PERSONALIZED WORD REPRESENTATIONS CARRYING PERSONALIZED SEMANTICS LEARNED FROM SOCIAL NETWORK POSTS

Zih-Wei Lin, Tzu-Wei Sung*, Hung-Yi Lee, and Lin-Shan Lee*

National Taiwan University
{r04942111, b03902042, hungyilee}@ntu.edu.tw, lslee@gate.sinica.edu.tw

ABSTRACT

Distributed word representations have been shown to be very useful in various natural language processing (NLP) application tasks. These word vectors learned from huge corpora very often carry both semantic and syntactic information of words. However, it is well known that each individual user has his own language patterns because of different factors such as interested topics, friend groups, social activities, wording habits, etc., which may imply some kind of personalized semantics. With such personalized semantics, the same word may imply slightly differently for different users. For example, the word ”Cappuccino” may imply ”Leisure”, ”Joy”, ”Excellent” for a user enjoying coffee, by only a kind of drink for someone else. Such personalized semantics of course cannot be carried by the standard universal word vectors trained with huge corpora produced by many people. In this paper, we propose a framework to train different personalized word vectors for different users based on the very successful continuous skip-gram model using the social network data posted by many individual users. In this framework, universal background word vectors are first learned from the background corpora, and then adapted by the personalized corpus for each individual user to learn the personalized word vectors. We use two application tasks to evaluate the quality of the personalized word vectors obtained in this way, the user prediction task and the sentence completion task. These personalized word vectors were shown to carry some personalized semantics and offer improved performance on these two evaluation tasks.

***Index Terms*—** Distributed Word Representation, Personalized Word Vectors, Skip-gram Model, Social Network Data

1. INTRODUCTION

In many natural language processing tasks, a word is a discrete token and usually represented as a vector with one-hot encoding, where the dimensionality of the vector is the vocabulary size and the position of one corresponds to the index of the word in the vocabulary. One well-known limitation of such one-hot encoding method is that it says nothing regarding the semantic relationship between words. Various approaches to learn distributed word representations have been proposed to partly solve this problem [1–7]. Word2vec [8, 9] is an unsupervised approach which has been shown to offer word representations carrying plenty of syntactic and semantic information, and found very useful in many applications such as identifying words with given semantics [10–12].

On the other hand, it is well known that each individual user has his own language patterns because of different factors such as interested topics, friend groups, social activities, wording habits, etc., which may imply some kind of personalized semantics. With such personalized semantics, the same word may imply slightly differently for different users. For example, the word ”Cappuccino” may imply ”Leisure”, ”Joy”, ”Excellent” for a user enjoying coffee, by only kind of drink for someone else. Such personalized semantics will certainly be helpful in improving the performance of the various natural language processing applications for each individual user. In fact personalization has been an important trend for many Internet services today, for example personalized retrieval [13–17], personalized learning [18, 19], and personalized recommendation systems [20–25]. An important step towards such personalized services is the personalized language processing [26–33]. However, the standard universal word representations learned from huge corpora produced by many people are certainly not able to describe personalized semantics. As a result, word representations adapted to different users is definitely a good step toward such a direction.

Substantial works have been reported on different ways for representing words as vectors to deal with different natural language processing problems [34–41], but much less works were reported to investigate the mismatch between the universal word representations learned from general corpora and the personalized corpus produced by different individual users. One good reason for this is perhaps the difficulty in collecting personalized corpus. However, this situation has changed in recent years. Nowadays, many individuals post large quantities of texts over social networks, which can be a good source for constructing personalized corpus. In a series of efforts towards this direction, we implemented a cloud-based applica-

[*] Fist author and second author are equal in contribution.

tion to collect personalized linguistic data produced by many individual users from the social media. The data collected in this way are usually casual and short, but may carry plenty of personalized semantics.

In this paper, we proposed two approaches based on the skip-gram model of Word2vec to obtain personalized word vectors using individual social posts. The first approach simply tries to retrain the universal Word2vec model with the personalized corpus, while the second approach tries to insert an adaptive linear transformation layer within the skip-gram model. In both approaches, an universal Word2vec model was first trained with the background corpora produced by many people, then this Word2vec model was fine-tuned to be adapted to the personalized corpus. We used two different tasks to evaluate the quality of the obtained personalized word vectors, with which improved performance was obtained. We also found the second approach of inserting an adaptive linear transformation layer performed better.

2. PROPOSED APPROACH

We first illustrate the scenario of personalized word vectors in Subsection 2.1, and briefly summarize the training criterion of skip-gram model in Subsection 2.2. We then describe the two proposed approaches to train personalized word vectors for each individual user in Subsections 2.3 and 2.4.

2.1. Scenario of Personalized Word Vectors

The scenario of personalized word vectors is shown in Fig. 1. Universal background corpora including numerous articles collected from different domains are first used to train a set of universal background word vectors using the skip-gram model. For each individual user, we then collect his (or her) social posts from the social media taken as the personalized corpus, with which we tune the universal background word vectors to obtain the personalized word vectors. The personalized word vectors are based on the same lexicon as used in the background corpora, but they are different in vector representations. These personalized word vectors are then used in various natural language processing applications.

2.2. Skip-gram Model

In this work, we choose the skip-gram model to train the word vectors. Given a sequence of training words $w_1, w_2, ..., w_T$, and the contexts w_j for each word $w_t, t-b \leq j \leq t+b, b \neq 0$, where the context window length is $2b+1$, the goal of the skip-gram model as shown in Fig. 2 (a) is to find the parameters $\mathcal{W}, \mathcal{W}'$ so as to maximize the log of the conditional probability

$$\arg\max_{\mathcal{W},\mathcal{W}'} \sum_{t=1}^{T} \sum_{j=t-b,b\neq 0}^{t+b} \log p(w_j | w_t; \mathcal{W}, \mathcal{W}') \quad . \tag{1}$$

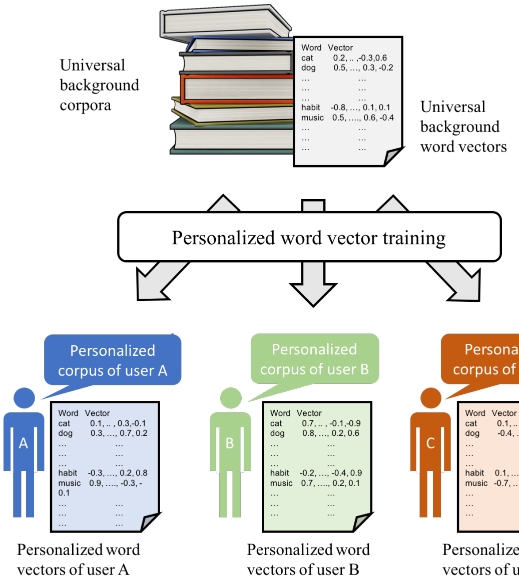


Fig. 1. Scenario of personalized word vectors.

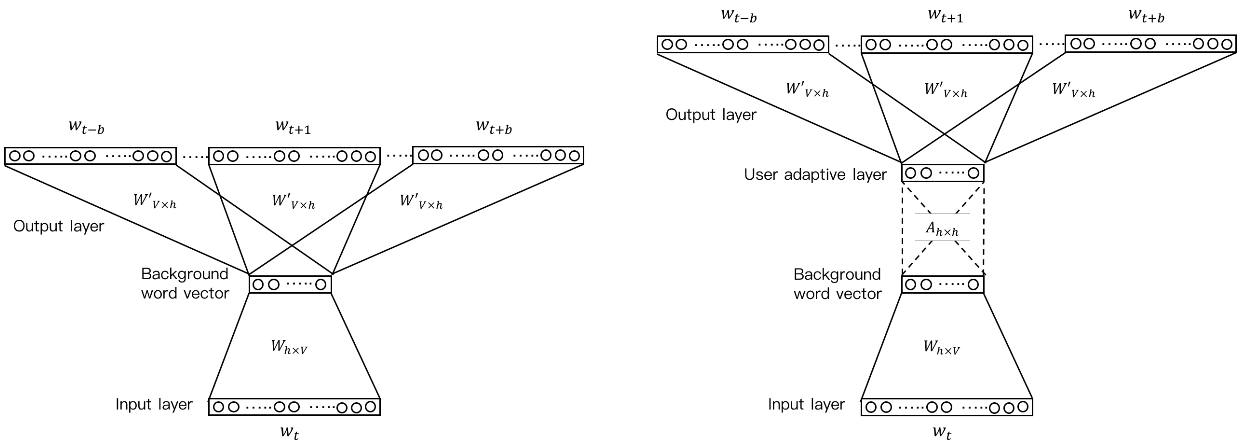
To approximate the conditional probability $p(w_j | w_t; \mathcal{W}, \mathcal{W}')$ in Eq.(1), negative sampling can be used to optimize the model parameters $\mathcal{W}, \mathcal{W}'$ so as to minimize the objective function for each word w_t, $J(w_t)$, defined as

$$J(w_t) = -\log\left(\frac{1}{1+e^{-v'_{w_j} \cdot v_{w_t}}}\right) - \sum_{neg} \log\left(1 - \frac{1}{1+e^{-v'_{w_{neg}} \cdot v_{w_t}}}\right), \tag{2}$$

where v_{w_t} and v'_{w_j} are the vector representations for the target word w_t and contexts w_j, and v'_{w_j} is also called positive example. neg is a function which randomly samples words w_{neg} from the whole corpus, which are different from w_j and called negative examples, according to their word frequencies. Empirically, w_{neg} is picked from the distribution $U(w)^{\frac{3}{4}}/Z$, where $U(w)$ is the unigram distribution of the corpora, and Z is a normalization constant. The goal of this objective function $J(w_t)$ in Eq.(2) is to increase the quantity of $v'_{w_j} \cdot v_{w_t}$ for word-context pairs, and decrease $v'_{w_{neg}} \cdot v_{w_t}$ for randomly sampled irrelevant pairs. Therefore, vectors of words that share many contexts will be clustered together, and as a result these vectors can exhibit some semantics including the linear structure that makes analogical reasoning possible.

2.3. Approach 1 - Retrain the Model

With the universal background word vectors trained with the skip-gram model as mentioned above using the universal background corpora, for each user, we retrain the background word vectors with the personalized corpus for each user with the same model, but simply fine-tune the parameters of the model to fit the personalized corpus. The fine-tuned word vectors are the personalized word vectors.



(a) Skip-gram model

(b) Inserting a user adaptive layer to the skip-gram model

Fig. 2. Skip-gram model and inserting a user adaptive layer to the skip-gram model.

2.4. Approach 2 - Inserting a User Adaptive Layer

This approach is shown in Fig. 2 (b), which is very similar to the skip-gram model in Fig. 2 (a), except we insert a user adaptive layer, which is a linear layer, between the original hidden and output layers.

As shown in Fig. 2 (b), we first train the background word vectors with the standard skip-gram model. This includes the input layer weights $\mathcal{W}_{h\times V}$ and output layer weights $\mathcal{W}'_{V\times h}$, where h is the dimensionality of the word vectors, and V is the vocabulary size. These weights are trained with the universal background corpora. Then the additional user adaptive layer is inserted into the model with weights $\mathcal{A}_{h\times h}$, where the weights $\mathcal{A}_{h\times h}$ are randomly initialized. We now fix the parameters for the universal background model, $\mathcal{W}_{h\times V}$ and $\mathcal{W}'_{V\times h}$, but only $\mathcal{A}_{h\times h}$, or the user adaptive matrix, is fine-tuned for each user based on the personalized corpus. The training algorithm is the same as that in Section 2.2. We wish to find the best parameters $\mathcal{A}$ to maximize the conditional probability

$$\arg\max_{\mathcal{A}} \sum_{t=1}^{T} \sum_{j=t-b, b\neq 0}^{t+b} \log p(w_j|w_t; \mathcal{A}, \mathcal{W}, \mathcal{W}') \tag{3}$$

for each individual user. We finally multiply the background word vectors $\mathcal{W}_{h\times V}$ by the adaptive weights $\mathcal{A}_{h\times h}$ to obtain the personalized word vectors for each individual user.

3. EVALUATION TASKS

Given a set of word representations or embeddings $\{v_1, \dots, v_V\}$ for the corresponding vocabulary $\{w_1, \dots, w_V\}$, where the vector representation of w_i is v_i, where V is the vocabulary size, so $M = \{(w_1, w_2, \dots, w_V) : (v_1, v_2, \dots, v_V)\}$ is a mapping or the word representation being considered. Our goal is to evaluate whether M is a "good" representation, or a "good" set of embeddings.

We introduce here two tasks to perform the evaluation.

3.1. User Prediction

Assume a user produces a document of N sentences, $D = \{s_1, s_2, \dots, s_N\}$, where s_n is the n-th sentence, $1 \leq n \leq N$. We wish to predict the user producing this document out of a group of users $U = \{u_1, u_2, \dots\}$, each user u having a personalized word representation or mapping M_u.

3.1.1. Document Classification Approach

The approach proposed to perform document classification with respect to different domains using Word2vec [42] by maximizing the log likelihoods of words and their contexts can be used here, except each domain corresponds to a user. This is parallel to the objective function defined in Subsections 2.2 and 2.4.

Consider a sentence s_n with L words, $s_n = [w_{n_1}, \dots, w_{n_L}]$, the log likelihood of s_n based on the mapping or word representation $M = \{(w_1, \dots, w_V) : (v_1, \dots, v_V)\}$ is

$$\log p_M(s_n) = \sum_{t=1}^{L} \sum_{j=t-b, b\neq 0}^{t+b} \log p_M(w_{n_j}|w_{n_t}) \quad , \tag{4}$$

where w_{n_t} is the n_t-th word and w_{n_j} is its context word, and $p_M(w_j|w_t)$ can be obtained with the mapping M,

$$p_M(w_j|w_t) = \frac{e^{v'_j \cdot v_t}}{\sum_{i=1}^{V} e^{v'_i \cdot v_t}} \quad , \tag{5}$$

where v_t is the representation of w_t and so on, and the summation in the denominator is over all words in the vocabulary.

So the document D, $D = \{s_1, \ldots, s_N\}$, has log likelihood

$$\log p_M(D) = \sum_{i=1}^{N} \log p_M(s_i) \quad . \tag{6}$$

The posterior probability $p(u|D)$ that D is produced by user u can be derived from Bayes rule as follows:

$$p(u|D) = \frac{p_{M_u}(D)\pi_u}{\sum_{u' \in U} p_{M_{u'}}(D)\pi_{u'}} \tag{7}$$

where π_u is the prior probability of user u, M_u is the personalized word vectors for user u, and the summation in the denominator is over all users considered.

Finally, the predicted user is $\hat{u}$:

$$\hat{u} = \arg\max_u p(u|D) \quad . \tag{8}$$

3.1.2. Evaluation measure

Two measures are used here:

1. Prediction accuracy: Percentage of documents for which the corresponding user is correctly predicted.
2. Mean reciprocal rank (MRR): If the correct user is predicted as the r-th candidate, the reciprocal rank is $\frac{1}{r}$. The mean reciprocal rank is the average of the reciprocal ranks so MRR should be less than 1.0, and the closer to 1.0 the better.

3.2. Sentence Completion

In this task, from each test sentence we scoop the word with maximum `TF-IDF`, and then use the semantics from the word vectors to find the best word to fill up the blank. This can be achieved by taking the average of embeddings of the remaining words in the sentence, then ranking all words based on the cosine similarity with respect to this average. The scooped word is taken as the correct answer and the mean reciprocal rank (MRR) is used in the evaluation. Higher MRR implies the word embedding is better.

4. EXPERIMENTAL SETUP

4.1. Corpus

First of all, 2.6M sentences including 42,558 distinct words in lexicon were collected from Plurk, a popular social networking site. These data from Plurk were used as the universal background corpora for training the universal background word vectors. The testing experiments were conducted on a set of personalized corpus crawled from Facebook. In order to obtain the personalize Facebook posts, we implemented a cloud-based application capable of helping users to access their social network via voice. Each user can log in his Facebook account and grant our application the authority to collect his linguistic data for experiment purposes. A total of 40 users did so. As a result, all data accessible to the accounts of these 40 target users were collected. This resulted in a total of 67,656 sentences. The number of sentences for each user ranged from 308 to 5,140, with 10.6 words (Chinese or English or mixed) per sentence in average. For each target user, 3/5 of his corpus is taken as the training set, 1/5 as the validation set, and the rest 1/5 for testing.

The code-mixing phenomenon appears in the sentences collected from both Plurk and Facebook. Most sentences were produced in Chinese, but some words or phrases were naturally produced in English and embedded in the Chinese sentences. The mix ratio for the Chinese characters to English words in the Facebook data is roughly 10.5:1.

4.2. User Prediction & Sentence Completion

In the user prediction task, we divide each target user's testing set into smaller documents, each containing at most 30 sentences, and the total number of documents is 473. Each testing document is labeled with the user who produced the document. The user prior probabilities π_u in Eq.(7) is taken as uniform for all users.

In the sentence completion task, we preprocessed all users' testing set by means of scooping words as mentioned in Subsection 3.2. In total, there are 13,512 sentences for testing.

5. EXPERIMENTAL RESULTS

5.1. User Prediction

This is for the tests mentioned in Subsection 3.1. Table 1 shows the MRR and prediction accuracy averaged over the testing set for the two approaches discussed in Subsections 2.3, 2.4 compared to a baseline. The first section (A) (No Background) is for the results when all personalized word vectors were trained directly with the personalized corpus only, without using the background corpora. The second section (B) (Retrain) and third section (C) (Adaptive Layer) are respectively for the two proposed approaches summarized in Subsections 2.3 and 2.4, all with word vector dimensionality of 128, 192 and 256.

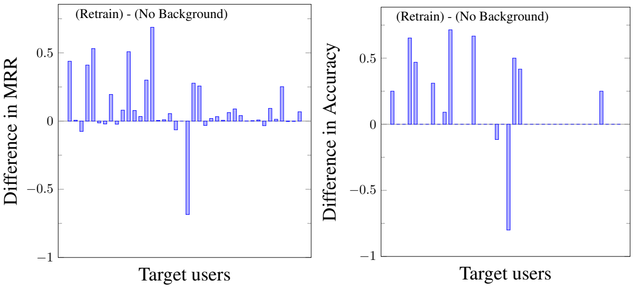
We see from section (A) the word vectors trained with personalized corpus only without background corpora got the worst MRR and accuracy in the table, obviously because the personalized corpus is too sparse to offer reasonably good word vectors. With the help of the background corpora and the universal background word vectors, we see the MRR and accuracy were significantly better and increased as the embedding size went bigger in sections (B)(C). Comparing the two proposed methods, we see Adaptive Layer (section (C)) was clearly better than Retrain (section (B)) with the same

Approaches	Embedding Size	MRR	Prediction Accuracy
(A) No Background	128	0.256	0.140
	192	0.296	0.204
	256	0.336	0.226
(B) Retrain	128	0.402	0.309
	192	0.424	0.340
	256	0.430	0.340
(C) Adaptive Layer	128	0.580	0.485
	192	0.610	0.523
	256	0.630	0.512

Table 1. Evaluation results for the user prediction task using different approaches, all with the personalized data.

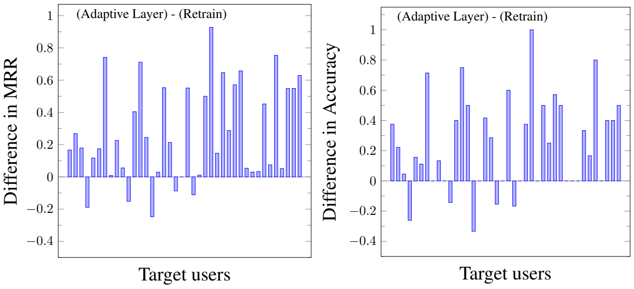
embedding size. There can be at least two reasons for this. First, there are much more parameters to be trained for the Retrain approach, i.e., there are $V \times h \times 2$ parameters to be trained for the matrices $\mathcal{W}_{h\times V}$ and $\mathcal{W}'_{V\times h}$, where V is vocabulary size and h is the embedding size. In contrast, in Adaptive Layer approach only $h \times h$ parameters for the matrix $\mathcal{A}_{h\times h}$ are to be trained. The former is much larger because the vocabulary size V is usually at the order of ten thousands and the embedding size is at the order of hundreds. So much more personalized data are needed for the Retrain approach to learn high quality personalized word vectors. Second, in Retrain approach, the vectors for those words appearing in the personalized corpus were fine-tuned to fit the personalized corpus. However, for those words not appearing in the personalized corpus, the corresponding word vectors were almost never trained and simply remained primarily unchanged from those learned from the universal background corpora. So the words were in fact divided into two separate groups, the unseen words trained with the background corpora and the observed words trained with the personalized corpus. In contrast, the Adaptive Layer approach used an additional linear transformation layer to adapt the whole set of word vectors according to the personalized corpus. In other words, the linear adaptive layer learned a full transformation matrix $\mathcal{A}_{h\times h}$, although small, which mapped the whole set of background word vectors to a new space of personalized semantics. This linear transformation also prevented the word vectors from overfitting to the personalized corpus.

Since the averages didn't actually tell how the different approaches compared with each other for each individual user, we plot in addition the differences in MRR and prediction accuracy across all the 40 target users in Figs. 3 and 4. Each bar in the figures represents the score obtained with one approach minus that with another, all with embedding size of 256. Fig. 3 is for the Retrain approach minus No Background, while Fig. 4 is for the Adaptive Layer approach minus the Retrain approach. From Figs. 3 and 4, we see the differences are quite apparent for most target users.



(a) MRR Differences (b) Prediction Accuracy Differences

Fig. 3. For the user prediction task: difference in (a) MRR and (b) Prediction Accuracy for each individual user for those obtained with Retrain approach minus No Background approach, all at embedding size of 256.

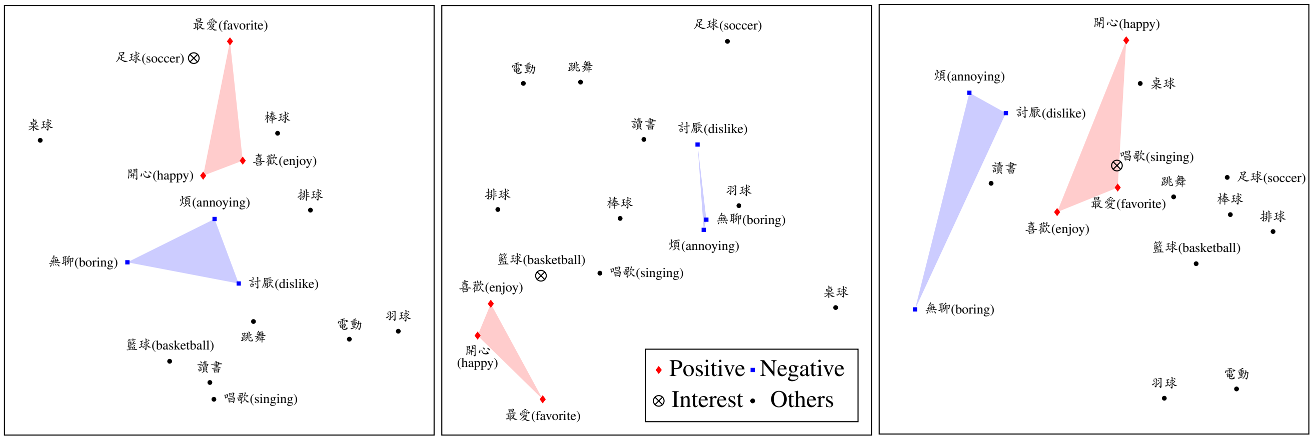


(a) MRR Differences (b) Prediction Accuracy Differences

Fig. 4. For the user prediction task: difference in (a) MRR and (b) Prediction Accuracy for each individual user for those obtained with Adaptive Layer approach minus Retrain approach, all at embedding size of 256.

5.2. Sentence Completion

Table 2 reports the MRR for four different approaches. The sections (A)(C)(D) are for exactly the same cases as those in Table 1, respectively for using personalized corpus alone, and the personalized word vectors by the proposed two approaches. The extra section (B) (Background) is for the word vectors trained with the background corpora only. Column (1) (Percentage within Top 500) lists the percentages of the test sentences for which the correct words for the blanks were ranked within the top 500 words found by the word vectors, and column (2) (MRR within (1)) reports the MRR values averaged over those sentences with the correct words ranked within 500. It can be found that the two proposed approaches (sections (C)(D)) performed significantly better with trends consistent with those observed in Table 1. Also, because many of the test sentences are very short with only a few words, so the sentence completion task is actually a very difficult tasks here. As a result, only 4.49% - 5.18% of them had correct words within 500, and the MRR obtained was not high.



(a) User A with interest in soccer

(b) User B with interest in basketball

(c) User C with interest in singing

Fig. 5. Personalized word vectors visualization of three different users.

Approaches	Embedding Size	(1) Percentage within Top 500 (%)	(2) MRR within (1)
(A) No Background	128	4.49	0.178
	192	4.50	0.168
	256	4.57	0.186
(B) Background	128	4.70	0.182
	192	4.73	0.194
	256	4.72	0.188
(C) Retrain	128	4.76	0.186
	192	4.77	0.196
	256	4.85	0.201
(D) Adaptive Layer	128	4.93	0.198
	192	4.98	0.210
	256	5.18	0.224

Table 2. Results for the sentence completion task: (1) percentage of test sentences with correct answer within top 500 and (2) MRR averaged for those sentences in (1).

5.3. An Example

We tried to visualize the personalized word vectors for three example users trained with the second approach of adaptive layer with dimensionality of 256, and plot small subsets of them with t-sne in Fig. 5 (a)(b)(c) respectively. The black points marked by "•" are Chinese words representing human activities such as singing(唱歌), dancing(跳舞), studying(讀書), basketball(籃球) and soccer(足球). The red triangle is a positive emotion triangle defined by three red points marked by "♢" for words indicating positive emotion: happy(開心), favorite(最愛), enjoy(喜歡), while the blue triangle is a negative emotion triangle defined by three blue points marked by "□" for words indicating negative emotion: dislike(討厭), boring(無聊), annoying(煩). Fig. 5 (a)(b)(c) are the word vectors for three different users A, B, C with different personal interests respectively in soccer(足球), basketball(籃球) and singing(唱歌), word vectors for which are respectively marked by "⊗" in the subfigures (a)(b)(c). In Fig. 5 (a), user A is interested in soccer(足球). We can see his word vector for soccer(足球) is close to the positive triangle but far from the negative triangle. However, in Fig. 5 (b)(c) for users B and C with different interests, the word soccer(足球) is more or less neutral in emotion. Similarly user B in Fig. 5 (b) is interested in basketball(籃球), so the word basketball(籃球) in Fig. 5 (b) is close to the positive emotion triangle but far from negative triangle, but is more or less neutral in Fig. 5 (a)(c). Same in Fig. 5 (c) for user C who is interested in singing(唱歌). These results demonstrate the approach proposed here is able to actually extract some personalized semantics as discussed earlier in this paper.

6. CONCLUSIONS

In this paper, we proposed a new framework for training personalized word vectors carrying personalized semantics using personalized data crawled from social networks. The word vectors are first trained with universal background corpora to learn the general knowledge, and then adapted towards the personalized data by fine-tuning the background word vectors. Two approaches were proposed for the adaptation, one by retraining the word vectors while the other by inserting an adaptation layer. Experimental results over a user prediction task and a sentence completion task showed that both approaches offered consistently better results, and the adaptive layer approach is better than the retrain approach.

7. REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin, "A neural probabilistic language

model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[2] Ronan Collobert and Jason Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.

[3] Joseph Turian, Lev Ratinov, and Yoshua Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[4] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng, "Improving word representations via global context and multiple word prototypes," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012, pp. 873–882.

[5] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, 2010, vol. 2, p. 3.

[6] Omer Levy and Yoav Goldberg, "Dependency-based word embeddings.," in *ACL (2)*, 2014, pp. 302–308.

[7] Jeffrey Pennington, Richard Socher, and Christopher D Manning, "Glove: Global vectors for word representation.," in *EMNLP*, 2014, vol. 14, pp. 1532–1543.

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[10] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig, "Linguistic regularities in continuous space word representations.," in *hlt-Naacl*, 2013, vol. 13, pp. 746–751.

[11] Marco Baroni, Georgiana Dinu, and Germán Kruszewski, "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors.," in *ACL (1)*, 2014, pp. 238–247.

[12] Omer Levy and Yoav Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in neural information processing systems*, 2014, pp. 2177–2185.

[13] Mirco Speretta and Susan Gauch, "Personalized search based on user search histories," in *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*. IEEE, 2005, pp. 622–628.

[14] Zhangjie Fu, Kui Ren, Jiangang Shu, Xingming Sun, and Fengxiao Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE transactions on parallel and distributed systems*, vol. 27, no. 9, pp. 2546–2559, 2016.

[15] Xuehua Shen, Bin Tan, and ChengXiang Zhai, "Implicit user modeling for personalized search," in *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 2005, pp. 824–831.

[16] Gui-Rong Xue, Jie Han, Yong Yu, and Qiang Yang, "User language model for collaborative personalized search," *ACM Transactions on Information Systems (TOIS)*, vol. 27, no. 2, pp. 11, 2009.

[17] Paul-Alexandru Chirita, Claudiu S Firan, and Wolfgang Nejdl, "Personalized query expansion for the web," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 7–14.

[18] Pei-Hao Su, Chuan-Hsun Wu, and Lin-Shan Lee, "A recursive dialogue game for personalized computer-aided pronunciation training," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 1, pp. 127–141, 2015.

[19] Chih-Ming Chen and Yi-Lun Li, "Personalised context-aware ubiquitous learning system for supporting effective english vocabulary learning," *Interactive Learning Environments*, vol. 18, no. 4, pp. 341–364, 2010.

[20] Yehuda Koren, Robert Bell, and Chris Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.

[21] Frank Edward Walter, Stefano Battiston, and Frank Schweitzer, "A model of a trust-based recommendation system on a social network," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 57–74, 2008.

[22] Xiwang Yang, Yang Guo, Yong Liu, and Harald Steck, "A survey of collaborative filtering based social recommender systems," *Computer Communications*, vol. 41, pp. 1–10, 2014.

[23] Shuiguang Deng, Longtao Huang, and Guandong Xu, "Social network-based service recommendation with trust enhancement," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8075–8084, 2014.

[24] Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho, "Location-based recommendation system using bayesian user's preference model in mobile devices," in *International Conference on Ubiquitous Intelligence and Computing*. Springer, 2007, pp. 1130–1139.

[25] Yoon Ho Cho, Jae Kyeong Kim, and Soung Hie Kim, "A personalized recommender system based on web usage mining and decision tree induction," *Expert systems with Applications*, vol. 23, no. 3, pp. 329–342, 2002.

[26] Yu-Yang Huang, Rui Yan, Tsung-Ting Kuo, and Shou-De Lin, "Enriching cold start personalized language model using social network information.," in *ACL (2)*, 2014, pp. 611–617.

[27] Arjumand Younus, Colm O'Riordan, and Gabriella Pasi, "A language modeling approach to personalized search based on users' microblog behavior.," in *ECIR*. Springer, 2014, pp. 727–732.

[28] Tsung-Hsien Wen, Hung-Yi Lee, Tai-Yuan Chen, and Lin-Shan Lee, "Personalized language modeling by crowd sourcing with social network data for voice access of cloud applications," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 188–193.

[29] Tsung-Hsien Wen, Aaron Heidel, Hung-yi Lee, Yu Tsao, and Lin-Shan Lee, "Recurrent neural network based language model personalization by social network crowdsourcing.," in *INTERSPEECH*, 2013, pp. 2703–2707.

[30] Bo-Hsiang Tseng, Hung-yi Lee, and Lin-Shan Lee, "Personalizing universal recurrent neural network language model with user characteristic features by social network crowdsourcing," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 84–91.

[31] Hung-Yi Lee, Bo-Hsiang Tseng, Tsung-Hsien Wen, and Yu Tsao, "Personalizing recurrent-neural-network-based language model by social network," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 3, pp. 519–530, 2017.

[32] Hans van Halteren, "Linguistic profiling for authorship recognition and verification," in *ACL*, 2004.

[33] Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Patrick Juola, Aurelio López-López, Martin Potthast, and Benno Stein, "Overview of the author identification task at pan 2015," in *CLEF 2015 Evaluation Labs and Workshop – Working Notes Papers*, Toulouse, France, 09/2015 2015, CEUR, CEUR.

[34] Xinxiong Chen, Zhiyuan Liu, and Maosong Sun, "A unified model for word sense representation and disambiguation.," in *EMNLP*, 2014, pp. 1025–1035.

[35] Thang Luong, Richard Socher, and Christopher D Manning, "Better word representations with recursive neural networks for morphology.," in *CoNLL*, 2013, pp. 104–113.

[36] Cicero D Santos and Bianca Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1818–1826.

[37] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.

[38] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[39] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin, "Learning sentiment-specific word embedding for twitter sentiment classification.," in *ACL (1)*, 2014, pp. 1555–1565.

[40] Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso, "Finding function in form: Compositional character models for open vocabulary word representation," *arXiv preprint arXiv:1508.02096*, 2015.

[41] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[42] Matt Taddy, "Document classification by inversion of distributed language representations," *CoRR*, vol. abs/1504.07295, 2015.