Attributed Multi-Relational Attention Network for Fact-checking URL Recommendation

Di You, Nguyen Vo, Kyumin Lee
Worcester Polytechnic Institute
Massachusetts, USA
{dyou,nkvo,kmlee}@wpi.edu

Qiang Liu
Alibaba Group
Hang Zhou, China
liuq0326@126.com

ABSTRACT

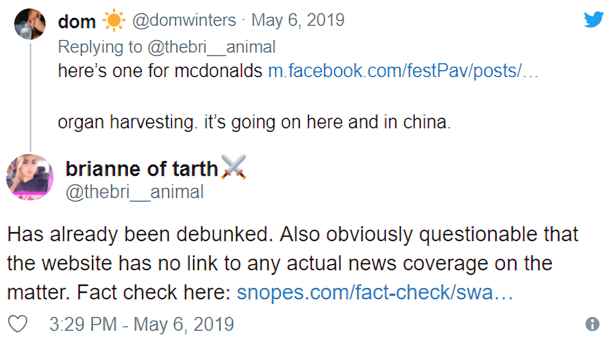
To combat fake news, researchers mostly focused on detecting fake news and journalists built and maintained fact-checking sites (e.g., Snopes.com and Politifact.com). However, fake news dissemination has been greatly promoted via social media sites, and these fact-checking sites have not been fully utilized. To overcome these problems and complement existing methods against fake news, in this paper we propose a deep-learning based fact-checking URL recommender system to mitigate impact of fake news in social media sites such as Twitter and Facebook. In particular, our proposed framework consists of a multi-relational attentive module and a heterogeneous graph attention network to learn complex/semantic relationship between user-URL pairs, user-user pairs, and URL-URL pairs. Extensive experiments on a real-world dataset show that our proposed framework outperforms eight state-of-the-art recommendation models, achieving at least 3~5.3% improvement. Our source code and dataset are available at *https://web.cs.wpi.edu/~kmlee/data.html*.

ACM Reference Format:
Di You, Nguyen Vo, Kyumin Lee and Qiang Liu. 2019. Attributed Multi-Relational Attention Network for Fact-checking URL Recommendation. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3358006

1 INTRODUCTION

While social media sites provide users with the revolutionized communication medium by bringing the communication efficiency to a new level, they can be easily misused for widely spreading misinformation and fake news. Fake news and misinformation have been a long-standing issue for various purposes such as political propaganda [1] and financial propaganda [24].

To fight against fake news, traditional publishers employed human editors to manually and carefully check the content of news articles to maintain their reputation. However, social media provided a new way to spread news, which lead to broader information sources and expanded audience (i.e., anyone can be a media and create news). In particular, users share news articles with their own opinion or read articles shared by their friends from whatever the source of news is with mostly blind trust [37] or with their own ideologies [8, 30]. Although social media posts usually have a very short life cycle, the unprecedented amount of fake news may lead to a catastrophic impact on both individuals and society. Besides from misleading users with false information [30], widely propagated fake news could even cause trust crisis of entire news ecosystem [36], even further affecting both the cyberspace and physical space.


dom ☀️ @domwinters · May 6, 2019
Replying to @thebri__animal
here's one for mcdonalds m.facebook.com/festPav/posts/…

organ harvesting. it's going on here and in china.

brianne of tarth ⚔️
@thebri__animal

Has already been debunked. Also obviously questionable that the website has no link to any actual news coverage on the matter. Fact check here: snopes.com/fact-check/swa…

3:29 PM - May 6, 2019

Figure 1: A real-world example of fact-checking behavior. *thebri_animal* is a fact-checker, who corrects the false claim with a fact-checking URL/article containing factual evidences.

In literature, researchers focused on four topics regarding fake news: characterization (i.e., types of fake news), motivation, circulation, and countermeasures [21, 52]. A large body of work has been done on fake news identification [3, 36, 40, 49] by exploiting multiple content-related and social-related components. However, we notice that the fake news still has been widely spread even after early detection [11]. Therefore, we propose to study a complementary approach to mitigate the spread and impact of fake news. Recently, community and journalists started building and maintaining fact-checking websites (e.g., Snopes.com). Social media users called *fact-checkers* also started using these fact-checking pages as factual evidences to debunk fake news by replying to fake news posters. Figure 1 demonstrates a real-world example of a fact-checker's fact-checking behavior on Twitter by debunking another user's false claim with a Snopes page URL as an evidence to support the factual correction.

In [44], researchers found that these fact-checkers actively debunked fake news mostly within one day, and their replies were exposed to hundreds of millions users. To motivate these fact-checkers further quickly engage with fake news posters and intelligently consume increased volume of fact-checking articles, in this paper we propose a novel personalized fact-checking URL recommender system. According to [29], co-occurrence matrix within the given context provides information of semantic similarity between two objects. Therefore, in our proposed deep-learning based recommender system, we employ two extended matrices: user-user co-occurrence matrix, and URL-URL co-occurrence matrix to facilitate our recommendation. In addition, users tend to form relationships with like-minded people [31]. Therefore, we incorporate each user's social context to capture the semantic relation to enhance the recommendation performance.

Our main contributions are summarized as follows:

- We propose a new framework for personalized fact-checking URL recommendation, which relies on multi-relational context neighbors.
- We propose two attention mechanisms which allow for learning deep semantic representation of both a target user and a target URL at different granularity.
- Experimental results show that our proposed model outperforms eight state-of-the-art baselines, covering various types of recommendation approaches. Ablation study confirms the effectiveness of each component in our proposed framework.

2 RELATED WORKS

In this section, we briefly review related works and position our work within the following areas: (1) fake news and misinformation; (2) advancements in recommender systems; and (3) graph convolutional networks.

2.1 Fake News and Misinformation

Fake news has attracted considerable attention since it is related to our daily life and has become a serious problem related to multiple areas such as politics [1] and finance [24]. Social media sites have become one of popular mediums to propagate fake news and misinformation. The dominant line of work in this topic is fake news detection [35] which was mostly formulated as a binary classification problem. Researchers began to incorporate social context and other features for identifying fake news at an early stage and preventing it from diffusion on the social network [36, 52]. Some other researchers focus on investigating the propagation patterns of fake news in social network [28, 50]. [45] also studied fake news intervention. Unlike most previous works, we follow the direction of [44] and propose to build a personalized recommender system for promoting the fact-checking article circulation to debunk fake news.

2.2 Advancements in Recommender System

Traditionally, recommendation algorithms can be divided into two categories: collaborative filtering [32] and content-based filtering. However, in the past few years, the recommendation has become a more integrated task due to the success of the deep neural network. Neural Networks (NNs) proves to be effective to capture underlying nonlinear relations [19]. Another advantage is that the NNs enhanced the model's capability of extracting knowledge from multimodal data [16, 41, 46], which serves as auxiliary information and provide solutions to address the data sparsity problem. More recently, researchers introduced attention mechanism into recommender systems, which has achieved great success in various fields [2, 42]. Researchers developed multiple variants of attention mechanism to improve both the recommendation precision and model interpretability [5, 34, 48, 53].

In this paper, we also propose two novel designs of attention mechanism. Following [7, 18], we further explore multi-relational context of given user-URL pair, aiming at discriminating the most important elements towards URL-dependent user preference.

2.3 Graph Convolutional Networks

With the surge of Graph-based Neural Network, GCN-based approaches have shown strong effectiveness on various tasks [12, 14, 23], including recommender system. The core idea is to iteratively aggregate attributed node vectors around each node, and messages propagates by stacking multiple layers. However, the original design of GCN is not suitable for our scenario because of the following reasons: First, existing GCN works [12, 14] do not distinguish different types of nodes, whereas in our case, it does not make sense to aggregate user and URL nodes together. And the aggregation function proposed in most GCN works treats all its adjacency nodes with the same importance. It is inappropriate in real-world applications and probably tends to neglect necessary information. [43] breaks this schema by using a multi-head attention mechanism to replace the convolution-like operator, yet it requires significant extra computation and memory.

Compared to the previous works, in this paper, we focus on a novel application and investigate both co-occurrence context and social context related influences for fact-checking URL recommendation. We also incorporate sets of auxiliary attributes, which enable more comprehensive learning of the compatibility between given pairs of user and URL. Moreover, we take advantage of advancements in graph neural networks and attention mechanisms, and solve the aforementioned research problems.

3 PROBLEM FORMULATION

We formally introduce definitions before describing our proposed framework. We define fact-checking behavior as a user (i.e., *fact-checker*[1]) embeds a fact-checking URL in his reply in order to debunk fake news. We regard each fact-checking behavior as an implicit interaction between target user i and target URL j.

Definition 1 (Fact-checking URL Recommendation Task). Let $\mathcal{U} = \{u_1, u_2, ..., u_n\}$ denotes a set of fact-checkers on social media, and use $C = \{c_1, c_2, ..., c_m\}$ to index fact-checking URLs. We construct user-URL interaction matrix $Y = \{y_{ij} | u \in \mathcal{U}, v \in C\}$ according to users' fact-checking behavior, where

$$y_{ij} = \begin{cases} 1, \text{if } (u_i, c_j) \text{ interaction observed,} \\ 0, \text{otherwise.} \end{cases} \tag{1}$$

each value of 1 for y_{ij} indicates the existence of implicit interaction between target user i and target URL j. Each user u_i and each URL c_j associate with a set of attributes. The goal of the recommendation task is to recommend top-N URLs from the URL set C to each user.

[1]We use terms user and fact-checker interchangeably in the paper.

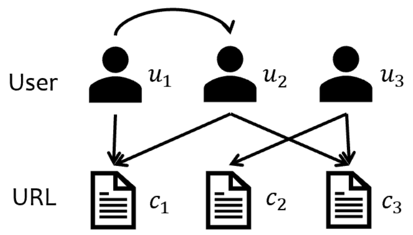


Figure 2: A toy example of multi-relational context w.r.t. given target user-URL pair.

We also construct the entire dataset as a heterogeneous graph, which is a special kind of information network that consists of either multiple types of objects or different types of links, or both.

Definition 2 (Heterogeneous Network) [38]. Formally, consider a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}(|V| = m + n)$ and E denote the node set and edge set, respectively. The heterogeneity represents by the node type mapping function: $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and edge type projection function: $\psi : \mathcal{E} \rightarrow \mathcal{R}$, where $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined node types and edge types, and $|\mathcal{A}| + |\mathcal{R}| > 2$. Note that we do not consider self-loop in our graph construction.

Definition 3 (Multi-relational Context). Given target user i, we define his following fact-checkers and co-occurrenced fact-checkers as his social context user neighbors and co-occurrenced context user neighbors, respectively. Similarly, we name the other URLs posted by target user i and co-occurrenced URLs of target URL j as historical context URL neighbors and co-occurrenced context URL neighbors, respectively. In general, we call all the context neighbors as multi-relational context of given target user-URL pair.

Example. Figure 2 illustrates the multi-relational context. In Figure 2, c_1, c_2, c_3 represents fact-checking URLs and u_1, u_2, u_3 are users who involve sharing these URLs. For example, $(u_1 \rightarrow u_2)$ indicates the social relationship between u_1 and u_2. Intuitively, we care more about the influence of u_2 on u_1. $(u_1 \rightarrow c_1 \leftarrow u_2)$ means u_1 and u_2 are co-occurrenced user neighbors. Similarly, we name c_1 and c_2 as co-occurrenced URL neighbors of u_3, and c_2 is historical context URL neighbor given target u_3-c_3 pair.

4 PROPOSED FRAMEWORK

We propose a novel framework called Attributed Multi-Relational Attention Network (*AMRAN*), to understand the influence of the multi-relational context to target user's fact-checking behavior. In this section, we elaborate our proposed AMRAN with using notations described in Table 1.

At the high level, *AMRAN* is composed of two modules as shown in Figure 3: (i) a convolutional spatial attention network (*CSAN*) and (ii) a heterogeneous graph attention network (*HGAN*). *CSAN* jointly models the influence of multi-relational context on target user-URL pair (Section 4.1). It enriches the neighborhood diversity, and expands the scope of information reception. *HGAN* leverages both global node connectivity and local node attributes, in order to incorporate the effect of information propagation and encode user's dynamic preference in depth (Section 4.2). At the final step, the model produces recommendations by combining wide context-aware target user embedding and URL embedding, multi-relational context user embedding and context URL embedding, and deep context-aware user embedding and URL embedding (Section 4.3).

Table 1: Notations.

Notations	Description
b_h	# of selected relation-based neighbors
S	Spatial weight tensor
L	Layer-wise weight tensor
C	Channel-wise wight tensor
M	Initial embedding matrix of each neighbor
N	Attended embedding matrix of each neighbor
A_{ij}	Weighted adjacency matrix in graph
W_{ϕ_i}	Node type specific transformation matrix
$\mathcal{N}_i^{\phi_t}$	Node type specific neighbor nodes
$e_{ij}^{\phi^{(l)}}$	Importance between node pair (i, j) at layer l
$\alpha_{ij}^{\phi^{(l)}}$	Weights between node pair (i, j) at layer l
p_i	Neighborhood embedding of user i
p_j	Neighborhood embedding of URL j
u'_i	Wide context-based embedding of user i
c'_j	Wide context-based embedding of URL j
$h_i^{(l)}$	Deep context-based embedding of node i

4.1 Convolutional Spatial Attention Network (CSAN)

The left bounding box in Figure 3 illustrates the structure of CSAN module. To provide a broad scope of knowledge for generating wide context-aware target user embedding and URL embedding, we adopt a multi-branch setting in CSAN. The two parallel branch models multi-relational context for target user and target URL respectively. Each branch contains two identical streams. We select b_h context neighbors for each stream (e.g., historical context URL neighbors and co-occurrenced context URL neighbors of target URL, social context user neighbors and co-occurenced user neighbors of target user). These streams are employed to learn the most discriminative features from multi-relational neighbors of target user and target URL. Then we employ a gated fusion layer to capture the optimal global level representation of target user-URL pair.

Note that we enable the embedding sharing within each branch as users/URLs share the same feature set.

4.1.1 Raw Attribute Input. User and URL associate with different feature sets. Therefore, CSAN starts from embedding the input attribute set of each context neighbor. We use s and t to denote the number of features related to user and URL, respectively. Note that the dimension of initial embedding for each attribute could be different since they may carry with different information volume. We use one-hot encoding for categorical feature inputs, and apply direct lookup on these features. However, the same solution performs poorly when it comes continuous attributes such as the post frequency of an URL. Empirically, we found that an available solution is to bucketize these features into small intervals. Specifically, we map these continuous attributes in range $[0, 1), [1, 2), ..., [2^k, 2^{k+1})$ into $0, 1, ..., k$ in this work.

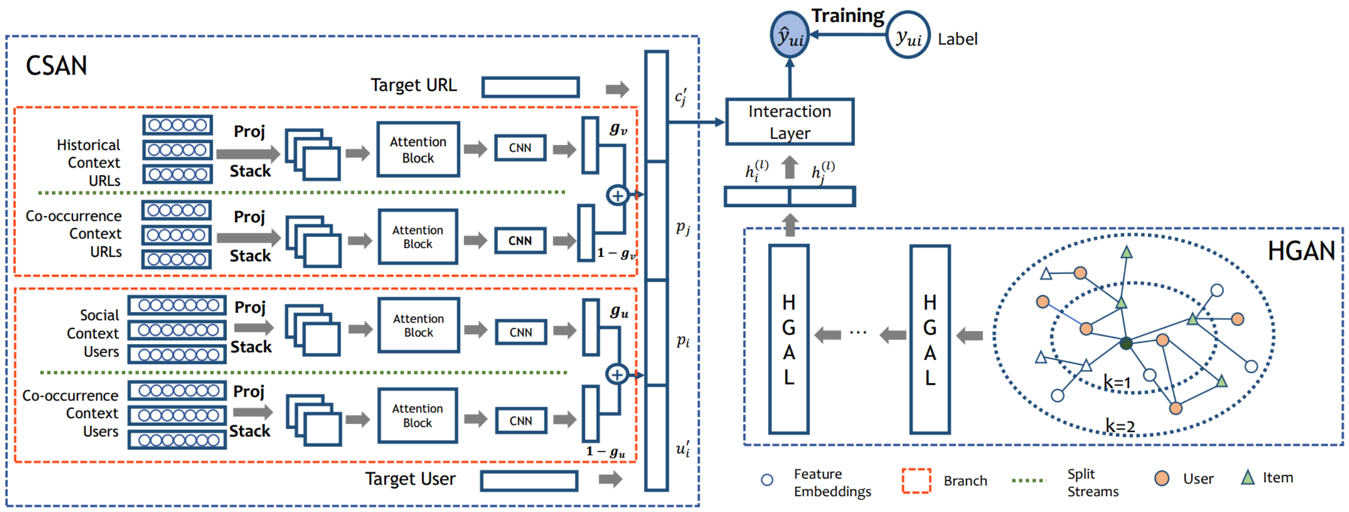


Figure 3: A schematic overview of our proposed Attributed Multi-Relational Attention Network (AMRAN), consisting of two modules: (1) a convolutional spatial attention network (CSAN); and (2) a heterogeneous graph attention network (HGAN).

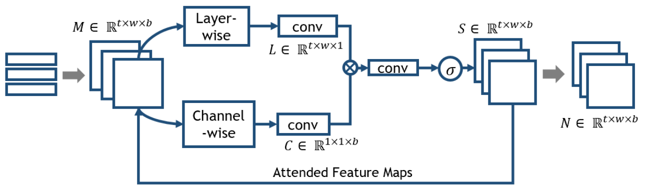

$M \in \mathbb{R}^{t\times w\times b}$
Layer-wise
conv
$L \in \mathbb{R}^{t\times w\times 1}$
conv
σ
$S \in \mathbb{R}^{t\times w\times b}$
Channel-wise
conv
$C \in \mathbb{R}^{1\times 1\times b}$
$N \in \mathbb{R}^{t\times w\times b}$
Attended Feature Maps

Figure 4: The illustration of Spatial Attention Mechanism (show an attention block in the historical context URL stream for illustration).

4.1.2 *Attribute Embedding Layer.* We then project them into the same latent space via a set of attribute-specific transformation matrices $W_1, W_2, ..., W_{s+t}$ to project all the attributes into a w-dimensional space. The attributes of each neighbor then are stacked as a matrix in shape of $s \times w$ for users and $t \times w$ for URLs.

However, we treat the target user-URL pair differently. After projecting attributes by the same attribute-specific transformation matrix as their relational neighbors, instead of stacking them as a matrix, we concatenate the attribute embedding vectors together and feed it through a linear projection to generate $u_i' \in \mathbb{R}^d$ and $c_j' \in \mathbb{R}^d$ for future reference.

4.1.3 *Spatial Attention Block.* To prevent some unknown misalignment and conduct better comparison among the neighborhood features, we proposed a schema for jointly learning the layer-wise and channel-wise attention. In particular, for each stream, we pile the neighbors' representation matrices together to obtain a 3-dimensional tensor M. Intuitively, the design helps improve the alignment quality of neighbor's features. Then, inspired by [20, 27], we employ a spatial attention block in each stream for jointly learning channel-level and layer-level soft attention. See figure 4 for a high-level illustration of our spatial attention block. All the streams adopt identical spatial attention blocks, and each block attends the input attribute representations independently.

In the figure, we use the historical context URL stream for illustration. The output of spatial attention block is an attention weight map $S \in \mathbb{R}^{t\times w\times b}$ which is in the same shape with the input tensor M. Intuitively, the layer-wise attention and channel-wise attention are dedicated to selecting the most discriminative features and the most important neighbors, respectively. Thus, they are highly complementary to each other in functionality; and we adopt a factorized manner for optimization and computational efficiency as:

$$S = L \times C \tag{2}$$

where $L \in \mathbb{R}^{t\times w\times 1}$ and $C \in \mathbb{R}^{1\times 1\times b}$ denote the layer-wise feature map and channel-wise feature map, respectively. S is the result of tensor multiplication.

Layer-wise Attention. Conceptually, the layer-wise attention learns globally important elements in the feature. We apply a cross-channel average pooling operation onto the input tensor, following by 2 convolution layers of 3×3 and 1×1 filter, respectively. Specifically, cross-channel average pooling operation is defined as:

$$L = \frac{1}{b}\sum_{b'=1}^{b} M_{1:t,1:w,b'} \tag{3}$$

where b is the number of selected neighbors.

Channel-wise Attention. The design of channel-wise attention is very similar to layer-wise attention, which aims to acquire a global view of discriminative users. Formally, the global average pooling is defined as:

$$C = \frac{1}{t\times w}\sum_{w'=1}^{w}\sum_{t'=1}^{t} M_{t',w',1:b} \tag{4}$$

where t and w are shared height and width of all channels. Similarly, we employ two convolution layers after the pooling operation.

Note that each convolution layer was followed by batch normalization operation. Furthermore, as other work of modern CNN structure [39], we append a ReLU activation function to assure $L > 0, C > 0$.

We further introduce one more convolution layer of $1\times1\times b$ filter for enhancing the fusion of the layer-wise attention and channel-wise attention. The output tensor then is fed through a sigmoid function for normalization and generate the final attention weight tensor of spatial attention block. Formally, the output of the spatial attention module is the element-wise product of initial feature tensor M and generated attention weights S:

$$N = M \odot S \tag{5}$$

Intuitively, the attended feature map learned fine-grained important elements via high alignment and compatible attentions.

4.1.4 Gated Branch Fusion Layer. We apply another CNN layer of 3×3 filter after the attended user representation of each stream for feature extraction and dimension :

$$N_{op} = ReLU(WN) \tag{6}$$

$$p^k = MAXPOOLING(N_{op}) \tag{7}$$

which produces the multi-relational context representation vectors: $o_{i_h}, o_{i_c}, o_{u_f}$ and o_{u_c} for each stream, respectively.

We employ a gated mechanism to assigns different weights to relation-specific neighborhood representation as:

$$p_i = g_u \cdot o_{u_f} + (1 - g_u) \cdot o_{u_c} \tag{8}$$

$$p_j = g_v \cdot o_{i_h} + (1 - g_v) \cdot o_{i_c} \tag{9}$$

where scalars g_u and g_v are learned automatically to control the importance of the two streams within each branch.

4.2 Heterogeneous Graph Attention Network (HGAN)

Following recent success in Graph Convolutional Network (GCN) [12, 14, 23, 33, 43]. We propose a heterogeneous graph attention network (HGAN) which is tailored for recommendation task. In particular, our proposed module adopts a parallel attention structure for the user neighbor and the URL neighbor of the central node, respectively. Considering a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the nodes represent objects in this network which can be either user or URL. The edges denote the relation between connected nodes. The node attributes pass along the edges during the propagation. We try to leverage between the local node attributes and global network structure. Our novelty lies in two aspects: (i) we differentiate the contribution of URL node and user node, respectively; and (ii) we consider both similarities of node and the influence of different relation types.

While the CSAN obtains information from multi-relational immediate neighbors, which expand the scope of knowledge for target user and target URL representations, HGAN aims at learning deeper semantic representations of target user and target URL.

4.2.1 Heterogeneous Graph Network. We try to capture different semantic relation behind various types of nodes and edges. For every single layer, if the central node is user node, its neighborhood contains its co-occurrenced users and posted URLs. If the central node type is URL, its neighborhood nodes consist of users who posted it and its co-occurrenced URLs.

We adopt similar embedding approach as we did in CSAN for the initial representation of each node, but we concatenate all the features into a long vector x_i for each node instead of stacking them as a matrix. Considering the different types of the node associated with the varied feature set, we use a set of node type-specific transformation matrices to project different types of node representation into the same feature space before aggregation as follows:

$$h_i^{(0)} = W_{\phi_i} \cdot x_i \tag{10}$$

Let $H^{(0)} \in \mathbb{R}^{(m+n)\times d}$ be the embedding matrix of all the attributed nodes, where $m + n$ is the total number of nodes and d is the dimension of latent embedding space; each row $h_i^{(0)}$ stands for the initial embedding vector of node i.

We define edges based on users' reference of URL (user-URL edges), user co-occurrence relation (user-user edges), and URL co-occurrence (URL-URL edges). We then introduce an adjacency matrix A of $\mathcal{G}$ based on the importance of each edge. In particular, to compute the weight of user-user edges and URL-URL edges, we adopt a matrix named Shifted Positive Point-wise Mutual Information (SPPMI) [26], a popular measure for word associations, to utilize the co-concurrence context information. In word embedding scenario, each cell within the matrix measures the relation of corresponding word-context pair. The factorization of such matrix is proved to be equivalent to skip-gram model with negative sampling (SGNS). The Point-wise Mutual Information (PMI) between node i and node j is computed as $PMI(i,j) = log\frac{P(i,j)}{P(i)P(j)}$ where $P(i,j) = \frac{\#(i,j)}{|D|}$ and $P(i) = \frac{\#(i)}{|D|}$. $|D|$ denotes the total number of observed word-context pairs within a predefined sliding window. $P(i,j)$ is the joint probability that word i and word j appear together within the window size. Furthermore, we introduce the SPPMI matrix as an extension based on PMI value:

$$SPPMI(i,j) = max\{PMI(i,j) - log(k), 0\} \tag{11}$$

where k is a hyperparameter, which represents the number of negative samples. Conceptually, a positive PMI value implies a semantically correlated word-context pair, Therefore, SPPMI, which only takes the positive value of PMI shifted by a global constant, reflects a closer semantic relation between word-context pairs. Inspired by this concept/idea, we use $|D|$ to denote the number of times of user (URL) co-occurrence and generate the user co-occurrence matrix in shape of $n \times n$ and URL co-occurrence matrix of $m \times m$. Note that we do not discriminate between the target node and context node.

Similarly, we learn from the TF-IDF concept and redefine it on recommendation task with implicit feedback [10] as:

$$TF - IDF_{ij} = TF_{ij} \times IDF_i = \frac{\#(i,j)}{\max_k \#(i,k)} log\frac{m}{m_i} \tag{12}$$

where $\#(i,j)$ represents the number of times URL j be posted by user i. TF_{ij} further normalizes it by the maximum number of post times of any URL by user i. The IDF_i is associated with the user's previous behavior as m denotes the total number of URLs and m_i is the number of URLs posted by user i.

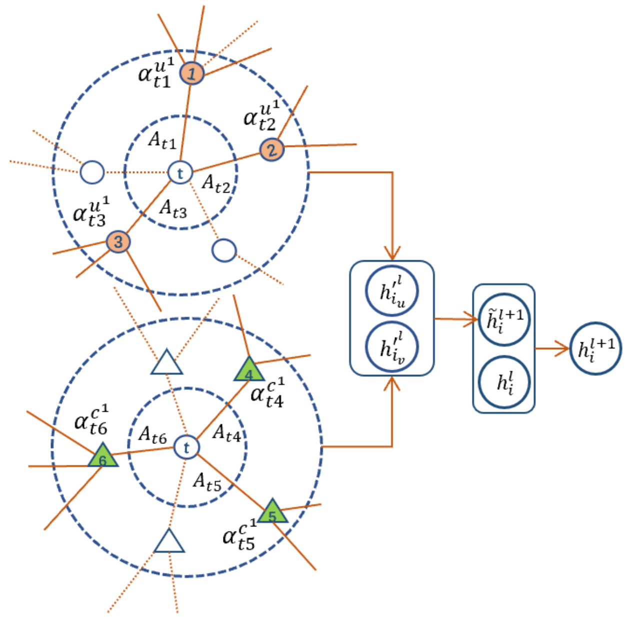


Figure 5: Graphical illustration of a single heterogeneous graph attention layer. In this example, we assume the central node as a user node. Circles denote users, and triangles denote URLs. Colored objects with a solid line are selected neighbors at each layer, and the nodes with a dotted line are randomly dropped. (Best viewed in color).

Formally, the weight of the edge between node i and node j is defined as:

$$A_{ij} = \begin{cases} SPPMI(i,j) & i,j \text{ are user (URL)} \\ TF-IDF_{ij} & i \text{ is user, } j \text{ is URL} \\ 1 & \text{i=j,} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

4.2.2 Heterogeneous Attention Layer (HGAL). Given the node's initial representation defined as above, we then pass messages to aggregate the neighborhood nodes' information and combine it with the target user's interests. A popular propagation strategy in existing GCN works is the normalized Laplacian matrix [23]. Even though it proves to be effective, it is not trainable and it assigns every adjacent node with the same weight. Following previous work [43], we propose to incorporate a hierarchical attention mechanism to learn the weight of each adjacent node adaptively.

Since the distribution of the number of neighbors of each node disperses greatly, sub-sampling becomes an essential procedure in our task to avoid an explosion of computation cost after multiple hops stacked. We adopt Weighted Random Selection (WRS) [9] to select a fixed number of nodes for both node types in each graph attention layer. Figure 5 shows a graphical illustration of one HGAL.

Assume that the central node is a user node. We separately calculate the attention weights between the user node and its user node neighbors, or between the user node and its URL node neighbors. The similarity between the target user's node representation $h_u^{(l)}$ and all of its selected neighbors are defined as:

$$\alpha_{ij}^{\phi(l)} = softmax(e_{ij}^{\phi(l)}) = \frac{exp(f(h_i^{(l)}, h_j^{(l)}))}{\sum_{k \in \mathcal{N}_i^{\phi_t}} exp(f(h_i^{(l)}, h_k^{(l)}))} \tag{14}$$

where $h_i^{(l)}$ is the representation of user i at layer l, and $\mathcal{N}_i^{\phi_t}$ denotes the node type-based neighbor. We adopt $f(h_i^{(l)}, h_j^{(l)}) = cosine(h_i^{(l)}, h_j^{(l)})$ as similarity function. Intuitively, α_{ij}^{ϕ} measures the importance of neighbor j towards central node i. Meanwhile, we obtain the edge weight A_{ij} as well.

After this, we aggregate the type-based neighborhood node representation and generate the embedding of neighborhood as the average of different types of nodes:

$$z_{ij} = ReLU(A_{ij} h_i^{(l)}) \tag{15}$$

$$\tilde{h}_i^{(l+1)} = \frac{1}{|\mathcal{A}|}\Big(\sum_{j \in \phi_{\mathcal{U}}} \alpha_{ij}^{\phi(l)} z_{ij} + \sum_{j \in \phi_{\mathcal{C}}} \alpha_{ij}^{\phi(l)} z_{ij}\Big) \tag{16}$$

To model the information propagation and capture higher-order relations, we stack the HGAL multiple times. In addition, we introduce the residual connection [15] to help train a HGAN with many layers.

$$g^{(l+1)} = \sigma(W_g^{(l)} h^{(l)} + b_g^{(l-1)}) \tag{17}$$

$$h^{(l+1)} = (1 - g^{(l+1)}) \odot \tilde{h}_i^{(l+1)} + g^{(l+1)} \odot h^{(l)} \tag{18}$$

where σ denotes the sigmoid function. $W_g^{(l)}$ and $b_g^{(l-1)}$ are the shared weight matrix and bias term at layer l, respectively. The node representation at l-th layer provides knowledge of l degrees away.

4.3 Interaction Layer

The interaction layer is tailored for recommendation tasks. Recall that we obtained wide context-based user embedding u_i' and URL embedding c_j', context representations p_i, p_j and deep context-based user embedding $h_i^{(l)}$ and URL embedding $h_j^{(l)}$ in the previous sections. Then we formulate the final URL-dependent user representation by using a fully connected layer as:

$$o_i = W_o[u_i' \oplus c_j' \oplus p_i \oplus p_j \oplus h_i^{(l)} \oplus h_j^{(l)}] + b_o \tag{19}$$

where W_o and b_o are a linear transformation weight matrix and bias term, respectively. $\oplus$ denotes vector concatenation. Note that the fully-connected layer can be replaced by other techniques (e.g. CNN). Finally, we feed it through a softmax function to calculate the probability that user interested in the given URL.

4.4 Training

We adopt the cross-entropy loss function during the training process.

$$\mathcal{L} = -\sum_{(i,j) \in Y^+ \bigcup Y^-} y_{ij} log(\hat{y}_{ij}) + (1 - y_{ij}) log(1 - \hat{y}_{ij}) \tag{20}$$

We follow a uniform sampling strategy to obtain negative samples $(i, j) \in Y^-$ from unobserved interactions. Since the entire architecture is differentiable, we use back propagation to achieve end-to-end training.

5 EVALUATION

In this section, we describe a dataset, baselines, experimental setting, and experimental results. In the experiments, we seek to answer the following research questions:

- **RQ1:** What is the performance of our model and baselines?
- **RQ2:** How beneficial is each submodule of our model?
- **RQ3:** How effective is our attention mechanisms?
- **RQ4:** What is sensitivity of our model with regard to hyperparameters?

5.1 Dataset

We evaluate our proposed model on a Twitter dataset obtained from the authors of [44][2]. The interaction behavior collected in the dataset is consistent with our definition in 3. As they did for their study, we only kept users who have at least three interactions (i.e., posting at least three fact-checking messages containing fact-checking URLs). We conducted additional preprocessing step by removing users whose posts are non-English, or their tweets were inaccessible, because some of our baselines require a fact-checker's tweets. Our final dataset consists of 11,576 users (i.e, fact-checkers), 4,732 fact-checking URLs and 63,429 interactions. The dataset also contains each user's social network information. Note that each user's social relationship is restricted within available users in the dataset. And we further take available feature values of both user and URL into consideration. For instance, a category of referred fact-checking article and the name of corresponding fact-checking website reveals linguistic characteristics such as writing style and topical interest of each URL; while the number of followers and number of followees of each user indicates the credibility and influence of the fact-checker. Statistics of the final dataset is presented in Table 2.

5.2 Baselines

To measure relative effectiveness of our model, we compare our model against eight state-of-the-art baselines including the traditional collaborative filtering method, neural network-based models, and context-aware approaches.

- **MF** [25] is a standard collaborative filtering technique. It factorizes an interaction matrix $X \in \mathbb{R}^{M \times N}$ into two matrices $U \in \mathbb{R}^{M \times d}$ and $X \in \mathbb{R}^{d \times N}$. U contains each user's latent representation, and X contains each URL's latent representation.
- **GAU** [44] is a framework specifically designed for fact-checking URL recommendation utilizing rich side information such as a user' social network, tweets, and referred fact-checking pages. It is the most relevant and domain-specific baseline.
- **NeuMF** [19] is a neural network based item recommendation algorithm. We adopted a composite version of MF jointly coupled with a MLP.

[2]https://github.com/nguyenvo09/CombatingFakeNews

Table 2: Statistics of our evaluation dataset.

Interaction #	User #	URLs #	Sparsity
63429	11576	4732	99.884%

- **CMN** [7] combines a global latent factor model with an augmented memory network to capture personalized neighbor-based structure in a non-linear fashion.
- **NAIS** [18] is an item-based collaborative filtering architecture that integrates attention mechanism to distinguish the contribution of previously consumed items. The authors proposed two versions of NAIS: (1) $NAIS_{concat}$ which concatenates two vectors to learn the attention weight; and (2) $NAIS_{prod}$ which feeds the element-wise product of the two vectors to the attention network. Therefore, we also build two versions of NAIS, and compare them with our model.
- **DeepCoNN** [51] was originally proposed for an item rating prediction task which jointly model user and item based on their textual reviews. The prior work shows that it significantly outperforms other topic modeling based methods.We re-implemented the baseline and adapted it for our recommendation task with implicit feedback.
- **NARRE** [4] is a deep neural network based framework for a item rating prediction task. It employs the attention mechanism to distinguish the importance of each review. We re-implemented the framework for our implicit feedback situation.
- **NGCF** [47] is a new recommendation framework based on graph neural network, explicitly encoding the collaborative signal in the form of high-order connectivity in user-item bipartite graph by performing embedding propagation.

Table 3 presents characteristics of baselines and our model, showing what information each model utilizes. Note that even though CMN and NAIS both utilize co-occurrence context, CMN only utilizes user co-occurrence context whereas NAIS looks into URL co-occurrence context.

5.3 Evaluation Protocol

We adopt the leave-one-out evaluation protocol to evaluate the performance of our model and baselines. The leave-one-out evaluation protocol has been widely used in top-K recommendation tasks. In particular, we held the latest interaction of each user as the test set and used the remaining interactions for training. Each testing instance was paired with 99 randomly sampled negative instances. Each recommendation model ranks the 100 instances according to its predicted results. The ranked list is judged by Hit Ratio (HR) [6] and Normalized Discount Cumulative Gain (NDCG) [17] at the position 10. HR@10 is a recall-based metric, measuring the percentage of the testing item being correctly recommended in the top-10 position. NDCG@10 is a ranked evaluation metric which considers the position of the correct hit in the ranked result. Since both modules in our framework introduce randomness, we repeat each experiment 5 times with different weight initialization and randomly selecting neighbors. We report the average score of the best performance in each training process for both metrics to ensure the robustness of our framework.

Table 3: Characteristics of baselines and our model.

	MF	GAU	NeuMF	CMN	NAIS	DeepCoNN	NARRE	NGCF	AMRAN
Implicit Feedback	√	√	√	√	√	√	√	√	√
Textual Content	\	√	\	\	\	√	√	\	\
Co-occurrence Context	\	√	\	√	√	\	\	\	√
Social Context	\	√	\	\	\	\	\	\	√
Higher-order Information	\	\	\	\	\	\	\	√	√
Deep Learning	\	\	√	√	√	√	√	√	√

5.4 Hyper-parameter Settings

We implement our framework by using Pytorch framework, initialize weight parameters by Xavier initialization [13], and optimize the model with Adam optimizer [22]. The mini-batch size is set to 128. In CSAN, we select 10 neighbors for each stream. In HGAN, we choose 8 user neighbors and 8 URL neighbors for each central node at a single layer, and the default number of graph attention layers is set to 2. If the object (i.e.g, user neighbor or URL neighbor) is not sufficient enough, we pad the sequence with zeros vectors.

In the proposed AMRAN model, all hyperparameters are tuned by using the grid-search on the validation set, which is formed by holding out one interaction of each user from the training data like the prior work [19]. We conduct the grid search over a latent dimension size from {8,16,32,64}, a regularization term from {0.1, 0.01, 0.001, 0.0001, 0.00001}, a learning rate from {0.0001, 0.0003, 0.001, 0.01, 0.05, 0.1}, and SPPMI shifted constant value *s* from {1, 2, 5, 10}. The number of negative samples w.r.t each positive interaction is set to 4. We adopt the same latent dimension size for all submodules. For a fair comparison, we also thoroughly optimize the baselines' hyperparameters by using the validation set.

5.5 RQ1: Performance of Our Model and Baselines

Table 4: Performance of our AMRAN and baseline models. AMRAN outperforms all baselines in both evaluation metrics.

Model	HR@10	NDCG@10
MF	0.537	0.364
GAU	0.589	0.372
NeuMF	0.621	0.389
CMN	0.589	0.382
NAIS_prod	0.617	0.392
NAIS_concat	0.624	0.398
DeepCoNN	0.609	0.377
NARRE	0.615	0.382
NGCF	0.600	0.373
our AMRAN	**0.657**	**0.410**

Table 4 presents performance of our model and baselines. Based on Table 3, we had the following observations. First, deep learning-based approaches usually obtained better performance than traditional models (e.g., MF and GAU). This observation makes sense because (1) traditional models failed to capture the important non-linear relationship between users and fact-checking URLs; (2) Most deep-learning based baseline models employ attention mechanism which helps better understand the semantic relation between user and URL; and (3) training tricks such as drop out and batch normalization also contribute to a better quality of training. In particular, $NAIS_{concat}$ achieves better performance than $NAIS_{prod}$. This result supports the reason (1).

The second observation is that models with text review achieve better results compared with collaborative filtering-based methods. It is not surprising since that textual content contains rich information which could be auxiliary information to implicit feedback data and thus improves the recommendation accuracy. However, we observed that text-based recommendation approaches usually have a high complexity. Third, social context and co-occurrence context play important roles in improving recommendation results. NAIS significantly outperforms CMN and becomes the strongest baseline model. It indicates that URL-URL co-occurrence relationship is more important than user-user co-occurrence relationship since semantic representation of each user is much complex than semantic representation of a fact-checking URL.

Overall, our AMRAN outperforms all baselines, achieving 0.657 HR@10 and 0.410 NDCG@10. It improves HR@10 by 5.3% and NDCG@10 by 3% over the best baseline (i.e., $NAIS_{concat}$).

Table 5: Performance of two submodules (CSAN and HGAN), and AMRAN.

Model	HR@10	NDCG@10
our CSAN	0.642	0.387
our HGAN	0.653	0.403
our AMRAN	**0.657**	**0.410**

5.6 RQ2: Effectiveness of our submodules

In this experiment, we are interested in measuring effectiveness of our submodules of AMRAN: CSAN and HGAN. Table 5 the experimental result. CSAN achieves 0.642 HR@10 and 0.387 HR@10, whereas HGAN achieves 0.653 HR@10 and 0.403 NDCG@10. Both of the submodules outperform all the baselines in HR@10. HGAN outperforms all the baselines, and CSAN is competitive over the baselines. This experimental result confirms that both CSAN and HGAN positively contributed to the performance of our AMRAN.

5.7 RQ3: Effectiveness of our Attention Mechanisms

We proposed two attention mechanisms: (1) spatial attention block in CSAN; and (2) graph attention mechanism in HGAN described in Section 4. In this experiment, we are interested in studying the impact of the attention mechanisms. In particular, we run each

Table 6: Performance of submodules with/without our proposed attention mechanisms.

	HR@10	NDCG@10
Without Spatial Attention Block	0.614	0.368
CSAN	0.642	0.387
Without Graph Attention Mechanism	0.638	0.389
HGAN	0.653	0.403

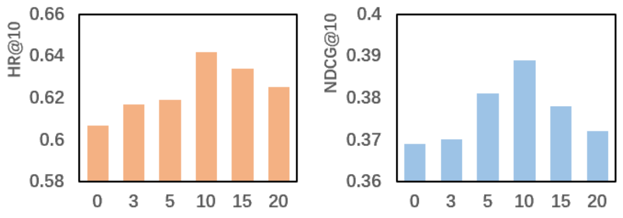


Figure 6: Performance of CSAN when varying the number of neighbors in each stream.

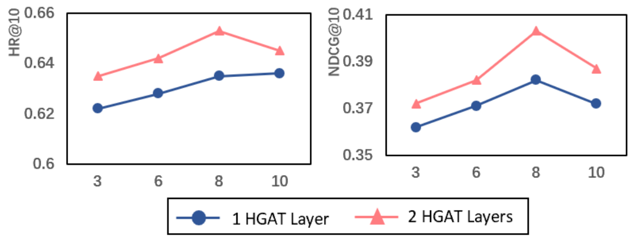


Figure 7: Performance of HGAN when varying a size of neighbor nodes at each layer (HGAL).

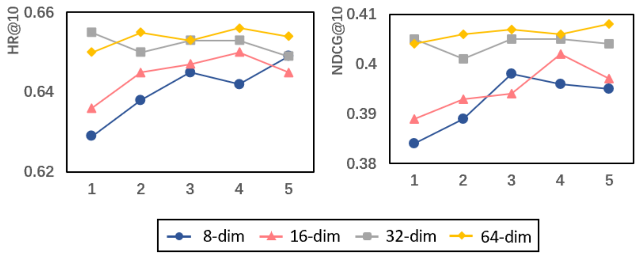


Figure 8: Performance of AMRAN when varying the number of negative samples and the size of latent semantic space (i.e., embedding size).

submodule of AMRAN (i.e., CSAN or HGAN) with/without a corresponding attention mechanism. Table 6 shows performance of these models. Our proposed attention mechanisms positively improved the performance of both submodules, confirming the positive impact toward correctly recommending fact-checking URLs.

5.8 RQ4: Hyperparameter Sensitivity

Now, we turn to analyze how our model is sensitive to hyperparameter values, and which hyperparameter value produces the best recommendation result. Recall that we utilize the context information to generate comprehensive embedding of given user and URL.

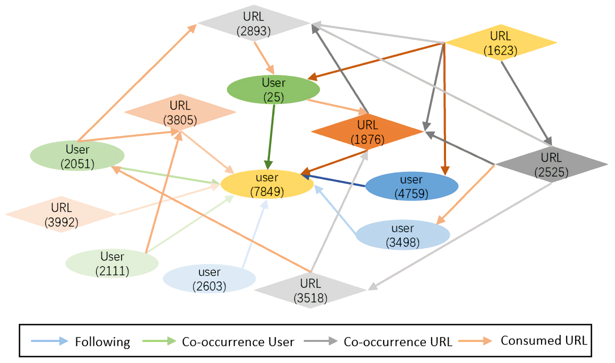


Figure 9: Visualization of relevance propagation of a user 7849. Objects in yellow denote target user and target URL (best viewed in color).

In CSAN, we employ four streams to capture fine-grained context characteristics and share the embedding weight matrix with the target user and target URL representations. In the first experiment, we vary the number of neighbors associated with each steam in CSAN to show how CSAN's performance is changed. Figure 6 shows that both *HR*@10 and *NDCG*@10 have similar trends, and selecting 10 neighbors at each stream produced the best result.

Next, we measure how performance of HGAN is changed when varying the number of HGALs and a size of selected neighbor nodes at each layer. Figure 7 demonstrates the necessity of employing 2 HGALs, which consistently outperforms the one HGAL. The best performance was achieved when a size of selected neighbor nodes was set to 8. In addition, we vary the number of negative samples, and a size of latent semantic space for the target user and target URL (i.e., an embedding vector size of the target user and target URL). Figure 8 shows high dimensional latent semantic space produces high performance of AMRAN. 64 dimensional embeddings produced the best results. We also observe that one negative sample would not be enough to produce good results in especially when an embedding vector size is small. The top performance is achieved when one positive instance paired with 3 or 4 negative instances.

5.9 Case Study: Visualization of Relevance Propagation

Attention mechanism not only improve recommendation performance of our model, but also provide explainability of our model. As a case study, we specifically chose an example to demonstrate relevance propagation. In particular, we randomly sampled a user 7849 as the example as shown in Figure 9. The user 7849 has 3 co-occurrenced users, 3 following users, and posted 4 URLs. Note that we omit less important 2nd-degree neighbors for simplicity. The most relevant neighbors and the propagation paths are highlighted automatically via the attention mechanism. In general, based on the user's historical context URLs, we observe that the topic that user 7849 would like to participate in debunking is fauxtography. However, in this very particular case, the most influential context neighbors of the user are user 25 (co-occurrence user) and user 4759 (social context) given URL 1623. Both of the context neighbors share the similar taste with user 7849 on the favorite website (Politifact.com). Moreover, we found that URL 2525 appeared in

2nd-degree neighborhood of the user 7849, and was originated from the same website (Snopes.com) with URL 1623.

6 CONCLUSION

In this paper, we proposed a novel framework, which effectively recommends relevant fact-checking URLs to *fact-checkers*. The proposed framework inspired by recent advancements in graph neural network and attention mechanism leveraged user-URL specific context information to capture deep semantic and complex structure between target user and target URL. We compared the performance of our model, AMRAN, with eight state-of-the-art baselines. Experimental results showed that our model achieved up to 5.3% improvement against the best baseline. Both submodules of AMRAN positively contributed to the recommendation results.

ACKNOWLEDGMENTS

This work was supported in part by NSF grant CNS-1755536, AWS Cloud Credits for Research, and Google Cloud. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of economic perspectives* 31, 2 (2017), 211–36.
[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2015).
[3] Gisel Bastidas Guacho, Sara Abdali, Neil Shah, and Evangelos Papalexakis. 2018. Semi-supervised Content-Based Detection of Misinformation via Tensor Embeddings. In *ASONAM*.
[4] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *WWW*.
[5] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*.
[6] Mukund Deshpande and George Karypis. 2004. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* 22, 1 (Jan. 2004), 143–177.
[7] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *SIGIR*.
[8] Ullrich KH Ecker, Stephan Lewandowsky, and David TW Tang. 2010. Explicit warnings reduce but do not eliminate the continued influence of misinformation. *Memory & cognition* 38, 8 (2010), 1087–1100.
[9] Pavlos S Efraimidis and Paul G Spirakis. 2006. Weighted random sampling with a reservoir. *Inform. Process. Lett.* 97, 5 (2006), 181–185.
[10] Usama M Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy. 1996. Advances in knowledge discovery and data mining. (1996).
[11] Álvaro Figueira and Luciana Oliveira. 2017. The current state of fake news: challenges and opportunities. *Procedia Computer Science* (2017).
[12] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. 2018. Large-Scale Learnable Graph Convolutional Networks. In *KDD*.
[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*.
[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *CVPR* (2016).
[16] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*.
[17] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*.
[18] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 30 (2018), 2354–2366.
[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*.
[20] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-Excitation Networks. *CVPR*.
[21] Bente Kalsnes. 2018. Fake News. https://oxfordre.com/communication/view/10.1093/acrefore/9780190228613.001.0001/acrefore-9780190228613-e-809
[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[23] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016).
[24] Shimon Kogan, Tobias J Moskowitz, and Marina Niessner. 2017. Fake News in Financial Markets. *Working Paper* (2017).
[25] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37.
[26] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *NIPS*.
[27] Wei Li, Xiatian Zhu, and Shaogang Gong. 2018. Harmonious attention network for person re-identification. In *CVPR*.
[28] Yang Liu and Yi-Fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *AAAI*.
[29] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.
[30] Brendan Nyhan and Jason Reifler. 2010. When corrections fail: The persistence of political misperceptions. *Political Behavior* 32, 2 (2010), 303–330.
[31] Walter Quattrociocchi, Antonio Scala, and Cass R Sunstein. 2016. Echo chambers on Facebook. *Available at SSRN 2795110* (2016).
[32] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *WWW*.
[33] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *ESWC*.
[34] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *RecSys*.
[35] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter* 19, 1 (2017), 22–36.
[36] Kai Shu, Suhang Wang, and Huan Liu. 2019. Beyond News Contents: The Role of Social Context for Fake News Detection. In *WSDM*.
[37] Craig Silverman. 2016. This analysis shows how viral fake election news stories outperformed real news on Facebook. *BuzzFeed News* (2016).
[38] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.
[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *CVPR*.
[40] Sebastian Tschiatschek, Adish Singla, Manuel Gomez Rodriguez, Arpit Merchant, and Andreas Krause. 2018. Fake News Detection in Social Networks via Crowd Signals. In *WWW*.
[41] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *NIPS*.
[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.
[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[44] Nguyen Vo and Kyumin Lee. 2018. The Rise of Guardians: Fact-checking URL Recommendation to Combat Fake News. In *SIGIR*.
[45] Nguyen Vo and Kyumin Lee. 2019. Learning from Fact-checkers: Analysis and Generation of Fact-checking Language *(SIGIR)*.
[46] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. 2017. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *WWW*.
[47] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*.
[48] Xuejian Wang, Lantao Yu, Kan Ren, Guanyu Tao, Weinan Zhang, Yong Yu, and Jun Wang. 2017. Dynamic attention deep model for article recommendation by learning human editors' demonstration. In *KDD*.
[49] Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection. In *KDD*.
[50] Liang Wu and Huan Liu. 2018. Tracing fake-news footprints: Characterizing social media messages by how they propagate. In *WSDM*.
[51] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *WSDM*.
[52] Xinyi Zhou, Reza Zafarani, Kai Shu, and Huan Liu. 2019. Fake News: Fundamental Theories, Detection Strategies and Challenges. In *WSDM*.
[53] Yousong Zhu, Chaoyang Zhao, Jinqiao Wang, Xu Zhao, Yi Wu, and Hanqing Lu. 2017. Couplenet: Coupling global structure with local parts for object detection. In *ICCV*.