the set of pairwise in a form and model must be able to determine the ranking order in the instance [6]. In the label ranking, the label is given instance space x and label set y, where the preference information is given in the form $yi > x\ yj$ where x indicates preference instance in yi rather than yj. A set of preference information is used as training data in a model, where the purpose of the task of the model is to find a preference ranking among all the labels for all instances. Label ranking can be an additional extension in conventional classification setting [7].

III.1. Bipartite Ranking

In the bipartite ranking, the instances will be given a label of positive or negative with the purpose of creating a score function that minimalizes the possibility of miss-ranking that maximizes the area under the ROC curve, where ROC curve is a graphical plot used to illustrate the performance of binary classifier system. Previous experiment obtained quantitative bounds for bipartite ranking in terms of a broad class of proper (composite) loss function given proper term strongly [9]. It is proven that this technique used is considerably simple and relies on properties of proper (composite) losses as elucidated recently.

In the bipartite ranking, the problem of the learning is given a training sample that consists of a set sequence of positive training examples and a set sequence of negative examples and its goal is to learn a real-valued ranking function that ranks future positive instances higher than negative ones and assigns values to positive instances higher than to negative ones.

Example of its application in information retrieval is to retrieve documents from multiple databases that are relevant to a given subject, where the training examples given to the learner consist of documents labeled as relevant (positive) or irrelevant (negative), and the end goal is to produce a list of documents containing relevant documents in the top position, and irrelevant documents in the bottom position. In other words, documents that are relevant to the search keywords will have a higher ranking than the irrelevant.

The purpose of this learning is to find a ranking function that can accurately perform the process of ranking the instances, the learning algorithm that is used should have a ranking function with minimal expected ranking error. More details are as follows, if a learning algorithm select a ranking function of a class of ranking functions F, positive instance distribution $\mathcal{D}_+$, and negative instance distribution $\mathcal{D}_-$ then the output of the ranking function $f \in \mathcal{F}$ with the expected error $\mathcal{R}_{\mathcal{D}+,\mathcal{D}-}(f)$ which is close to the best possible with class $\mathcal{F}$ [10].

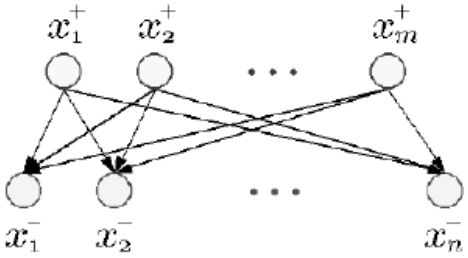

Fig. 1. Bipartite Ranking

III.2. K-Partite Ranking

K-partite algorithm used for recommending items use a diverse set of features [11]. In k-partite, the algorithm used is not a ranking system but it is a rating system, which will give a rating ranging from 1 with increment of 1. The concept is similar to bipartite which will have nodes k. Various domain has a diverse set of features available that requires a recommendation decision, for example, when giving a recommendation in music, the feature correspond to the terms in the music title, the name of singer or band, and the music genre. The most common way to incorporate features into the heterogeneous features into the random walk algorithm is by using a k-partite graph. To personalize a recommendation to every user, it takes a query centered on the random walk on the graph.

A k-partite graph is a graph that has nodes that can be partitioned into k disjoint sets, so there will be no two nodes on the same partition that is adjacent. Let $G = \{V1, V2, \ldots, Vk, E\}$ denote a k-partite graph, where each Vi is a partition of nodes and E is the set of edges. For example in the case of music recommendation using $k = 3$ where the three partitions have correspondence to the users, music, and genre.

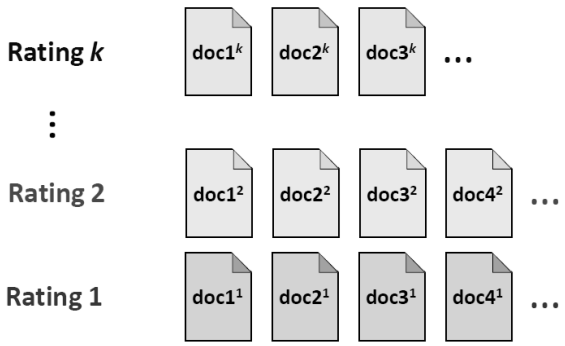

Fig. 2. K-Partite Ranking

In k-partite algorithm where the instance space $\mathcal{X}$, has a training input sample $\mathcal{S} = (\mathcal{S}1, \mathcal{S}2, \ldots, \mathcal{S}k)$, with output ranking function $f : \mathrm{X} \rightarrow \mathbb{R}$ where:

$$\mathcal{S}k = (x_1^k, \ldots, x_{nk}^k) \in \mathrm{X}^{nk} \quad (1)$$

$$\mathcal{S}2 = (x_1^2, \ldots, x_{n2}^2) \in \mathrm{X}^{n2} \quad (2)$$

$$\mathcal{S}1 = (x_1^1, \ldots, x_{n1}^1) \in \mathrm{X}^{n1} \quad (3)$$

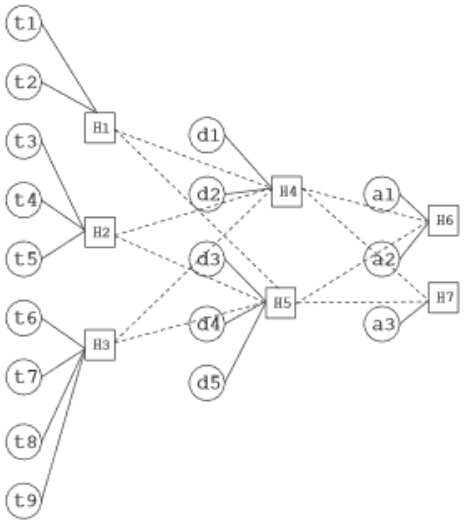

Fig. 3. Multi-way clustering on k-partite graph. t means terms, d means music file, a means the singer. The nodes labeled as H1, H2,..., H7 correspond to cluster of terms, files, and singer.

Examples of its application in the experiment [12] that has been done using the data sets OHSUMED [13] and MOVIELENS [14], where OHSUMED test collection is the benchmark data sets used to evaluate the performance of recommender systems, MOVIELENS the data sets consisting of 100,000 movie ratings provided by 943 users on 1682 movies. OHSUMED tests conducted by using a standard collaborative filtering approach. For data MOVIELENS compared its performance with collaborative filtering algorithms that exist, the result is due to the number of movies while its rating limited use integers in the range between 1 and 5, the problem is many movies have identical ratings.

III.3. Ranking by Pairwise Comparison

The most important benefit in ranking by pairwise comparison is to reduce the problem to the rank label binary several classification problem. Pairwise learning is well-known in the context of classification, because it allows one to transform a multi-class classification problem. Pairwise classification has been used in many areas for example in statistics, neural networks, support vector machine and others [15]. In general, this technique is more accurate than the method in the general classification.

For example, the problem of $m > 2$ classes $\mathcal{L} = \{\lambda 1 \ldots \lambda m\}$,, into a number of binary problems. Base learner $\mathcal{M}ij$ is trained for each pair of labels $(\lambda i, \lambda j) \in \mathcal{L}, 1 \leq i < j \leq m$; where the total number of $m(m-1)/2$ models are needed. $\mathcal{M}ij$ intended to separate the object with a label λi from those having label λj. During classification time, every prediction models will be determined for voting $\mathcal{M}ij$ between λi and λj, label with the highest value of the results of voting will be proposed as a final prediction.

Ranking by pairwise comparison overall complexity depends on the average number of preference provided for each training example. While being a quadratic in the number of labels if the full rank is given, it will be only linear for setting classification. In any case, it isn't more expensive than the constraints classification and can be cheaper if the complexity of the basic learner is super-linear.

IV. Collaborative Filtering

Collaborative filtering is one of the models used in most recommender systems. Recommender systems can create personalized recommendations for each user as required by the user to suit user preferences [16]. Personal Intelligence agent here will use the learning method to learn in advance what is preferred by the user, what is usually done by the user, and what actions are usually performed by the user. Relating it to the ranking means that every action taken by each user will be stored in the database.

The workings of collaborative filtering (CF) is making observations preferences of each user whose data was collected from the targeted user and will be compared with all the preferences of all users. CF models perform calculations to estimate the preferences of all items available, then from the items that is to be sorted by estimated preferences, eventually a subset of top items will be shown to the user as a recommendation.

The important thing to consider is how to make the process of training and evaluation of the system CF.

Evaluating the performance of the CF system is based on the collected ratings from all user that will be partitioned into a training set and a test set. Model will learn on the training set and be evaluated using a test set.

The process of learning to rank is done using supervised learning problem where each labeled training data is $\mathcal{D}$ is assumed. Training data sets $\mathcal{D}$ consists of various tasks performed by users who have inputted in accordance with the standard priorities by the various users who have a list of query-comparison pairs (qi, li) labeled with the corresponding relevance score $\mathcal{D} = \{(qi, li, yi): i \in [1..n]\}$. Where labeling yi of each task is done using three values, namely high, medium, and low with the higher priority task will be weighted higher value, so that the low will have a value of 1, medium will have a value of 2, and the high will have a value of 3, where the value will increase in accordance with the number of the task.

In evaluating the "learning to rank" will be used Normalized Discounted Cumulative Gain (NDCG) [17]. NDCG has one user-defined parameter k and two functions (gain function and discount function) that make it desirable in the ranking setting. The gain function allows a user to set the significance of each relevance level. The discount function makes items lower in the ranked list contribute less to the NDCG score. Let y become a vector of relevance values for sequences of items and π denote a permutation over the sequence of items in y, then πq is the index of the qth item in π and $y_{\pi q}$ relevance is the actual value of this item. The Discounted Cumulative Gain (DCG) for this Permutation π is defined as:

$$DCG@k(y, \pi) = \sum_{q=1}^{k} \frac{2^{y\pi_q - 1}}{\log_2(2+q)} \qquad (4)$$

Normalized Discounted Cumulative Gain (NDCG) can be defined as:

$$NDCG@k(y, \pi) = \frac{DCG@k(y,\pi)}{DCG@k(y,\pi^*)} \qquad (5)$$

V. Experiment Result

To determine the ranking algorithm which can be used in managing the priority ranking in task management to be applied in personal intelligence agent, we conducted experiments with quantitative method in gathering information of student profile (shown in figure 5) with 55 respondent students in various majors at universities in Indonesia. Each respondent will fill the profile data that is used to determine the behavior of the user and his daily activities.

A Personal Assistant shown in figure 6 will help to interact with the user. A Personal Assistant is a software agent that acts semi-autonomously for and on behalf of a user, modelling the interests of the user and providing services to the user or other users. It is unobtrusive but ready when needed and rich in knowledge about the users and their areas of work. This is the generalized notion of a Personal Agent from the agents' standards body, Foundation for Intelligent Physical Agents [18].

The functions of a Personal Agent can be as varied as carrying out one or more of the following activities: managing a user's diaries, filtering and sorting email, managing a user's desktop environment, managing a user's activities, plans and tasks, locating and delivering multimedia information, recommending entertainment, purchasing desired items, and, planning travel [19].

There is a list of possible proactive activities that an assistive agent might perform on behalf of its user to support task management, which is divided into several categories: act directly, act indirectly, collect information, and remind, notify, ask. [20].

Acting directly, the agent can perform the next step or steps of a shared task, perform or prepare for future steps of a shared task now, initiate the first step of a shared or agent task, suggest (shared) tasks the agent can take over and perform, establish a learning goal (i.e., to learn new capabilities).

Acting indirectly, the agent can suggest a user task to be delegated to a teammate, or suggest that the user offer to take on the task of a teammate, suggest a meeting to be rescheduled, suggest a lower-priority task to be postponed to free resources, suggest a task to be promoted or demoted in priority, suggest (better) ways to achieve a (shared) task, anticipate failures of (shared) tasks and look for ways to reduce the failure likelihood or reduce the impact of a failure.

Collecting information, the agent can gather, summarize information that is relevant to a user or a shared task, monitor the status of tasks delegated to a teammate, monitor and summarize resource levels and commitments, analyze possible consequences/requirements of a (shared) task.

Remind, notify, and ask, the agent can remind of the user's next step in a shared task, notify upcoming deadlines and events, ask for feedback or guidance from user, ask for clarification or elaboration of a (shared) task, monitor and filter incoming messages.

TABLE I
USER AND AGENT ACTIVITIES

Entity	Action
Agent	• Get different requests and messages from its human via user interface • Do analysis using appropriate algorithm • Do calculation of priority of tasks • Inform and display information of tasks based on the priority of task

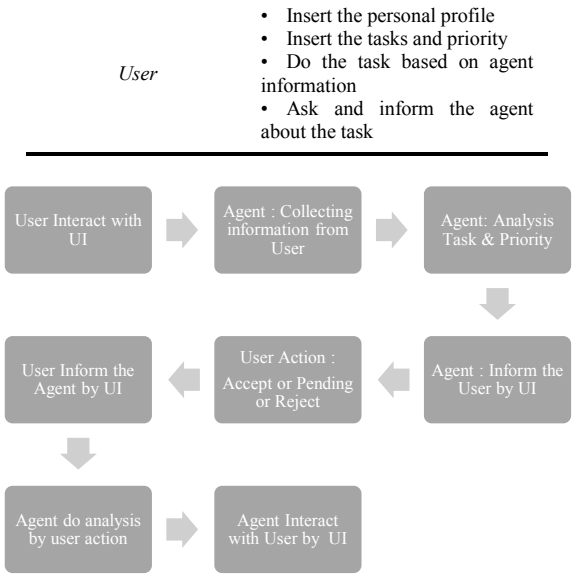

Fig. 4. Agent and User Flow Activity

Figure 4 shows the agent and user activity. First the user will interact with the user interface to input the profile, then the agent will process the input data and calculate the priority according to user preference, then the agent will process the data input from the user using the algorithm ranking (Bipartite Ranking, k-partite Ranking, and Ranking by pairwise comparison), then it will provide the information of the task by using collaborative filtering to the user which has been prepared, analyzed, and calculated by the agent.

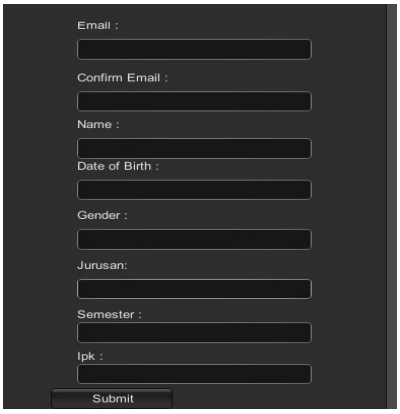

Fig. 5. Entry User Profile

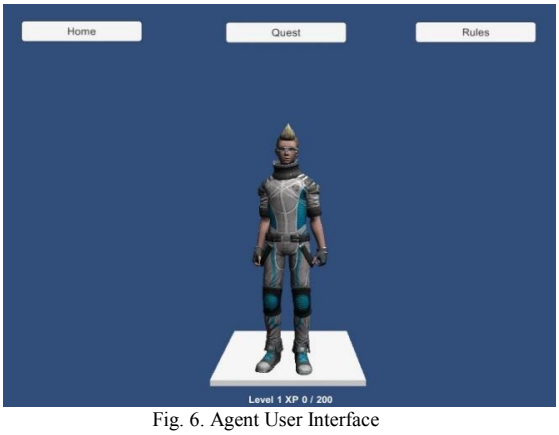

Fig. 6. Agent User Interface

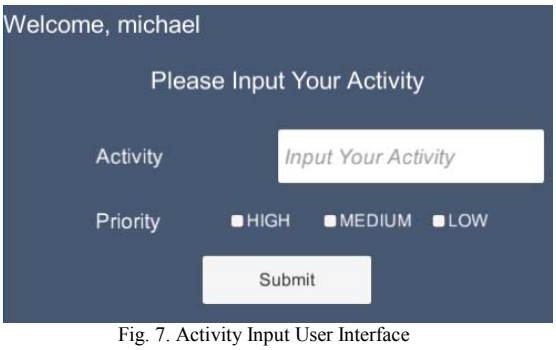

Fig. 7. Activity Input User Interface

Fig. 8. Illustrations of Activities to be performed

TABLE II
PROFILE OF RESPONDENTS

Age	Respondent	Percentage
<18	3	5%
19-21	35	64%
22-24	15	27%
>25	2	4%

Activities that have been inputted by the user will have its priority level determined between high, medium, or low. Every activity will be mapped using a ranking algorithm. Using bipartite ranking where high and medium will recorded into positive value and the low will be into negative value. Using k-partite ranking, each task will be given a value, the most common activities performed by the user will be given a higher rating than the other. Using rank by pairwise comparison, each activity will be recorded as query-pair comparison

between activities with priorities.

TABLE III
RESULT OF CALCULATE MAPPING THE PRIORITY

Activity	Bipartite Ranking	k-partite Ranking	Ranking by pairwise comparison
High	Positive	3	Activity - High
Medium	Positive	2	Activity - Medium
Low	Negative	1	Activity - Low

Evaluation for the same activities that other users have made on bipartite ranking shows that there are some activities in priority high that is under the medium priority because these activities is often appeared or performed by another user, because the priority of high and medium are in the same positive value.

In k-partite ranking every activity has been given a rating by the agent, the level of efficiency of the algorithm is quite good, with the negative side is when there is a new activity that is inputted by the user and is not in the database before, it will be given a lower rating, but the agent will still help to define and calculate to keep a high rating on the right priorities.

In rank by pairwise comparison, each activity will be sorted as high, medium, then low, the activity that is often done by the user will be placed on higher priority compared to the other activity, so any new activity can be inserted in accordance to the priorities.

From the speed of the process for a dataset, bipartite ranking processing speed is higher than the other (shown in table 4), while k-partite ranking has a lower processing speed because it must compare all of the data to the existing data, and rank by pairwise comparison would have the average speed compared with the two other algorithms because the process of indexing the data has been created using the label value to every existing query comparison.

TABLE IV
RESULT OF COMPARISON ALGORITHM

Algorithm	Speed in processing data	Accuracy
Bipartite Ranking	High	Low
k-partite Ranking	Low	High
Ranking by pairwise comparison	Medium	High

VI. Conclusion

After doing training in the training dataset and evaluate the validation dataset using NDCG, it is found that the collaborative ranking have a more accurate value / lower variance test evaluation because it uses a large dataset and smaller training dataset. There are variety of methods and algorithms that can be used to overcome the ranking problem. From algorithm Bipartite Ranking, k-partite Ranking, and Ranking by pairwise comparison used in this study, we found that for labeling more than 2 values it is not recommended to use a bipartite ranking there are many repetitive data, because later it has to be combined with other methods to get the accurate task ranking in accordance to user needed. K-partite ranking and rank by pairwise comparison are both able to be used for multi-dimensional data. K-partite ranking will generate data with a large variation, while rank by pairwise comparison will consistently follow standard ranking that has been established. In the experiments that have been conducted it is found that pairwise comparison ranking algorithm is the best solution that can be used in determining the priority task.

In the future development, the other methods can be used in the ranking problem to ensure getting the appropriate algorithm in the task priority ranking with multi-dimensional data. Agents can also combine ranking algorithm with other method to get the best result. To determine which is the best ranking algorithm that can be used in managing the priority ranking in task management to be applied in personal intelligence agent, each activity to be inputted by the user should always be processed and checked again by the agent so that the task priority informed to the user is more accurate and uses more data sets so that the training and the test set can decrease the bias in the model predictions.

References

[1] Hang LI, A Short Introduction to Learning to Rank, *The Institute of Electronics, Information and Communication Engineers, Special Section on Information-Based Induction Sciences and Machine Learning*, Vol.E94-D, No.10 October 2011.

[2] Neil Yorke-Smith, Shahin Saadati, Karen L. Myers, David N. Morley, The Design of A Proactive Personal Agent for Task Management, *International Journal on Artificial Intelligence Tools*, Vol. 21, No. 1 (2012) 1250004.

[3] Xin Geng, Longrun Luo, Multilabel Ranking with Inconsistent Rankers, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3742-3747.

[4] Johannes Fürnkranz and Eyke Hüllermeier, Preference Learning: An Introduction in Johannes Fürnkranz and Eyke Hüllermeier (Ed.), *Preference Learning, Springer*, 2010, pp. 1-17.

[5] Weiwei Cheng, Jens H¨uhn, Eyke H¨ullermeier, Decision Tree and Instance-Based Learning for Label Ranking, *proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009, pp. 161-168

[6] Johannes Fürnkranz and Eyke Hüllermeier, Preference Learning and Ranking by Pairwise Comparison in Johannes Fürnkranz and Eyke Hüllermeier (Ed.), *Preference Learning, Springer*, 2010, pp. 65-82.

[7] Shankar Vembu and Thomas Gärtner, Label Ranking Algorithms: A Survey in Johannes Fürnkranz and Eyke Hüllermeier (Ed.), *Preference Learning, Springer*, 2010, pp. 45-64.

[8] Vikas C. Raykar, Shipeng Yu, Eliminating Spammers and Ranking Annotators for Crowdsourced Labelling Tasks. *Artificial Intelligence 172*, 2008, pp. 1897–1916, Elsevier.

[9] Shivani Agarwal, Surrogate Regret Bounds for Bipartite Ranking via Strongly Proper Losses, *Journal of Machine Learning Research*, 15:1653-1674, 2014.

[10] Shivani Agarwal and Dan Roth, Learnability of Bipartite Ranking Functions, *P. Auer and R. Meir (Eds.): COLT 2005, LNAI 3559*, pp. 16–31, 2005.

[11] Xin Liu (刘 欣) and Tsuyoshi Murata, Detecting Communities in K-Partite K-Uniform (Hyper) Networks, *Journal Of Computer Science And Technology Task Management*, 26(5): 778{791 Sept. 2011.

[12] Haibin Cheng, Pang-Ning Tan, Jon Sticklen, William F. Punch, Recommendation via Query Centered Random Walk on K-partite Graph, *Seventh IEEE International Conference on Data Mining*. 2008.
[13] Hersh WR, Buckley C, Leone TJ, Hickam DH, OHSUMED: An interactive retrieval evaluation and new large test collection, *research Proceedings of the 17th Annual ACM SIGIR Conference*, pp. 192-201, 1994.
[14] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. GroupLens. An open architecture for collaborative filtering of netnews, *Proceedings of the ACM Conference on Computer-Supported Cooperative Work*, Chapel Hill, NC, 1994.
[15] Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, Klaus Brinkera, Label ranking by learning pairwise preferences, *Artificial Intelligence 172*, 2008, pp. 1897–1916, Elsevier
[16] Suhrid Balakrishnan, Sumit Chopra, Collaborative Ranking, *WSDM'12*, February 8–12, 2012, Seattle, Washington, USA, pp. 143-152.
[17] K. Jarvelin and J. Kekalainen, Cumulated gain-based evaluation of IR techniques, *ACM Trans. Inf. Syst.*, 20:422–446, October 2002.
[18] IEEE Foundation for Intelligent Physical Agents, 2014, Retrieved from http://fipa.org/specs/index.html
[19] Kumar, Subhash, et. al., a personal agent application for the semantic web, *American Association for Artificial Intelligence*, 2002.
[20] Smith, Saadati, Myers, Morley, *Proc. of 8th Int. Conf. on Autonomous Agents and Multi agent Systems* (AAMAS 2009), Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 337-344.

Authors' information

Michael Yoseph Ricky is Head of Program Game Application and Technology in Bina Nusantara University. He obtained a S.Kom. in Computer Science from Bina Nusantara University in 2009 and a M.M. in Business Management from Binus Business School in 2011. He is 2nd year Ph.D. student in Bina Nusantara University in Doctorate Computer Science, DKI Jakarta, Indonesia. He was born in Sukabumi, 13 July 1987. He is interested in Intelligence Agent, Multimedia and Games.