**Dynamic Graph Attention Network**

1. **Sessions**

* Let denote a set of volunteers and a set of organisers. Each volunteer is associated with a set of organisers ordered by timestamps :



where is the -th session of volunteer .

* A session can be viewed as a sequence of volunteer participation in a short horizon.
* Each session is characterized by a set of user behaviours:



where is the total amount of organisers in a session and is the -th organizer ‘consumed’ by user in the -th session.

* Individual volunteer’s preferences are represented as short-term sessions given by



* Individual volunteer’s current session is given by:



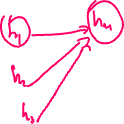
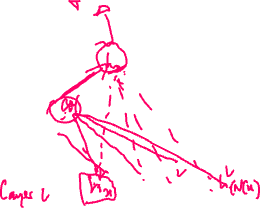
* Neighbor’s actions are given by:



* Neighbor’s short-term preference is given by



* To reduce computational cost, the friends’ short-term interests were represented by the most recent sessions.



A modified schematic view of DGRec for dynamic volunteer recommendation system

Diagram

Description automatically generated



1. **Volunteer’s individual interests**



Individual interests of volunteers are expressed as short-term dynamic interests which are modeled using Recurrent Neural Network (RNN). We represent short-term dynamic preferences of a user as their most recent session: where each token, in the session corresponds to the volunteer’s participation for a task initiated by a particular organizer . RNN infers a user’s current session: by recursively combining representations of all previous tokens with the latest token: , where represents the user’s interests and is a non-linear function (LSTM) denoted as follows:



is the sigmoid activation function.

1. **Neighbor’s short-term interests.**

The interests of a neighbor are categorised as short and long-term preferences. For a target user’s current session , the neighbour’s short-term interests are represented by their sessions just before session . Neighbor’s latest task participation, is modeled by RNN, where each token in the session corresponds to the neighbor’s participation in a task initiated by an organizer . The final output of the short-term interests is given by . Here we use instead of to capture time dependencies.

* 1. **Neighbour’s long-term interests**

Since long-term interests express a neighbour’s average preference, we represent them with a time insensitive single vector , where is the corresponding -th row of the user embedding matrix, . The user embedding matrix is a matrix reflecting a volunteer’s task participation.

We finally combine the short and long-term preferences of a neighbour using a non-linear transformation where is the transformation matrix and .

1. **Volunteer task preference**

One important task is to represent volunteers’ participation in organizers’ tasks while inherently capturing their preferences. Each task is characterized by location and we associate each user with task location denoted by . Additionally, we use LDA to extract topics from task description. Six main topics were identified and associated with each task, denoted as . We define volunteers’ task-based preference as a joint combination of task type and its corresponding location, represented as , where is opinion aware interaction volunteer representation, and is a Multi-Layer Perceptron (MLP) fusing the information of task type and its location.

1. **Unified Graph**

We construct the social network graph, where nodes indicates target volunteers and their neighbors; and edges correspond to their friendship. Each node utilizes its respective user’s dynamic representations as features.

A target user’s behavior depends on his/her dynamic individual preference and social influence from neighbors. Therefore, a unified representation of a volunteer’s preference is obtained by using an attention mechanism to determine the weight of neighbor on target volunteer that is propagated along the edges of the network.

The dynamic feature graph is constructed such that the nodes correspond to the user and her friends, that is, for target user , with neighbors, the graph consists of nodes. The initial representation of target volunteer is used as node features . Similarly, a neighbor’s node features are given by the combined representation of the short and long-term interests . While the features of the target volunteer changes whenever the volunteer participates in a new task, the neighbor’s features remain unchanged for the duration of the timestep .

Following DGRec, Song et al (2019), the features of each node are propagated using an attention mechanism as follows:

where is the similarity between target volunteer and all aggregated neighbors’ preferences and represents the weight of neighbor on target volunteer . This design preserves volunteer’s individual preference while harnessing the impact of each neighbor

A self-connection edge to reflect a user’s revealed interest, , is also included and we further provide the weights to combine the features as follows:

where represents a mixture of user ’s neighbours’ interests at layer , then followed by a non-linear transformation and is the learned weight matrix at layer The final representation of each node is obtained by stacking this attention layer times.

1. **Recommendation**

In our framework, a volunteer’s interest is determined by her dynamic short-term preference and the social influence from neighbors. The final representation of a volunteer is therefore given by integrating volunteer’s recent behavior and social influence as follows:

, where denotes the user’s dynamic interests and is the combined final user’s representation that reflects social influence. is a linear transformation matrix.

We let to be the next organizer and predict the probability of using a softmax function:

where is the embedding of organiser and is the total number of organisers. corresponds to ’s set of friends while denotes the neighbor’s actions.

1. **Training**

Since we are considering user ’s preference and influence from social network, the training loss is defined by maximizing the log likelihood of observed organisers in all user sessions as follows:

The loss function is optimized using gradient descent.