Joint User-Entity Representation Learning for Event Recommendation in Social Network

Lijun Tang
Facebook
1 Hacker Way
Menlo Park, CA 94025
Email: leotang@fb.com

Eric Yi Liu
Facebook
1101 Dexter Ave N
Seattle, WA 98101
Email: liuyi@fb.com

***Abstract*—User-managed-events is a popular feature on social networks. Take Facebook Events as an example: over 135 million events were created in 2015 and over 550 million people use events each month. In this work, we consider the heavy sparseness in both user and event feedback history caused by short lifespans (transiency) of events and user participation patterns in a production event system. We propose to solve the resulting cold-start problems by introducing a joint representation model to project users and events into the same latent space. Our model based on parallel Convolutional Neural Networks captures semantic meaning in event text and also utilizes heterogeneous user knowledge available in the social network. By feeding the model output as user and event representation into a combiner prediction model, we show that our representation model improves the prediction accuracy over existing techniques (+6% AUC lift). Our method provides a generic way to match heterogeneous information from different domains and applies to a wide range of applications in social networks.**

***Index Terms*—Representation Learning, User modeling, Neural Network, Recommendation**

1. Introduction

User-managed-events (*events* in the remaining of the text) is a social network feature for organizing both online and offline activities. Any user can host a public or private event ranging from birthday parties to ticketed concerts by specifying a basic set of information such as Date and Time, Location, Nature of the Event, Attendees (See Figure 1 for an example of an event on a social network). *Events* is also a channel for organizers and promoters to reach desired audiences. Each month, over 550 million users use *Facebook Events*; more than 135 million events were created on Facebook in 2015; 35 million users view a public event each day[1]. In this paper, we use participation to denote the action that a user agrees to attend an event (e.g., clicks the join button). We consider the event recommendation problem by predicting the participation decision for a user on a set of events.

Seattle Ice Cream Festival
1 friend is going

Category	Public / Tasting
Hosted by	Kurt Farm Shop
Date & Time	Sunday, May 22 at 12 PM
Location	Chophouse Row 1424 11th Ave, Seattle, Washington 98122
Details	First annual Seattle Ice Cream Festival located at Chophouse Row on Capitol HIll. A dozen of Seattle's best ice cream makers sampling and selling their unique ice creams. No admission fee; open to all.

Figure 1. An example of event on a social network

Despite the long existence of the events feature in social networks (e.g., *Facebook Events* dates back to 2005[2]), little research has been done to address the specific needs of the event recommendation problem. The dynamic nature of events makes its recommendation problem highly challenging. Most importantly, events generally have very short lifespans (from creation to the actual event time). Once an event expires, it is no longer eligible for any further consideration for the system to recommend. Thus event recommendation requires making timely recommendations with little user feedback on the current active event set.

Besides standard attribute matching such as location, date and time availability, and friends' event participation, the first issue in solving cold-start is to extract and match event semantic knowledge accurately. Existing recommendation methods usually fall back to relatively simple retrieval models or semantic models such as keyword/tag matching, probabilistic latent semantic analysis (PLSA) [1] and latent Dirichlet allocation (LDA) [2]. These approaches, though efficient, have limited expressive power and accuracy over semantic knowledge. For example, most text models

1. Facebook Events Playbook: https://s0.wp.com/wp-content/themes/vip/fb-events-2015/pdfs/Event_Playbook.pdf

2. http://www.wired.com/2015/11/inside-facebook-events-updates/

adopted bag-of-words representation and assume topic independence. A more significant limitation with these text models is that they do not allow different featurization for users and items. As a result, in order to project user and item into the same topic distribution space, a user has to be represented by (an aggregate of) the same type of items explicitly or implicitly. For examples, in representing a user, [3] uses aggregated events for event recommendation; [4] uses aggregated point-of-interests for point-of-interest recommendation; [5] learns user vectors only from scientific literature for literature recommendation. The implicit homogeneity restriction on user representation becomes an information bottleneck in using heterogeneous user data available in social networks.

The ability to make use of heterogeneous user data is especially important because the user-level history in event space alone is exceedingly sparse. Although the total number of events and participation counts are substantial, the individual level participation history is often unavailable or very limited. The relatively low per-user participation rate in events can be attributed partly to the much higher requirement both physically and mentally in attending an event than that in clicking on an ad or a movie link. It is also because most events are confined in small communities and very few events could attract a broad audience like a popular movie or music does in other recommendation problems.

Given such sparse user history and short event lifespans, our solution lies in leveraging rich user attributes and matching them to event semantic knowledge accurately. Social networks often possess a richer set of user attributes than other online services. Here, user attributes could include demographic information, self or auto-identified keywords & topics, and even action logs of other types of activities (previous posts/comments, likes, subscriptions). Note that, however, these attributes are often non-text numerical and categorical values and can not share the same feature representation with event knowledge in the text form (events can be seen as short text documents). Also, user attributes that could be expanded into text forms (e.g., previous posts) have very different text distribution than that of events. It thus forbids us from using standard topic models or information retrieval methods to match user attributes to event knowledge.

In this work, we describe a practical approach to improve event recommendation at very large scale. We propose a two-stage processing system by decoupling event semantic model and user profile model from final prediction model. In the first stage, we perform joint representation learning to construct event model and user model simultaneously. Our user model allows a very flexible input set of user attributes and both models requires minimal feature pre-processing or engineering. Our method is based on deep Convoluted Neural Networks that take full context into consideration in comparison to bag-of-words-based approaches such as PLSA and LDA. The user and event models together project any given user and event into the same latent space and enables efficient and accurate matching. In the second stage, we feed the matching result as a feature, together with other standard features, into a gradient-boosting-decision-trees (GBDT) based combiner model [6]. Experiment results show that our jointly learnt representation models improve the prediction accuracy. Our work demonstrates an effective way to integrate entity representation learning into very large scale recommendation system. Our joint learning method can also be extended to many other types of entity matching problems where the feature domains do not match.

2. Related Work

2.1. Event Recommendation

De Pessemier et al. [7] propose a general framework CUPID for event recommendation. To address sparsity problem, the authors suggest to start with user-based collaborative filtering to self-enhance user-profile and then perform content-based filtering. Relying heavily on user feedback, it does not address the sparsity problem adequately and it provides no guarantee on the correctness of self-enhancement. In addition, the proposed system uses a keyword/entity indexing approach that does not capture well the underlying semantic knowledge. Kayaalp et al. [8] build a concert event recommendation system using similarity computed from shared artists and user ratings. It also includes social information by showing recommendation to friends. The study does not address the sparsity or semantic extraction issue. Zhang et al. [9] introduce a more comprehensive recommendation framework that combines three aspects of information: 1) topic-distribution similarity between user and event using LDA 2) friends' attendance information 3) user-specific topic preferences. The study relies on LDA to represent both user and event in topic space. To represent a user, the authors uses words from user interest in addition to past activity text. The apparent difference in word distributions (and hence learnt topic distributions) between user doc and event doc is not addressed. The model quality also relies heavily on the availability of user history.

There are also studies focusing on the effect of a particular type of information used in event recommendation. Daly and Geyer [10] focus on evaluating the use of location information in event recommendation within organizations. They also try to identify global events and socially independent events for cold-start recommendations. Also, observing the short lifespans of events, Chen and Sun [3] investigate the use of friends data on top of PLSA-based event matching. Similarly, Bogaert et al. [11] work on isolating the impact of using friends data. For events available in a specially structured form (LODE ontology [12]), research has been done to exploit the structure to improve semantic extraction [9], [13].

The baseline event recommendation model used in this study is a production model on a large social network website that covers many types of the information used in the above-mentioned studies. As discussed previously, existing methods rely on keyword indexing and matching or bag-of-words text models and do not adequately capture

semantic information for cold-start prediction. Also, the text models used in existing methods (PLSA/LDA) can only represent a user by historical events they participated. Our method is the first that provides a generic way to extract event semantic knowledge and map to an arbitrary set of user data. Finally, our training and serving scale is several orders of magnitude larger than existing evaluations and our study is validated on an experimental data set that includes all event types across a large social network website.

2.2. Other Recommendation Problems

Recommendation algorithms have contributed to the success of web applications in different domains (see [14], [15] for recent surveys). Most recommendation systems adopt collaborative filtering and/or content-based filtering which rely heavily on implicit or explicit feedback observations. Most recently Wang et al. [16] used a deep neural network to extract content features from items and incorporated them into a collaborative filtering framework to improve the recommendation results. Although cold-start needs to be addressed for any real production system, most recommendation problems can assume relatively stable item set and/or user set. In problems such as movie recommendation, point-of-interest (POI) recommendation, and scientific literature recommendation, old items almost never get removed. On the contrary, in event recommendation problem, new events get created and old events phase out at an unparalleled pace. Combined with user side sparsity, it makes accurate semantic extraction and matching much more important in event recommendation.

A widely adopted approach is to enhance filtering algorithms with semantic models such as PLSA and LDA. Besides being inadequate in capturing item semantics, these models suffer from the user-representation problem as discussed in the previous section. For example, studies in [4], [5], [17], [18] learn or adjust user representation only based on the content of the particular type of items under investigation. For general recommendation, rich user information is not always available. But in the context of a social network, we may leverage this information with item content to address the user history sparsity in event recommendation.

2.3. Representation Learning Algorithms

Our goal of matching entities (user and event) from different feature domains is similar to intent matching problems in search engines where search query and result come from different domains. Our loss function formulation and text feature extraction is similar to that in [19], [20], [21]. Our representation learning design also uses techniques similar to those in [22], [23].

3. Joint User and Event Representation Learning

In this section, we present the general network design of our joint user and event representation model and its training process. Our model takes in deidentified and privacy-preserving user data in the form of an arbitrary set of categorical and text features. Examples of user data for this experimental set include demographic and geographic information, self-labeled or auto-generated keywords & topics, selected types of activities (e.g., subscribed pages and celebrities). By assigning each feature-value pair a distinct id, we treat all categorical features as id features and do not consider categorical feature values as natural language texts. Some user data such as subscribed pages are included as both categorical features (using page id) and text features (using page title). Numerical user features can be easily converted to categorical features by bucketing or K-means. Our event data includes only raw text features such as title, description, and category. For both user and event, we combine text features into a single text document. An event is then represented simply by a text document, and our study is validated on an experimental data set that includes all event types. A user is represented by a text document and an unordered list of id features (corresponds to categorical feature-value pairs).

Throughout the section, we denote matrices by uppercase letters and vectors by lowercase letters in bold. $\mathbf{x}_{(i)}$ denotes the i-th entry in vector $\mathbf{x}$ and $M_{(i,j)}$ denotes the (i,j)-th entry of the matrix M. We consider a neural network parameterized by θ, which includes all projection matrices between network layers and lookup table values.

3.1. Convolutional Feature Extraction

A core component of our representation network is the convolutional feature extraction module. We consider a flexible class of feature extraction modules that can be used for both user and event data. Each instance of extraction module is characterized by input data, tokenization method, and convolution setting. We apply multiple extraction modules on user data to cover different aspects of information. The extracted feature vectors (output of extraction modules) are then concatenated to form the complete user feature vector. Similarly, multiple extractions are done for events and the output vectors are concatenated to form the event feature vector.

Figure 2 shows the general architecture of our convolutional feature extraction module. The input data (at the bottom) consists of a sequence of "words". Here the notion of word sequence can represent general ordered text (with punctuations replaced or removed) and also an unordered list of id features (each id is considered a word). The sequence of words is then processed through a tokenizer that generates a list of "tokens". In this work, we mainly consider two types of tokenizers, letter trigram tokenizer for text document and word unigram tokenizer for id features. The next layer maps each token into a feature vector by a lookup table operation ($t_i \rightarrow \mathbf{v}_{t_i}$). The lookup table is part of the network parameter θ to be trained and can be randomly initiated (Section 3.2.1 discusses further on unsupervised initialization). To limit the table size, we apply document frequency based filtering to remove rare tokens.

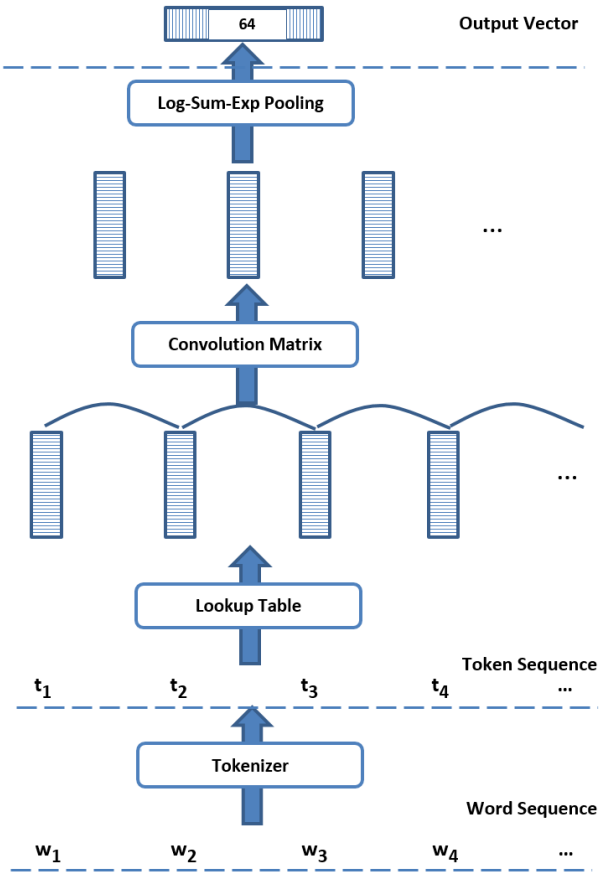

Figure 2. A Convolutional Feature Extraction Module

Given the list of token feature vectors $\{\mathbf{v}_{t_1}, \mathbf{v}_{t_2}, \mathbf{v}_{t_3}, \ldots\}$ and a specified convolution window size d, we then apply a convolution layer as follows. For each window starting at token i, a window vector $\mathbf{v}_{w_i}$ is generated by concatenating the token vectors $v_i^t \ldots v_{i+d-1}^t$ within the window. We then obtain a new list of window vectors by applying a convolution matrix M_c:

$$\{\mathbf{v}'_{w_1}, \mathbf{v}'_{w_2}, \mathbf{v}'_{w_3}, \ldots\} = M_c \times \{\mathbf{v}_{w_1}, \mathbf{v}_{w_2}, \mathbf{v}_{w_3}, \ldots\}$$

The final output vector $\mathbf{v}$ is obtained by performing log-sum-exp pooling over all dimensions of $\{\mathbf{v}'_{w_1}, \mathbf{v}'_{w_2}, \mathbf{v}'_{w_3}, \ldots\}$. To be more specific, the k-th element of $\mathbf{v}$ is given by:

$$\mathbf{v}_{(k)} = \mathbf{v}'^{*}_{(k)} + log \sum_i e^{\mathbf{v}'_{w_i(k)} - v'^{*}_{(k)}}$$

where $\mathbf{v}'^{*}_{(k)} = max(\mathbf{v}'_{w_1(k)}, \mathbf{v}'_{w_2(k)}, \mathbf{v}'_{w_3(k)}, \ldots)$.

The dimensionality of the token vectors and final output vector can be chosen to balance the complexity of the model and training effort. In this work, we use 64 for both. The convolution matrix M_c is thus of size $64 \times (d \times 64)$. Different convolution window sizes (d) are used together as explained below.

3.1.1. Feature Extraction for Users. As described previously, a user is represented by a list of categorical features and a text document. We combine four convolutional feature extraction modules, one for categorical features and three for text document. The categorical feature extraction module uses unigram word tokenizer and has convolution window size set to 1, which essentially preserves feature values in their original form and treat all of them independently. As explained earlier in this section, categorical features include a rich set of user demographic and geographic attributes in addition to other profile attributes and selected activity observations. The three text feature extraction modules use letter trigram tokenizer that has been proven effective in reducing token space size and increasing coverage on uncommon words [20]. Convolution window size 1,3,5 are used for the text extraction modules to cover semantic segments of different lengths. The final concatenated user feature vector is of size $64 \times 4 = 256$, as shown in Figure 3.

3.1.2. Feature Extraction for Events. Similar to user feature extraction, we combine multiple convolutional feature extraction modules for event text data. The text document for an event is simply the concatenation of event meta texts (title, description and category). Since we only use text information for events, we apply three modules with letter trigram tokenizer and convolution window size 1,3,5. The final concatenated event feature vector is of size $64 \times 3 = 192$.

3.2. Joint User and Event Model

With the convolutional extraction modules defined and user and event feature vectors extracted, we construct a joint user and event model as shown in Figure 4. The model consists of two parallel sub-models for user and event (left & right halves of Figure 4) that are only connected in the last step. The user and event sub-models are of identical design except using different sets of extraction modules as explained in Sections 3.1.1 and 3.1.2.

Inside a sub-model, with the user (event) feature vectors extracted, we connect the concatenated feature vector to an affine hidden layer followed by a tanh activation function for non-linearity. We then feed the output of the hidden layer into our representation layer by a linear projection. To increase training effectiveness, we also feed the feature vector directly into the representation layer bypassing the hidden layer (similar to the residual net idea in [24]). The representation layer is also applied with a tanh activation function. The output at representation layer, $\mathbf{v}_u$ ($\mathbf{v}_e$), is the output of the user (event) sub-model and represents a latent projection of the user (event) information.

The user and event sub-models are then connected using a cosine function in the last step. It is interpreted as the similarity between a user and an event.

$$s_\theta(u, e) = \frac{\mathbf{v}_u \cdot \mathbf{v}_e}{\|\mathbf{v}_u\| \|\mathbf{v}_e\|}$$

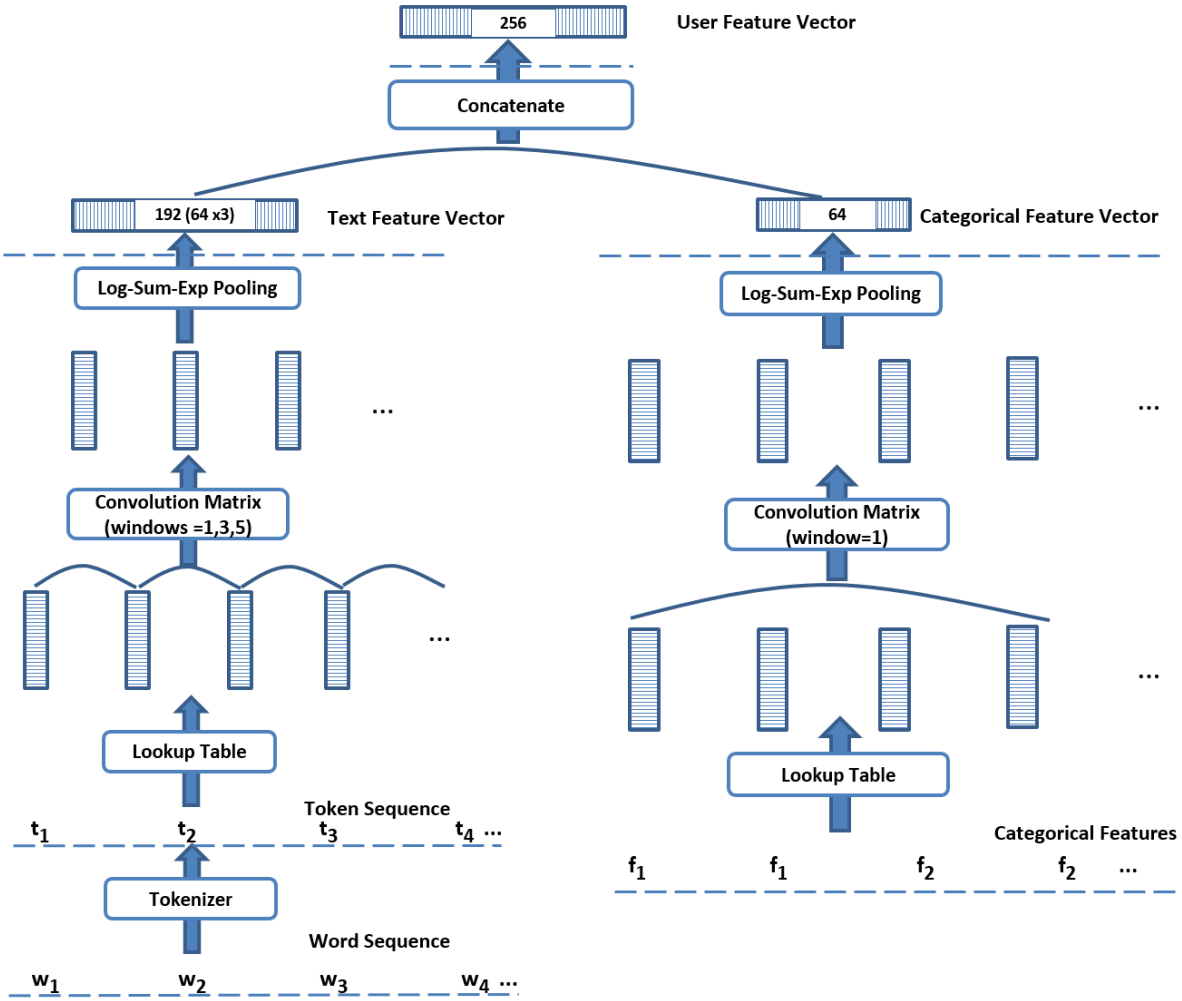

Figure 3. The Feature Extraction Modules For Users. Left: Three convolutional text feature extraction modules for three different convolution windows (condensed to save space), Right: Single convolutional categorical feature extraction module with convolution window size $= 1$

3.2.1. Model Training. Given the similarity score defined, we apply the following loss function to perform training:

$$L_\theta(u,e) = \begin{cases} 1 - s_\theta(u,e) & \text{if } y_{(u,e)} = 1 \\ max(0, s_\theta(u,e) - \theta_r) & \text{if } y_{(u,e)} = 0 \end{cases} \quad (1)$$

where $y_{(u,e)}$ indicates whether user u participated in event e and θ_r is a parameter chosen to indicate the desired dissimilarity level between a negative pair of user and event. If we consider $1 - s_\theta(u,e)$ as the distance between user u and event e, the loss function essentially encourages a positive pair of user and event to have small distance and a negative pair to have a distance greater than $(1 - \theta_r)$. We found that the training is not very sensitive to the choice of θ_r and we use zero for all experiments presented in this work. Here the loss function is intended for point-wise prediction. Another possible way of defining the loss function is to consider a user's relative preference over a set of events (ranking loss). Though more flexible, it adds training complexity and fits more with ranking data.

Our neural network is trained using Torch [25]. We set the length of all lookup table values (vectors) to be 64. The output vectors of convolutional extraction modules are also set to be of length 64. In the joint model, the hidden layer is set to have 256 nodes and the final representation layer has 128 nodes on each side. The weights and lookup table values are all randomly initialized and updated by back-propagation. We apply a simple document frequency (DF) filter so that our total lookup table size is kept below 500k (specifically, 236k for user text features, 78k for user categorical features, and 99k for event text features). Note that some lookup table values can be partially initialized from other sources such as a general embedding trained on text corpus [22], [26]. We discuss a simple initialization approach using the same network architecture later. The user categorical features, however, cannot be covered by such text embeddings. To avoid overfitting, we apply early-stopping with decaying learning rate. We adjust learning rate to 90% of its value after each epoch and our model converges well in

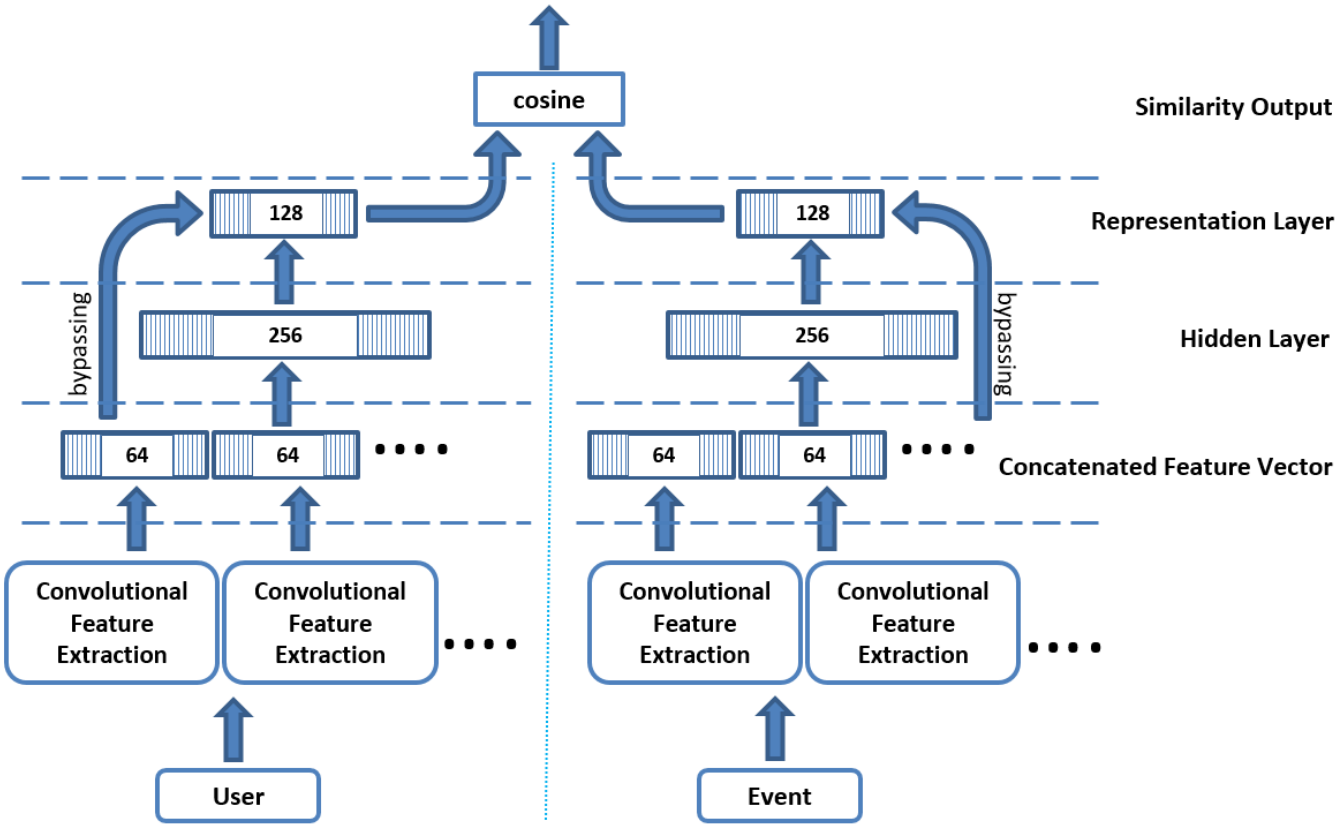

Figure 4. Joint User and Event Neural Network

under 20 epochs (with a randomly initialized lookup table). We attribute the small number of epochs needed partly to the large size of our training data.

With limited user-event observations (in size and date range), it could be difficult to train all lookup table values fully. We find a simple method that helps initialize event lookup table without any user feedback. We take the event sub-net (right half of Figure 4) and construct a Siamese Network [27]. We then sample a large number of events and feed the title and body text into the network as positive training instances. We also randomly pair title and body text from different events and use these as negative training instances. This initialization model alone is already an excellent event-only semantic model. It improves the semantic-search in events ("related events" in which user information is not considered) noticeably over using n-gram based text model (data not shown).

4. Combiner Prediction Model

A combiner prediction model is used to aggregate all features generated and make the final prediction for a recommendation. The combiner model is trained with gradient boosting decision trees (GBDT) [28], which is very effective in finding high-order feature interactions. In training the GBDT model, we minimize the cross-entropy loss over observed user and event pairs.

$$L_\theta = -\sum_{u,e} y_{(u,e)} \log \hat{y}_{(u,e)} + (1 - y_{(u,e)}) \log(1 - \hat{y}_{(u,e)})$$

The combiner feature set covers standard user and event attributes and engineered statistics on matching user attributes with event attributes. It also contains extended social network knowledge about user and event and multiple collaborative filtering features were constructed using the social connections and different types of user reactions. The output of our joint representation model is merged into the combiner model as new features aiming to improve semantic-based matching. Here we consider two sources of information to be incorporated from our joint representation model. We can include the similarity score ($s_\theta(u, e)$) as a numerical feature. We can also include the representation vectors ($\mathbf{v}_u$ and $\mathbf{v}_e$) to allow latent topic interaction in the projected space. Note that with a GBDT combiner no feature normalization or other monotonic transformation is needed to include general numerical features. In Section 5 we show that having latent topic interaction based on representation vectors covers almost all knowledge from the representation model. Being a large neural network, our representation model can be computationally intensive to calculate for each pair of user and event at recommendation

time. The computation, however, can be greatly reduced by pre-computing and caching the user and event representation vectors. User and event vectors are only computed upon creation and important information change. They can be cached in distributed data store such as [29] for quick access at recommendation time.

5. Experiments

5.1. Data Set and Evaluation Method

We train and evaluate our proposed models on a large-scale data set sampled from traffic in a real-world production system. Note that the only user data seen by authors are aggregated and anonymized outputs from model evaluation. Also, only public events are used in this research. Our evaluation is on offline data only and no user's experience was altered in any way as a result of the research. Each data instance in the data set is an impression of an event shown to a user. For one impression, the label is given by whether user participation is achieved from the impression. We sampled from 6-week period to train and evaluate the performance of our proposed method. We down-sampled negative instances and have approximately 1:4 positive to negative ratio. For fair evaluations, we split the data into three parts disjoint in time (4 weeks + 1 week + 1 week). The first part (20 million impressions) is used for representation model training and testing. The second part (6 million impressions) is for combiner model training and the last part (1 million impressions) is for evaluation. This date-based partition is consistent with the actual production system deployment behavior. It also ensures that all model knowledge comes from the data before evaluation period. This is important in producing reliable evaluation when behavior statistics features are used (comparing to using random partition). In the final evaluation, we report the precision and recall (P/R) achieved at different thresholds and also area under the ROC curve (AUC). We focus on high recall region as recommendation diversity is highly important and we do not want to fit to a small set of repeating events.

Our baseline model uses all features described in Section 4 except features from the representation model. The baseline model covers many aspects discussed in existing literature (Section 2.1) and includes multiple collaborative filtering features based on different types of user feedback (e.g., like/dislike, join, interested in) and social connections (e.g., friend, organizer/performer, and events). All experiment combiner models and baseline combiner model are trained using the same setting (200 trees, 12 leaves per tree).

5.2. Main Results

As described in Section 4, we consider different ways of integrating representation model outputs as features into the final combiner model. To be more specific, we consider the following configurations:

- Baseline Only
- Baseline + representation vectors $\mathbf{v}_u$ and $\mathbf{v}_e$
- Baseline + similarity score $s_\theta(u, e)$ & representation vectors

To be more informative, we also include a configuration using the representation vectors only for GBDT training (i.e., not using any baseline features).

- Representation vectors $\mathbf{v}_u$ and $\mathbf{v}_e$ only

Figure 5 shows the P/R curves achieved under different configurations and Table 1 lists the precision and AUC obtained under each configuration. Using the representation vectors alone is able to achieve decent prediction quality (AUC=0.754), comparing to the baseline (AUC=0.810). The precision at recall=0.80 (denoted thereafter by PR80, similarly PR60 for precision at recall=0.60) is 0.215 comparing to baseline PR80=0.262. A major lift happens (+6% over baseline for AUC=0.861, and +29% over baseline for PR80=0.339) when we combine the representation vectors with baseline features that allows interaction among latent topic dimensions and also between latent topics and existing features. We found almost no additional gain (AUC=0.862 and PR80=0.346) by including similarity score on top of representation vectors. This is not surprising as the GBDT trainer captures well high-order feature interactions and per-latent-dimension interactions provide much more information than a summary score alone. In the rest of experiments, we use the representation vectors only as representation features to be included in combiner model. Note that the integration choice can be different for different types of combiner models. For example, for logistic regression, one may need to design additional interaction features and include multiple types of summary scores.

TABLE 1. EFFECT OF DIFFERENT INTEGRATION SETTINGS

Integration Setting	PR60	PR80	AUC
Rep. Vectors	0.289	0.215	0.754
Baseline	0.388	0.262	0.810
Add Rep. Vectors	0.516	0.339	0.861
Add Score and Rep.	0.521	0.346	0.862

We also break down the baseline feature set to provide further insights into the generalizability of our representation model. By splitting the baseline feature set into base feature set and collaborative filtering (CF) feature set, we consider the following combinations of feature sets and the results are presented in Figure 6 and Table 2:

- Base features only (No-CF)
- Base features and Collaborative Filtering features (Full Baseline)
- Base features (No-CF) and Representation Features
- All features available

Both collaborative filtering and our representation model try to generalize knowledge across items using user feedback. As we discussed in previous sections, the effect of collaborative filtering is greatly limited by the short lifespans of events. To overcome sparseness in user and event history,

TABLE 2. COMPARISON ON COMBINATIONS OF FEATURE SETS

Feature Combinations	PR60	PR80	AUC
Base Features (No-CF)	0.364	0.252	0.796
Base and CF Features	0.388	0.262	0.810
Base and Rep. Features	0.516	0.339	0.859
All Features	0.521	0.346	0.862

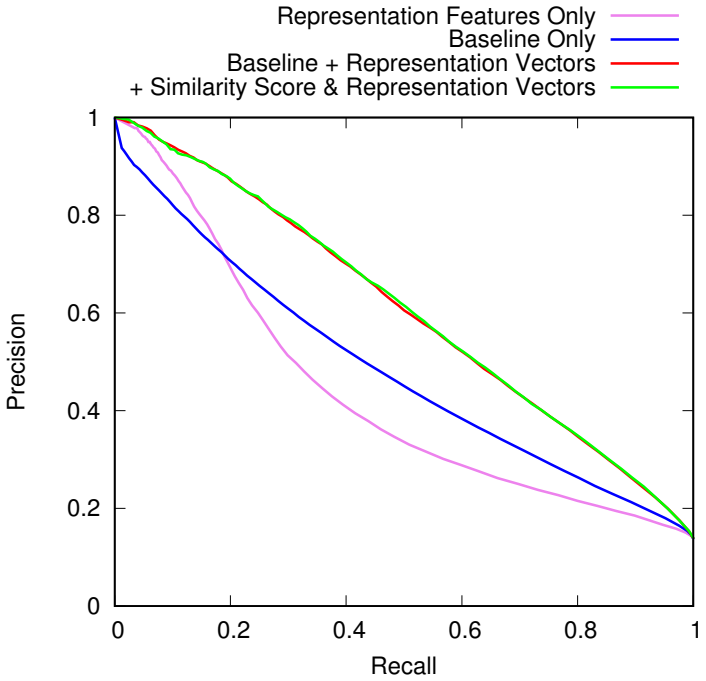

Figure 5. P/R curves for different ways of integrating representation model outputs

multiple types of user feedback and social connections were adopted in constructing collaborative filtering features. Our representation model, although only utilizing user participation as feedback in training, generalizes knowledge by accurate semantic extraction and projection. Adding representation features alone outperforms adding collaborative filtering features by a significant margin. We notice a big overlap between the gains brought by representation features and collaborative filtering features when both are included. This is expected as much of the generalization can be achieved by different information sources. For example, information propagated from friends' activity data can also be seen in work/school information if the friends work/study together.

5.3. Event Representation Analysis

To help understand the representation learnt via soft max-pooling (log-sum-exp), we show some examples of event texts of different lengths in Figure 7. For each text example, we highlight the top 5 words (in bold) that account for most convolution windows achieving max value in the pooling layer. To be more specific, for each of the 64 max values obtained at pooling layer, we trace back to

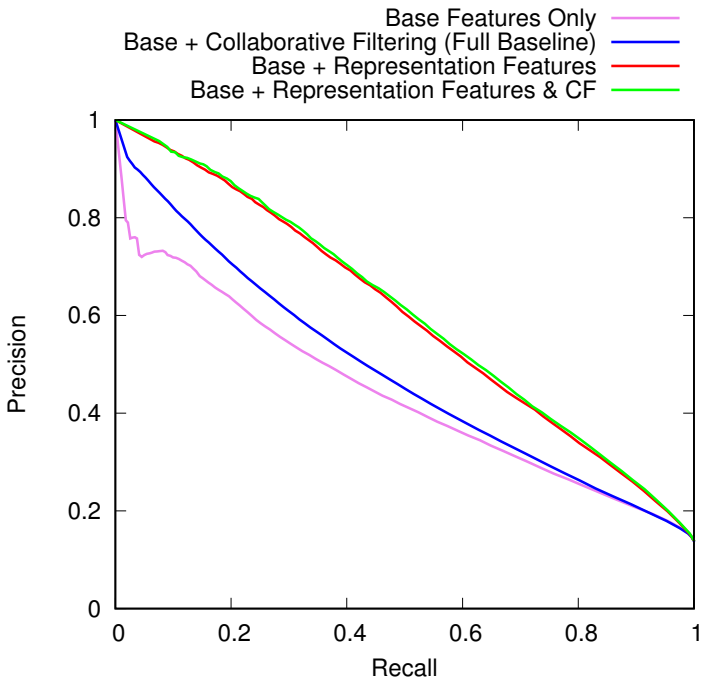

Figure 6. P/R curves for combinations of feature sets

Seed Event	*Paella Feast* In the art of entertaining, little can rival the aromatic, colorful presentation of Palla. Native to Spain, where affection is exchanged as much in spoonfuls as it is in kind words, this dish is a feast ...
Similar Events	*Taste of Romania* - *Greek Dinner at 7pm* The Art House will be inspired by Greek cuisine ... *Chequamegon Chef's Exhibition* Chequamegon Chef's Exhibition on Madeline Island ...

TABLE 3. AN EXAMPLE OF SIMILAR EVENTS DISCOVERED BY A SEED EVENT IN FOOD CATEGORY, USING THE EVENT REPRESENTATION MODEL

the window position and the word(s) overlapping with the window. For a max-value window covering d words, we consider each word contributing $1/d$ to the pooling layer. We go through all 64 max-value windows and sort all words based on their accumulated contribution to the max values. We compute the top words for convolution window sizes of 1,3,5 (i.e., we highlight 3×5 words in total). In Table 7, the numbers in subscript following each highlighted word denote the convolution window size that has the word ranked top. In all three examples, informative nouns and verbs are contributing the most to the pooling layer. In some cases, descriptive adjectives such as "engaging" and "creative" also are ranked high. As the text length increases, top words under different convolution window sizes tend to differ and

Event Text Examples	
Short	Seattle Ice Cream Festival: First Annual Seattle Ice Cream **Festival**$_{1,3,5}$ located at **Chophouse**$_{1,3}$ Row on Capitol Hill. A dozen of **Seattle**$_{1,3,5}$'s best ice cream makers **sampling**$_5$ and selling their **unique**$_{1,3}$ ice **creams**$_5$. No **admission**$_{3,5}$ **fee**$_1$; open to all.
Medium	AutoFest: We will be back for the 4th year looking forward to bring everyone a great event! Please make sure to mark your **calendars**$_3$ and share this with all your friends! **Vendors**$_3$ space is still available please contact us either by email or pm. If you would like us to get back to you just leave a comment and we will. VIP will be open in 2 weeks and **limited**$_5$. We will be **pre-screening**$_{1,3,5}$ so make sure to send in **quality**$_1$ **photos**$_{1,5}$ that **represent**$_{1,3}$ your build the best. If anyone has any **questions**$_5$ please feel free to contact us and we will get back to you. Please check back frequently we will be updating the info as we go along. Thank you and see you at **AutoFest**$_{1,3,5}$!
Long	Easter at hope city: Celebrate with the whole family at hope city! 6 services and 1 massive **easter**$_5$ egg hunt. Egg hunt: Saturday 3:00pm - face **painting**$_{3,5}$, interactive **games**$_5$, bounce houses, super slides and food **trucks**$_3$, **egg**$_{3,5}$ hunt at 4:00pm. Services: Saturday 5pm & 7pm (baptism after 7pm service). **Sunday**$_1$ 8:00am / 9:30am / 11:00am / 12:30pm (baptism after 12:30pm service) Memorial high school 935 **Echo**$_1$ Ln, Houston, TX 77024. Kids ministry is open at all services. Visit our website: http://yourhopecity.Com/. What can I expect? This weekend at hope city will be exciting, casual, and relaxed. You'll be warmly greeted in the parking lot and directed towards the **theatre**$_3$ entrance. Before service, pick up a freshly **brewed**$_3$ **cup**$_1$ of **coffee**$_1$. Then make your way to a seat and prepare for a high-energy and uplifting worship experience as well as a relevant, **engaging**$_1$ message from our pastor, jeremy foster. We love kids! Hope city kids is a children's ministry uniquely designed with your child in mind! This weekend all children (babies through 5th grade) will experience safe, age-appropriate environments where the bible is taught in a **creative**$_5$ and relevant way. Our team will meet your family and assist you through our secure check in process. And finally, if you are a guest, we will not embarrass you, have you stand up, or ask you to give money. We know that many people want to check out the church and "be anonymous" for awhile-and that's fine with us! Come in, enjoy the relaxed atmosphere, and see if this is the church for you.

Figure 7. Top words spotted by the event representation model

capture different aspects of underlying semantic knowledge. There are also other informative words ranked high but not made into top five. For example, all three words in "ice cream makers" are ranked high in the short text example. In the long text example, "family", "baptism", and "hunt" in "egg hunt" are words immediately following the top-five words.

To further demonstrate the effectiveness of our model in capturing semantic knowledge, we include examples of similar events that are discovered by our representation model. Using the event representation model alone, we derive a representation vector for each event and compute event-to-event similarity just as we compute user-to-event similarity. Setting a high threshold in similarity score (0.95), we identify many event pairs that are similar in semantic topics but do not necessarily overlap much in the word space. Table 3 illustrates the top three similar events discovered from a given seed event in food category. Due to space limit, we only show the title of the events.

6. Conclusion and Future Directions

Event recommendation in social networks deals with cold-start problems in both user and event domains. Matching user knowledge from heterogeneous sources to event semantic knowledge is a way to overcome challenges in event recommendation. In this work, we investigated a Convolutional Neural Network based approach to building user and event representation jointly. On the user side, our representation model allows flexible input of user knowledge in addition to sparse event attendances. It can utilize rich user attributes and products (e.g., page or group) that have much longer lifespans than events do. The user interaction history on these products can then be transferred to help solve the cold-start problem in event recommendation. On the event side, we have shown with examples that our model captures well event semantic knowledge and it can be projected with users into the same space for matching. The representation learning method proposed in this work boosts prediction accuracy and provides a generic way to match heterogeneous information from different domains. It provides a solution to solve cold-start problems with rich heterogeneous information and applies to a wide range of

applications where heterogeneous information is available to help.

In the future, we plan to consider more types of user feedback in addition to attendance. For instance, clicks and views information could be integrated into the training process. We also plan to extend our work to other entity recommendation problems in social networks.

References

[1] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '99. New York, NY, USA: ACM, 1999, pp. 50–57. [Online]. Available: http://doi.acm.org/10.1145/312624.312649

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[3] C. C. Chen and Y.-C. Sun, "Exploring acquaintances of social network site users for effective social event recommendations," *Inf. Process. Lett.*, vol. 116, no. 3, pp. 227–236, Mar. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.ipl.2015.11.013

[4] B. Liu and H. Xiong, "Point-of-interest recommendation in location based social networks with topic and location awareness." in *SDM*, vol. 13. SIAM, 2013, pp. 396–404.

[5] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 448–456.

[6] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers *et al.*, "Practical lessons from predicting clicks on ads at facebook," in *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM, 2014, pp. 1–9.

[7] T. De Pessemier, S. Coppens, K. Geebelen, C. Vleugels, S. Bannier, E. Mannens, K. Vanhecke, and L. Martens, "Collaborative recommendations with content-based filters for cultural activities via a scalable event distribution platform," *Multimedia Tools Appl.*, vol. 58, no. 1, pp. 167–213, May 2012. [Online]. Available: http://dx.doi.org/10.1007/s11042-010-0715-8

[8] M. Kayaalp, T. Özyer, and S. T. Özyer, "A collaborative and content based event recommendation system integrated with data collection scrapers and services at a social networking site," in *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining*, ser. ASONAM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 113–118. [Online]. Available: http://dx.doi.org/10.1109/ASONAM.2009.41

[9] Y. Zhang, H. Wu, V. S. Sorathia, and V. K. Prasanna, "Event recommendation in social networks with linked data enablement." in *Proceedings of the 15th International Conference on Enterprise Information Systems*, ser. ICEIS '13', 2013, pp. 371–379.

[10] E. M. Daly and W. Geyer, "Effective event discovery: Using location and social information for scoping event recommendations," in *Proceedings of the Fifth ACM Conference on Recommender Systems*, ser. RecSys '11. New York, NY, USA: ACM, 2011, pp. 277–280. [Online]. Available: http://doi.acm.org/10.1145/2043932.2043982

[11] M. Bogaert, M. Ballings, and D. Van den Poel, "The added value of facebook friends data in event attendance prediction," *Decis. Support Syst.*, vol. 82, no. C, pp. 26–34, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2015.11.003

[12] R. Shaw, R. Troncy, and L. Hardman, "Lode: Linking open descriptions of events," in *The Semantic Web*. Springer, 2009, pp. 153–167.

[13] H. Khrouf and R. Troncy, "Hybrid event recommendation using linked data and user diversity," in *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 2013, pp. 185–192.

[14] A. Rajaraman, J. D. Ullman, J. D. Ullman, and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press Cambridge, 2012, vol. 1.

[15] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.

[16] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1235–1244.

[17] D. Agarwal and B.-C. Chen, "flda: matrix factorization through latent dirichlet allocation," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 91–100.

[18] H. Shan and A. Banerjee, "Generalized probabilistic matrix factorizations for collaborative filtering," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 1025–1030.

[19] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, vol. 2. IEEE, 2006, pp. 1735–1742.

[20] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2333–2338.

[21] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, ser. CIKM '14. New York, NY, USA: ACM, 2014, pp. 101–110. [Online]. Available: http://doi.acm.org/10.1145/2661829.2661935

[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[23] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[25] R. Collobert, S. Bengio, and J. Mariéthoz, "Torch: a modular machine learning software library," IDIAP, Tech. Rep., 2002.

[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[27] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 539–546.

[28] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[29] N. Bronson, Z. Amsden, G. Cabrera, P. Chakka, P. Dimov, H. Ding, J. Ferris, A. Giardullo, S. Kulkarni, H. C. Li *et al.*, "Tao: Facebook's distributed data store for the social graph." in *USENIX Annual Technical Conference*, 2013, pp. 49–60.