正準相関分析に基づく異なるドメイン間のアカウントマッチングに関する検討

呂　遠† 天方 大地† 前川 卓也† 原 隆浩† 牛 コウ††
米川 慧†† 黒川 茂莉††

† 大阪大学大学院情報科学研究科 〒 565–0871 大阪府吹田市山田丘 1-5
†† KDDI 総合研究所 〒 102–8460 東京都千代田区飯田橋 3 丁目 10 番 10 号
E-mail: †{lyu.yuan,amagata.daichi,maekawa,hara}@ist.osaka-u.ac.jp,
††{ha-niu,ke-yonekawa,mo-kurokawa}@kddi-research.jp

あらまし オンラインショッピングは多くの消費者に浸透しており，推薦システムを用いて消費者に商品を提示する Web サイトも増加している．推薦システムは単一ドメインにおける情報を基にしているため，推薦に必要な情報が不足しがちという問題がある．この問題を解決するアプローチとして，他のドメインの情報を利用することが考えられる．これを実現するためには，異なるドメイン間で同一のユーザを特定する必要がある．本稿では，テキスト情報を用いてユーザアカウントをベクトル化し，正準相関分析を用いてベクトルのマッチングを計算する手法を提案する．実験により，提案手法はランダムマッチングや線形回帰よりも高精度であることを示す．

キーワード account matching, canonical correlation analysis

1 Introduction

Online shopping has nowadays become more and more popular. Because of abundant items, customers may suffer from selecting items. To avoid such situations, online shopping systems employ item recommendation. Normally, recommender systems utilize content-based recommendations [1]. Based on item information of user histories, user preference profiles are generated. Firstly, some features of items are used to represent the items. Secondly, some history information of users is used to generate user preference profiles. Finally, by comparing user preference profiles and features of items, the system recommends items which are most related to users [2].

Content-based recommendations are a part of collaborative filtering. Many tasks use collaborative filtering because that people often get the best recommendations from someone with similar tastes and interests to themselves. Based on this phenomenon, collaborative filtering is a technique that match people with similar interests and making recommendations [3]. To be more specific, collaborative filtering is the method of filtering for information or patterns utilizing techniques involving collaboration among multiple agents, viewpoints, data sources, etc [4]. When tasks use collaborative filtering, it usually has a large data set. Collaborative filtering methods have been used in many situations. For example of used data, it includes sensing and monitoring data in mineral exploration, environmental sensing over large areas and multiple sensors, financial data, such as financial service institutions that integrate many financial sources; or in electronic commerce and web applications where the focus is on user data [5].

The above recommender systems are usually based on information from their own domains. That is to say, data for analyzing is limited. For instance, we assume that a customer purchased a diaper in a shopping site. Accordingly, the customer would have an interest in baby products, thus the recommender system of the shopping site would put baby products into his/her recommendation list. However, except for baby products, no items can be added into the list. The reason is that the recommender system has no idea about his/her preference in other situation, resulting in a lack of information. Using information from other domains may solve this problem.

A user usually has multiple accounts among different domains. Besides an account of the shopping site, we assume that the user has several accounts of other domains. It is useful for recommender systems if it can obtain information from other domains. On the other hand, for a certain user, the shopping site usually does not have a knowledge about the user's account ID of other domains. As a result, although there is abundant information on a certain user in other domains, it is impossible for recommender systems to utilize them. Therefore, matching account IDs of the same user in different domains would be a key solution for this problem.

In this paper, we consider the following situation. A user has accounts of two domains. One of the domains

is an e-commerce site. In this domain, there are descriptions (mainly text) of products and purchase histories of users. The other domain is an Ad-Network service. The Ad-Network service puts advertisement on some webpages, thus can record URLs that users have accessed. Our task is to match accounts of the same user in these two domains, by only using the above information.

The proposed method comprises the following steps. Firstly, we make vectors for accounts based on text information of users. Secondly, since vectors from different domains are difficult to compare, we map these vectors into a shared space. Eventually, after vectors are mapped into the shared space, we match accounts according to the similarity.

In our preliminary experiments, we tested on canonical correlation analysis compared with linear regression. The results show that canonical correlation analysis achieved better precision than linear regression and random selection. Our further experiments show that cosine similarity is more proper than Euclidean distance.

The rest of this paper is organized as follows. In Section 2, we introduce some related work. In Section 3, we describe our problem and situation. In Sections 4 and 5, our method is described. In Section 6, we introduce our experiments and results.

2 Related Work

Since cross-domain matching use information from other domains, it can be regarded as a problem of domain adaptation [6]. Domain adaptation is a problem utilizing machine learning and transfer learning. This problem occurs when we aim at learning from a source data distribution and a model with good performance on a different but related target data distribution. For instance, a representative task is in picture recognition. In this task, the source distribution can be a model of human-being recognition, and it need to be adapted into a target distribution of animal recognition.

There are several types of domain adaptation. They differ in the information given in the target distribution. Unsupervised domain adaptation is that the learning sample contains a set of labeled source examples, while a set of unlabeled source examples and an unlabeled set of target examples are given in learning and prediction [7]. In supervised domain adaptation, we have all the samples labeled [8]. Since labeled examples are difficult to obtain, it is not realistic in most situations. To solve this problem, semi-supervised domain adaptation is utilized. In this situation, a small set of labeled examples in target domain are used [9].

3 Problem Statement

In this paper, we consider the following problem. There are two domains. One of the domains is an e-commerce site. We call it Domain A in this paper. The other domain is an Ad-Network service. We call it Domain B in this paper.

For Domain A, many items which belong to several categories are selling in a shopping site. For every item, several kinds of information is given, such as price, text descriptions, category, selling duration, etc. Meanwhile, information about users is also given. It includes user's profile, such as gender, mail address, username, etc. Besides, purchase histories of users are also available, such as purchase item, time, amount, price, etc. For Domain B, when opening a URL including an advertisement provided by this Ad-Network service, the advertisement will be displayed on its web page. When users access this advertisement, some information on users is recorded, such as user's IP address, access time, accessed URL, etc.

We assume that a user has accounts of the two domains. This user has only one account in Domain A. Conversely, this user can have more than one account in Domain B. This is because that accounts in Domain B automatically update with a certain period of time, if the user constantly receives advertisements. That is to say, the user automatically gets a new account by the certain period of time. The problem in this paper is to match accounts belonging to the same user, while only using information listed above. Additionally, we only utilize purchase histories in Domain A and access log of URLs in Domain B, to solve this problem. The structure of our account domains is shown in Figure 1.

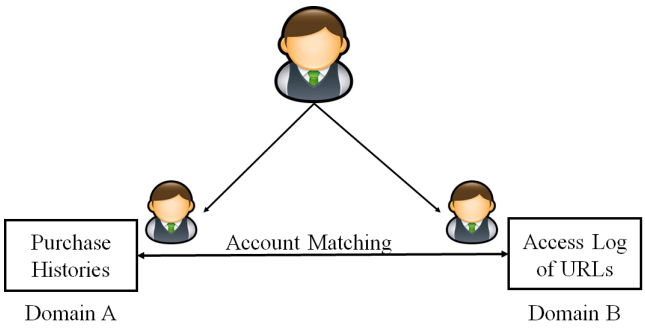

図 1 Account structure

4 Vector Generating

For account matching, comparing the similarity of vectors of users between different domains may be the most realistic method. Considering purchase histories and access log of URLs, the most meaningful information is articles involved.

For example, the articles in purchase histories can be descriptions of items, while the articles in URLs can be news. After we have obtained articles, we use them to generate user features.

4.1 Obtaining of articles

Access logs of URLs and purchase histories involve some articles.We mainly use crawling to obtain these articles with special tags or in specific parts.

In Domain A, we have information on purchase histories of users. We assume that items purchased by a user can represent the features of the user. Therefore, we use description articles of each item that the user has purchased, to obtain user's features. As for items in Domain A, item title, promotion slogan and detailed description are combined to be their articles.

In Domain B, we have access log of users, in which accessed URLs are recorded. For most URLs, web pages include articles, which could represent preferences of users, thus can be utilized to get features of them. In HTML source file, since articles are usually involved in tag a and tag p [10], we do crawling based on tag a, tag p, as well as page title. In web pages, tag a and tag p are usually the body part, thus include the main articles. The article of a web page is combined by these three parts.

4.2 Converting articles to vectors

After we have obtained articles from these two domains, we convert the articles into vectors. We utilize MeCab [11] for text segmentation, and make vectors by Doc2Vec [12]. The two procedures are shown below.

MeCab is an open-source text segmentation library for use with text written in the Japanese language. It can analyze and segment a sentence into its parts of speech. Doc2Vec requires a list of words to make vectors, for which a long article is not available. Therefore, it is necessary to divide the article into several words by MeCab.

The goal of Doc2Vec is to create a numeric representation of a document, regardless of its length. It learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents. Based on Word2Vec, it represents each document by a dense vector which is trained to predict words in the document [13].

4.3 Vectors of user account

After we have converted article into vectors, we make a vector for a user's account based on article vectors. In Domain A, many users have purchased more than one item. In Domain B, all the users have accessed more than one URL. That is to say, both in Domain A and Domain B, every account can have more than one article vector. Therefore, it is necessary to get one account vector from several article vectors.

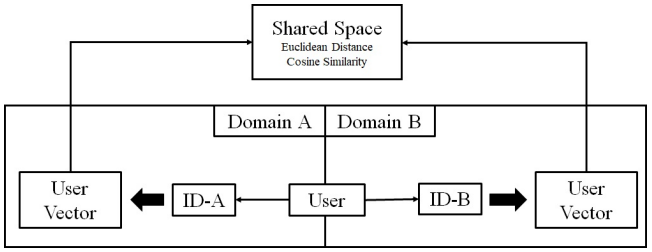

図 2 Matching System: a user has account in both Domain A and Domain B. After obtaining vectors of account, we map them into a shared space for matching

We assume that account r has a set of s articles: $\boldsymbol{A}_r = \{a_{r1}, a_{r2}, a_{r3}, ..., a_{rs}\}$, where a_{ri} is the i-th article of account r. Additionally, by converting articles into vectors, set of articles $\boldsymbol{A}_r$ has a set of vectors converted: $\boldsymbol{T}_r = \{\boldsymbol{t}_{r1}, \boldsymbol{t}_{r2}, \boldsymbol{t}_{r3}, ..., \boldsymbol{t}_{rs}\}$, where $\boldsymbol{t}_{ri}$ is the vector of the i-th article of account r. Therefore, the vector of account r is calculated as:

$$\boldsymbol{t}_r = \frac{1}{s}\sum_{k=1}^{s}\boldsymbol{t}_{rk} \tag{1}$$

That is to say, for a certain account, we use the average of all article vectors as account vector.

5 Account Matching

Since we have obtained account vectors of two different domains, we then match accounts of the same user based on these account vectors. A common solution is to match by similarity of vectors. However, since these vectors are generated from different domains, they would have different feature distributions. Therefore, it is impossible to use the similarity of these vectors directly, and it is necessary to map these vectors in a shared space. In this section, we firstly map account vectors into a shared space, and then calculate the similarity of the mapped vectors. Our matching system is shown in Figure 2.

A common solution is to calculate a conversion matrix among the two domains using linear regression. By using linear regression, it is possible to use straight lines to fit sample points [14]. We assume that an n-dimensional vector X and an m-dimensional vector Y, where $\mathrm{X} = [x_1, x_2, x_3, ..., x_n]^\top$ and $\mathrm{Y} = [y_1, y_2, y_3, ..., y_m]^\top$. Both X and Y have several features, thus we want to analyze the relationship between X and Y. If $\mathrm{X} \in \mathbb{R}^n$ and $\mathrm{Y} \in \mathbb{R}^m$, we can establish an equation $\mathrm{X} = \mathrm{WY}$ as:

$$\begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_m \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & ... & w_{1n} \\ w_{21} & w_{22} & ... & w_{2n} \\ ... & ... & ... & ... \\ w_{m1} & w_{m2} & ... & w_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_n \end{bmatrix} \tag{2}$$

where W is a conversion matrix for linear regression:

$$\mathrm{W} = \begin{bmatrix} w_{11} & w_{12} & ... & w_{1n} \\ w_{21} & w_{22} & ... & w_{2n} \\ ... & ... & ... & ... \\ w_{m1} & w_{m2} & ... & w_{mn} \end{bmatrix}. \tag{3}$$

Since $y_i = w_i^\top x$, it is necessary to train m times to get w_i of amount m, where y_i is the i-th feature of y and w_i is the i-th row in W. However, there is a problem that each feature of Y is correlated to every feature of X, but has no correlation to other features of Y itself. To solve this problem, we utilize Canonical Correlation Analysis (CCA), permitting to calculate multiple objective variables simultaneously.

5.1 Canonical Correlation Analysis

Canonical correlation analysis can be used as a method to find out the correlation of two pairs of vectors. If we have two vectors $\mathrm{X} = (x_1, x_2, x_3, ..., x_n)$ and $\mathrm{Y} = (y_1, y_2, y_3, ..., y_m)$, canonical correlation analysis will use linear calculations to find out the combinations of X and Y which have maximum correlation with each other, and give a solution for the maximum correlation [15].

We assume that X and Y are account vectors in different domains of the same user, canonical correlation analysis seeks vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ for the following situation [16]:

$$\max \quad \rho = \mathrm{corr}(\boldsymbol{a}^\top \mathrm{X}, \boldsymbol{b}^\top \mathrm{Y}) \tag{4}$$

By maximize the correlation between $\boldsymbol{a}^\top \mathrm{X}$ and $\boldsymbol{b}^\top \mathrm{Y}$, we can convert paired vectors X and Y into similar vectors X' and Y', as a result of mapped into a shared space, where $\mathrm{X}' = \boldsymbol{a}^\top \mathrm{X}$ and $\mathrm{Y}' = \boldsymbol{b}^\top \mathrm{Y}$ [17]. Therefore, we can calculate the similarity of vectors in the two different domains.

In canonical correlation analysis, the converted vectors X' and Y' usually have the same number of dimensions. The number of dimensions can be selected according to our needs. The more the number of dimensions is, the more features the vectors would have. However, more number of dimensions may require longer time while training models [18].

5.2 Similarity Calculation

After we have got converted vectors, we can match vectors by similarity. Normally, when calculating the similarity of vectors, Euclidean distance and cosine similarity are usually used. In mathematics, Euclidean distance is the ordinary straight-line distance between two points in Euclidean space. The Euclidean distance between X and Y is calculated as:

$$\mathrm{distance} = d(\mathrm{X}, \mathrm{Y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{5}$$

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. For vectors X and Y, the cosine similarity is calculated as:

$$\mathrm{similarity} = \cos(\theta) = \frac{\mathrm{X} \cdot \mathrm{Y}}{||\mathrm{X}||||\mathrm{Y}||} \tag{6}$$

6 Experiment and Result

Our experiment is based on real-world database of two different domains. A user has accounts in both Domain A and Domain B. In Domain A, every user has only one account. In Domain B, one user has at least one account.

6.1 Data set

As for our dataset, we utilize data collected from May 11$^{\mathrm{th}}$, 2017 to March 31$^{\mathrm{th}}$, 2018. In the dataset, we have 82 thousand users. Therefore, in Domain A, there are 82 thousand accounts, for which there is at least one purchase record. Meanwhile, in Domain B, since one user could have more than one account, there are 92 thousand accounts, for which there are several URL access records.

For the URLs accessed in Domain B, we have calculated the access time of each URL. Since accessed URLs have an amount of more than 11 million, it is almost impossible to do crawling to get all articles. Therefore, we just crawled articles of URLs with more than 1,000 accesses.

After we have obtained all needed articles from the two domains, we make vectors by MeCab and Doc2Vec for articles. Vectors are made to be 256-dimension.

6.2 Evaluation methodology

For the evaluation metric of our experiment, we use Precision@R. It stands for the matching performance by using the precision with which the true target instance is included in a set of R proportion of candidates instances, $S(R)$, found by each method. More formally, the precision is given by:

$$\mathrm{Precision@}R = \frac{1}{N_{te}} \sum_{i=1}^{N_{te}} \delta(t_i \in S_i(R)) \tag{7}$$

where N_{te} is the number of test instances in the target domain, t_i is the i-th true target instance, $S_i(R)$ is R candidate instances of the i-th source instance and $\delta(\cdot)$ is the binary function that returns 1 if the argument is true, and 0 otherwise.

In our experiments, we use 70% of data to train our model

and the rest 30% for test. For each different condition, we do experiment for five times with different allocation of training and testing data. After all the experiments of the same condition are completed, we average the result of each experiment as the final result.

6.3 Similarity calculation

Since we have two options (cosine similarity and Euclidean distance) to calculate similarity, we want to know which is suitable for our experiment. We have tested in many conditions, and find out cosine similarity is more suitable. An instance is given in Figure 3, where the dimension of mapped vectors is 9.

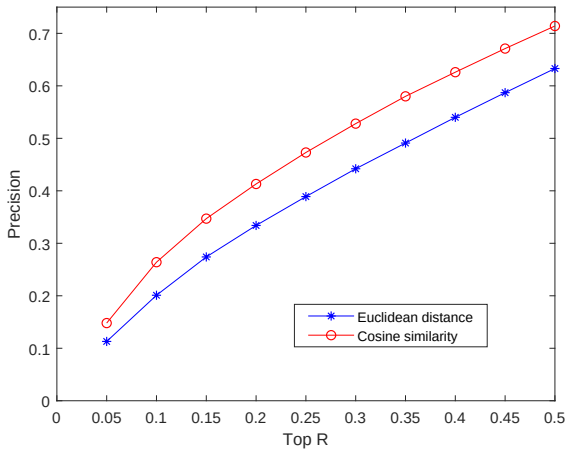

図 3 Similarity calculation: for each top R proportion of candidates, CCA's precision of Euclidean distance and cosine similarity

6.4 Dimension of mapped vectors

When training the model using canonical correlation analysis, we need to set the parameter of the dimension of mapped vectors. A larger dimension means more user features, but it also has some negative effects, such as more calculation consumption and feature noise. Since we have known that cosine similarity has better performance, we find the best dimension using cosine similarity.

We tested on the parameter of dimension of mapped vectors from 1 to 15. All results of canonical correlation analysis are shown in Tables 1 and 2, using Euclidean distance and cosine similarity. Bold values are the best performance under a certain dimension of mapped vectors. When dimension of mapped vectors is near 9, the performance has a peak value.

6.5 Comparison with linear regression

Linear regression is a baseline method in our problem. For comparison, we tested on it. Our method used 9 dimensions for account at vectors. Both the two experiments used cosine similarity. The result is shown in Figure 4. It is obvious that our method outperforms the baseline method. It can be regarded as high performance, but still can be improved.

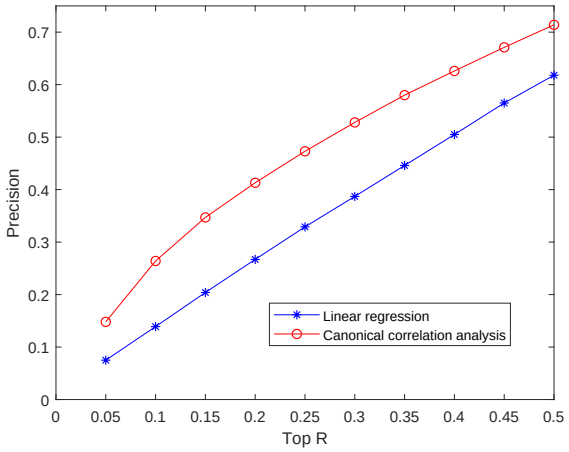

図 4 Comparison with linear regression: for each top R proportion of candidates, CCA's precision compared with linear regression

7 Conclusion

In this paper, we developed a system to match accounts in different domains of the same user. In order to use vector similarity to match accounts, we generate vectors based on articles which are related to users by MeCab and Doc2Vec. When matching accounts, we find that canonical correlation analysis and cosine similarity have better performance. The results are better than random and linear regression. It indicates that our method is realistic in our experiment situation.

As for future work, we plan to utilize some non-linear methods, such as deep canonical correlation analysis [19] and kernel canonical correlation analysis [20].

Acknowledgment

This research is partially supported by JST CREST Grant Number J181401085.

文　献

[1] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proc. 10th ACM Conference on Recommender Systems*, pages 191–198, 2016.

[2] Wenyi Huang, Zhaohui Wu, Chen Liang, Prasenjit Mitra, and C. Lee Giles. A neural probabilistic model for context based citation recommendation. In *Proc. 29th AAAI Conference on Artificial Intelligence*, pages 2404–2410, 2015.

[3] Dongsheng Li, Chao Chen, Qin Lv, Hansu Gu, Tun Lu,

表 1　Result on Euclidean distance of number of dimension from 1 to 15

R	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.05	0.075	0.088	0.094	0.101	0.104	0.108	0.111	0.112	0.113	0.113	0.113	0.113	0.113	0.113	**0.114**
0.1	0.142	0.165	0.177	0.187	0.193	0.198	0.201	0.201	0.201	**0.202**	0.202	0.201	0.201	0.201	0.200
0.15	0.205	0.236	0.250	0.261	0.267	0.272	0.274	0.274	0.274	**0.274**	0.273	0.272	0.271	0.271	0.270
0.2	0.263	0.300	0.314	0.325	0.331	0.334	**0.336**	0.335	0.334	0.333	0.332	0.332	0.331	0.330	0.329
0.25	0.316	0.359	0.373	0.383	0.388	**0.391**	0.391	0.391	0.389	0.388	0.387	0.386	0.385	0.383	0.382
0.3	0.369	0.416	0.429	0.438	0.443	**0.444**	0.443	0.442	0.442	0.440	0.439	0.438	0.436	0.435	0.433
0.35	0.421	0.470	0.483	0.490	0.494	**0.494**	0.494	0.492	0.491	0.490	0.489	0.488	0.487	0.485	0.484
0.4	0.473	0.522	0.534	0.540	0.543	**0.543**	0.542	0.541	0.540	0.538	0.536	0.535	0.535	0.533	0.532
0.45	0.523	0.572	0.584	0.589	0.591	**0.591**	0.590	0.589	0.587	0.585	0.584	0.583	0.582	0.580	0.578
0.5	0.572	0.622	0.633	0.636	**0.637**	0.637	0.636	0.634	0.633	0.631	0.630	0.629	0.628	0.626	0.625

表 2　Result on cosine similarity of number of dimension from 1 to 15

R	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.05	0.075	0.102	0.120	0.130	0.135	0.143	0.145	0.148	0.148	0.149	**0.150**	0.150	0.149	0.149	0.150
0.1	0.142	0.207	0.229	0.240	0.247	0.255	0.258	0.259	0.261	**0.261**	0.261	0.261	0.261	0.261	0.261
0.15	0.205	0.290	0.314	0.325	0.332	0.339	0.343	0.345	**0.347**	0.346	0.346	0.346	0.346	0.345	0.345
0.2	0.263	0.356	0.381	0.392	0.399	0.406	0.410	0.412	**0.413**	0.413	0.412	0.412	0.412	0.412	0.410
0.25	0.316	0.415	0.440	0.452	0.459	0.466	0.470	0.472	**0.472**	0.472	0.472	0.472	0.471	0.471	0.471
0.3	0.369	0.471	0.496	0.509	0.516	0.522	0.526	0.527	**0.528**	0.528	0.527	0.526	0.527	0.526	0.525
0.35	0.421	0.524	0.550	0.562	0.568	0.574	0.577	0.578	**0.580**	0.579	0.579	0.579	0.578	0.577	0.576
0.4	0.473	0.575	0.600	0.610	0.617	0.622	0.625	0.626	0.626	0.626	**0.626**	0.626	0.625	0.624	0.624
0.45	0.523	0.624	0.646	0.657	0.661	0.667	0.670	0.671	0.671	0.671	**0.671**	0.670	0.670	0.670	0.669
0.5	0.572	0.669	0.690	0.700	0.704	0.710	0.712	0.714	**0.714**	0.713	0.712	0.712	0.712	0.712	0.711

Li Shang, Ning Gu, and Stephen M. Chu. Adaerror: An adaptive learning rate method for matrix approximation-based collaborative filtering. In *Proc. 2018 World Wide Web Conference*, pages 741–751, 2018.

[4] Shakiba Rahimiaghdam, Pinar Karagoz, and Alev Mutlu. Personalized time-aware outdoor activity recommendation system. In *Proc. 31st Annual ACM Symposium on Applied Computing*, pages 1121–1126, 2016.

[5] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. 10th International Conference on World Wide Web*, pages 285–295, 2001.

[6] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Vaughan. A theory of learning from different domains. volume 79, pages 151–175, 2010.

[7] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proc. 2017 International Conference on Machine Learning*, 2017.

[8] Daniel Garcia-Romero and Alan McCree. Supervised domain adaptation for i-vector based speaker recognition. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4047–4051, 2014.

[9] Hal Daumé III, Abhishek Kumar, and Avishek Saha. Frustratingly easy semi-supervised domain adaptation. In *Proc. the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 53–59, 2010.

[10] Gengxin Miao, Junichi Tatemura, Wang-Pin Hsiung, Arsany Sawires, and Louise E Moser. Extracting data records from the web using tag path clustering. In *Proc. the 18th International Conference on World Wide Web*, pages 981–990, 2009.

[11] `http://taku910.github.io/mecab/`.

[12] Jey Han Lau and Timothy Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016.

[13] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proc. 2014 International Conference on Machine Learning*, pages 1188–1196, 2014.

[14] George AF Seber and Alan J Lee. *Linear regression analysis*, volume 329. 2012.

[15] David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

[16] Thomas R Knapp. Canonical correlation analysis: A general parametric significance-testing system. *Psychological Bulletin*, 85(2):410, 1978.

[17] Patricia Cohen, Stephen G West, and Leona S Aiken. *Applied multiple regression/correlation analysis for the behavioral sciences*. 2014.

[18] Francis R Bach and Michael I Jordan. A probabilistic interpretation of canonical correlation analysis. Technical report, University of California, Berkeley, 2005.

[19] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proc. 2013 International Conference on Machine Learning*, pages 1247–1255, 2013.

[20] Pei Ling Lai and Colin Fyfe. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems*, 10(05):365–377, 2000.