About *SIGKDD Explorations*

Explorations is published twice yearly, in June/July and December/January each year. After the first two volumes, frequency may increase to quarterly. The newsletter is distributed in hardcopy form to all members of the ACM SIGKDD. It is also sent to ACM's network of libraries. Additionally, issues are published on the web and are free to the general public (http://www.acm.org/sigkdd/explorations/).

Our goal is to make *SIGKDD Explorations* an informative, rapid means of publication and a dynamic forum for communication with the Knowledge Discovery and Data Mining community. SIGKDD membership is growing at a very fast pace, and with KDD being a multi-disciplinary field, we hope that *Explorations* will facilitate its fusion and enhance the sense of community. Submissions will be reviewed by the editor and/or associate and guest editors as appropriate. We are particularly interested in short research and survey articles on various aspects of data mining and KDD. *Explorations* is also a forum for publishing position papers, controversial positions, challenges to the community, product reviews, book reviews, news items and other items of interest to the field. Please see:

http://www.acm.org/sigkdd/explorations/instructions.htm

Advertiser Information:

Explorations accepts advertisements related to data mining and KDD, including company, book, vendor, and service advertisements. For rates and instructions on submitting an ad, please see:

http://www.acm.org/sigkdd/explorations/instructions.htm#advertise

Notice to Contributing Authors of SIGKDD Explorations:

By submitting your article for distribution in this Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library
- to allow users to copy and distribute the article for noncommercial, educational or research purposes.

However, as a contributing author, you retain the copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.

Notice to Past Authors of ACM-Published Articles:

ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written work that was previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform you respective editors and permissions@acm.org, stating the title of the work, the author(s), and where and when published.

Collaborative Filtering for Binary, Positive-only Data

Koen Verstrepen*
Froomle
Antwerp, Belgium
koen.verstrepen@froomle.com

Kanishka Bhaduri†
Apple, Inc.
Cupertino, USA
kanishka.bh@gmail.com

Boris Cule
University of Antwerp
Antwerp, Belgium
boris.cule@uantwerp.be

Bart Goethals
Froomle, University of Antwerp
Antwerp, Belgium
bart.goethals@uantwerp.be

ABSTRACT

Traditional collaborative filtering assumes the availability of explicit ratings of users for items. However, in many cases these ratings are not available and only binary, positive-only data is available. Binary, positive-only data is typically associated with implicit feedback such as items bought, videos watched, ads clicked on, etc. However, it can also be the results of explicit feedback such as likes on social networking sites. Because binary, positive-only data contains no negative information, it needs to be treated differently than rating data. As a result of the growing relevance of this problem setting, the number of publications in this field increases rapidly. In this survey, we provide an overview of the existing work from an innovative perspective that allows us to emphasize surprising commonalities and key differences.

1. INTRODUCTION

Increasingly, people are overwhelmed by an abundance of choice. Via the World Wide Web, everybody has access to a wide variety of news, opinions, (encyclopedic) information, social contacts, books, music, videos, pictures, products, jobs, houses, and many other *items*, from all over the world. However, from the perspective of a particular person, the vast majority of items is irrelevant; and the few relevant items are difficult to find because they are buried under a large pile or irrelevant ones. There exist, for example, lots of books that one would enjoy reading, if only one could identify them. Moreover, not only do people fail to find relevant existing items, niche items fail to be created because it is anticipated that the target audience will not be able to find them under the pile of irrelevant items. Certain books, for example, are never written because writers anticipate they will not be able to reach a sufficiently large portion of their target audience, although the audience exists. *Recommender systems* contribute to overcome these difficulties by *connecting* individuals with items relevant to them. A good book recommender system, for example, would typically recommend 3, previously unknown, books that the user would enjoy reading, and that are sufficiently different from each other. Studying recommender systems specifically, and the connection between individuals and relevant items in general, is the subject of *recommendation* research. But the relevance of recommendation research goes beyond connecting users with items. Recommender systems can, for example, also connect genes with diseases, biological targets with drug compounds, words with documents, tags with photos, etc.

1.1 Collaborative Filtering

Collaborative filtering is a principal problem in recommendation research. In the most abstract sense, collaborative filtering is the problem of weighting missing edges in a bipartite graph.

The concrete version of this problem that got most attention until recently is *rating prediction*. In rating prediction, one set of nodes in the bipartite graph represent *users*, the other set of nodes represent *items*, an edge with weight r between user u and item i expresses that u has given i the rating r, and the task is to predict the missing ratings. Since rating prediction is a mature domain, multiple overviews exist [23; 47; 1; 59]. Recently, the attention for rating prediction diminished because of multiple reasons. First, collecting rating data is relatively expensive in the sense that it requires a non-negligible effort from the users. Second, user ratings do not correlate as well with user behavior as one would expect. Users tend to give high ratings to items they think they should consume, for example a famous book by Dostoyevsky. However, they would rather read Superman comic books, which they rate much lower. Finally, in many applications, predicting ratings is not the final goal, and the predicted ratings are only used to find the most relevant items for every user. Consequently, high ratings need to be accurate whereas the exact value of low ratings is irrelevant. However, in rating prediction high and low ratings are equally important.

Today, attention is increasingly shifting towards collaborative filtering with *binary, positive-only data*. In this version, edges are unweighted, an edge between user u and item i expresses that user u has given positive feedback about item i, and the task is to attach to every missing edge between a user u and an item i a score that indicates the suitability of recommending i to u. Binary, positive-only data is typically associated with implicit feedback such as items bought, videos watched, songs listened to, books lent from

*This work was done while Koen Verstrepen was working at the University of Antwerp.

†This work was done while Kanishka Bhaduri was working at Netflix, Inc.

a library, ads clicked on, etc. However, it can also be the result of explicit feedback, such as *likes* on social networking sites. As a result of the growing relevance of this problem setting, the number of publications in this field increases rapidly. In this survey, we provide an overview of the existing work on collaborative filtering with binary, positive-only data from an innovative perspective that allows us to emphasize surprising commonalities and key differences. To enhance the readability, we sometimes omit the specification 'binary, positive-only' and use the abbreviated term 'collaborative filtering'.

Besides the bipartite graph, five types of extra information can be available. First, there can be item content or item metadata. In the case of books, for example, the content is the full text of the book and the metadata can include the writer, the publisher, the year it was published etc. Methods that exclusively use this kind of information are typically classified as *content based*. Methods that combine this kind of information with a collaborative filtering method are typically classified as *hybrid*. Second, there can be user metadata such as gender, age, location, etc. Third, users can be connected with each other in an extra, unipartite graph. A typical example is a social network between the users. An analogous graph can exist for the items. Finally, there can be contextual information such as location, date, time, intent, company, device, etc. Exploiting information besides the bipartite graph, is out of the scope of this survey. Comprehensive discussions on exploiting information outside the user-item matrix have been presented [53; 72].

1.2 Relation to Other Domains

To emphasize the unique aspects of collaborative filtering, we highlight the commonalities and differences with two related data science problems: classification and association rule mining.

First, collaborative filtering is equivalent to jointly solving many one-class classification problems, in which every one-class classification problem corresponds to one of the items. In the classification problem that corresponds to item i, i serves as the class, all other items serve as the features, the users that have i as a known preference serve as labeled examples and the other users serve as unlabeled examples. Amazon.com, for example, has more than 200 million items in its catalog, hence solving the collaborative filtering problem for Amazon.com is equivalent to jointly solving more than 200 million one-class classification problems, which obviously requires a distinct approach. However, collaborative filtering is more than efficiently solving many one-class classification problems. Because they are tightly linked, *jointly* solving all classification problems allows for sharing information between them. The individual classification problems share most of their features; and while i serves as the class in one of the classification problems, it serves as a feature in all other classification problems.

Second, association rule mining also assumes bipartite, unweighted data and can therefore be applied to datasets used for collaborative filtering. Furthermore, recommending item i to user u can be considered as the application of the association rule $I(u) \rightarrow i$, with $I(u)$ the itemset containing the known preferences of u. However, the goals of association rule mining and collaborative filtering are different. If a rule $I(u) \rightarrow i$ is crucial for recommending i to u, but irrelevant on the rest of the data, giving the rule a high score is desirable for collaborative filtering, but typically not for association rule mining.

1.3 Outline

After the Preliminaries (Sec. 2), we introduce our framework (Sec. 3) and review the state of the art along the three dimensions of our framework: Factorization Models (Sec. 4), Deviation Functions (Sec. 5), and Minimization Algorithms (Sec. 6). Finally, we discuss the usability of methods for rating prediction (Sec. 7) and conclude (Sec. 8).

2. PRELIMINARIES

We introduced collaborative filtering as the problem of weighting missing edges in a bipartite graph. Typically, however, this bipartite graph is represented by its adjacency matrix, which is called the preference matrix.

Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ a set of items. We are given a preference matrix with training data $\mathbf{R} \in \{0,1\}^{|\mathcal{U}|\times|\mathcal{I}|}$. $\mathbf{R}_{ui} = 1$ indicates that there is a known preference of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$. $\mathbf{R}_{ui} = 0$ indicates that there is no such information. Notice that the absence of information means that either there exists no preference or there exists a preference but it is not known.

Collaborative filtering methods compute for every user-item pair (u, i) a recommendation score $s(u, i)$ that indicates the suitability of recommending i to u. Typically, the user-item-pairs are (partially) sorted by their recommendation scores. We define the matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ as $\mathbf{S}_{ui} = s(u, i)$. Furthermore, $c(x)$ gives the count of x, meaning

$$c(x) = \begin{cases} \sum_{i\in\mathcal{I}} R_{xi} & \text{if } x \in \mathcal{U} \\ \sum_{u\in\mathcal{U}} R_{ux} & \text{if } x \in \mathcal{I}. \end{cases}$$

Although we conveniently call the elements of $\mathcal{U}$ *users* and the elements of $\mathcal{I}$ *items*, these sets can contain any type of object. In the case of online social networks, for example, both sets contain the people that participate in the social network, i.e., $\mathcal{U} = \mathcal{I}$, and $\mathbf{R}_{ui} = 1$ if there exists a friendship link between persons u and i. In image tagging/annotation problems, $\mathcal{U}$ contains images, $\mathcal{I}$ contains words, and $\mathbf{R}_{ui} = 1$ if image u was tagged with word i. In chemogenomics, an early stage in the drug discovery process, $\mathcal{U}$ contains active drug compounds, $\mathcal{I}$ contains biological targets, and $\mathbf{R}_{ui} = 1$ if there is a strong interaction between compound u and biological target i.

Typically, datasets for collaborative filtering are extremely sparse, which makes it a challenging problem. The sparsity $\mathcal{S}$, computed as

$$\mathcal{S} = 1 - \frac{\sum_{(u,i)\in\mathcal{U}\times\mathcal{I}} \mathbf{R}_{ui}}{|\mathcal{U}| \cdot |\mathcal{I}|}, \tag{1}$$

typically ranges from 0.98 to 0.999 and is visualized in Figure 1. This means that a score must be computed for approximately 99% of the (u, i)-pairs based on only 1% of the (u, i)-pairs. This is undeniably a challenging task.

3. FRAMEWORK

In the most general sense, every method for collaborative filtering is defined as a function $\mathcal{F}$ that computes the recommendation scores $\mathbf{S}$ based on the data $\mathbf{R}$: $\mathbf{S} = \mathcal{F}(\mathbf{R})$.

Since different methods $\mathcal{F}$ originate from different intuitions about the problem, they are typically explained from very

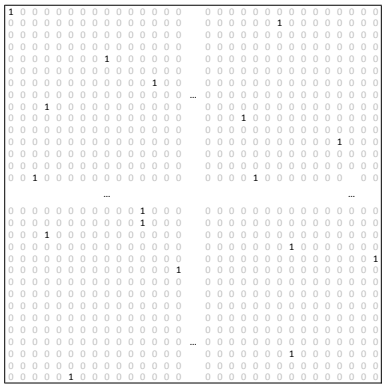

Figure 1: Typical sparsity of training data matrix $\mathbf{R}$ for binary, positive-only collaborative filtering.

different perspectives. In the literature on recommender systems in general and collaborative filtering specifically, two dominant perspectives have emerged: the model based perspective and the memory-based perspective. Unfortunately, these two are often described as two fundamentally separate classes of methods, instead of merely two perspectives on the same class of methods [47; 23]. As a result, the comparison between methods often remains superficial.

We, however, will explain many different collaborative filtering methods $\mathcal{F}$ from one and the same perspective. As such we facilitate the comparison and classification of these methods. Our perspective is a matrix factorization framework in which every method $\mathcal{F}$ consists of three fundamental building blocks: a factorization model of the recommendation scores $\mathbf{S}$, a deviation function that measures the deviation between the data $\mathbf{R}$ and the recommendation scores $\mathbf{S}$, and a minimization procedure that tries to find the model parameters that minimize the deviation function.

First, the **factorization model** computes the matrix of recommendation scores $\mathbf{S}$ as a link function l of a sum of T terms in which a term t is the product of F_t factor matrices:

$$\mathbf{S} = l\left(\sum_{t=1}^{T}\prod_{f=1}^{F_t}\mathbf{S}^{(t,f)}\right). \tag{2}$$

For many methods, l is the identity function, $T = 1$ and $F_1 = 2$. In this case, the factorization is given by:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}. \tag{3}$$

Because of their dimensions, $\mathbf{S}^{(1,1)} \in \mathbb{R}^{|\mathcal{U}|\times D}$ and $\mathbf{S}^{(1,2)} \in \mathbb{R}^{D\times|\mathcal{I}|}$ are often called the user-factor matrix and item-factor matrix, respectively. Figure 2 visualizes Equation 3. More complex models are often more realistic, but generally contain more parameters which increases the risk of overfitting.

Second, the number of terms T, the number of factor matrices F_t and the dimensions of the factor matrices are an integral part of the model, independent of the data $\mathbf{R}$[1]. Every entry in the factor matrices however, is a parameter that needs to be computed based on the data $\mathbf{R}$. We collectively denote these parameters as θ. Whenever we want

[1]High level statistics of the data might be taken into consideration to choose the model. There is however no clear, direct dependence on the data.

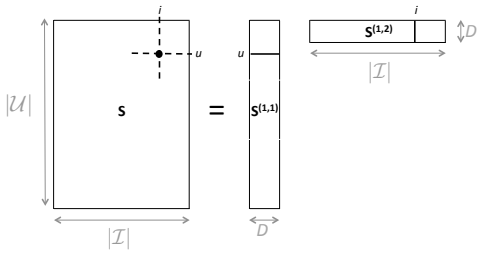

Figure 2: Matrix Factorization with 2 factor matrices (Eq. 3).

to emphasize that the matrix of recommendation scores $\mathbf{S}$ is dependent on these parameters, we will write it as $\mathbf{S}(\theta)$. The model in Figure 2, for example, contains $(|\mathcal{U}| + |\mathcal{I}|) \cdot D$ parameters. Computing all the parameters in a factorization model is done by minimizing the *deviation* between the data $\mathbf{R}$ and the parameters θ. This deviation is measured by the **deviation function** $\mathcal{D}(\theta, \mathbf{R})$. Formally we compute the *optimal* values θ^* of the parameters θ as

$$\theta^* = \arg\min_{\theta} \mathcal{D}(\theta, \mathbf{R}).$$

Many deviation functions exist, and every deviation function mathematically expresses a different interpretation of the concept *deviation*.

Third, efficiently computing the parameters that minimize a deviation function is often non trivial because the majority of deviation functions is non-convex in the parameters of the factorization model. In that case, minimization algorithms can only compute parameters that correspond to a local minimum. The initialization of the parameters in the factorization model and the chosen hyperparameters of the minimization algorithm determine which local minimum will be found. If the value of the deviation function in this local minimum is not much higher than that of the global minimum, it is considered a good minimum. An intuitively appealing deviation function is worthless if there exists no algorithm that can efficiently compute parameters that correspond to a good local minimum of this deviation function.

Finally, we assume a basic usage scenario in which model parameters are recomputed periodically (typically, a few times a day). Computing the recommendation scores for a user based on the model, and extracting the items corresponding to the top-N scores, is assumed to be performed in *real time*. Specific aspects of scenarios in which models need to be recomputed in real time, are out of the scope of this survey.

In this overview, we first survey existing models in Section 4. Next, we compare the existing deviation functions in Section 5. Afterwards, we discuss the minimization algorithms that are used for fitting the model parameters to the deviation functions in Section 6. Finally, we discuss the applicability of rating-based methods in Section 7 and conclude in Section 8.

4. FACTORIZATION MODELS

Equation 2 gives a general description of all models for collaborative filtering. In this section, we discuss how the specific collaborative filtering models map to this equation.

4.1 Basic Models

A statistically well founded method is probabilistic latent semantic analysis (pLSA) by Hofmann, which is centered around the so called aspect model [19]. Hofmann models the probability $p(i|u)$ that a user u will prefer an item i as the mixture of D probability distributions induced by the hidden variables d:

$$p(i|u) = \sum_{d=1}^{D} p(i,d|u).$$

Furthermore, by assuming u and i conditionally independent given d, he obtains:

$$p(i|u) = \sum_{d=1}^{D} p(i|d) \cdot p(d|u).$$

This model corresponds to a basic two-factor matrix factorization:

$$\begin{aligned} \mathbf{S} &= \mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)} \\ \mathbf{S}_{ui} &= p(i|u) \\ \mathbf{S}^{(1,1)}_{ud} &= p(d|u) \\ \mathbf{S}^{(1,2)}_{di} &= p(i|d), \end{aligned} \tag{4}$$

in which the $|\mathcal{U}| \times D$ parameters in $\mathbf{S}^{(1,1)}_{ud}$ and the $D \times |\mathcal{I}|$ parameters in $\mathbf{S}^{(1,2)}_{di}$ are a priori unknown and need to be computed based on the data $\mathbf{R}$. An appealing property of this model is the probabilistic interpretation of both the parameters and the recommendation scores. Fully in line with the probabilistic foundation, the parameters are constrained as:

$$\begin{aligned} \mathbf{S}^{(1,1)}_{ud} &\geq 0 \\ \mathbf{S}^{(1,2)}_{di} &\geq 0 \\ \sum_{i \in \mathcal{I}} p(i|d) &= 1 \\ \sum_{d=1}^{D} p(d|u) &= 1, \end{aligned} \tag{5}$$

expressing that both factor matrices are non-negative and that all row sums of $\mathbf{S}^{(1,1)}$ and $\mathbf{S}^{(1,2)}$ must be equal to 1 since they represent probabilities.

Latent dirichlet allocation (LDA) is a more rigorous statistical model, which puts Dirichlet priors on the parameters $p(d|u)$ [6; 19]. However, for collaborative filtering these priors are integrated out and the resulting model for computing recommendation scores is again a simple two factor factorization model.

The aspect model also has a geometric interpretation. In the training data $\mathbf{R}$, every user is profiled as a binary vector in an $|\mathcal{I}|$-dimensional space in which every dimension corresponds to an item. Analogously, every item is profiled as a binary vector in a $|\mathcal{U}|$-dimensional space in which every dimension corresponds to a user. Now, in the factorized representation, every hidden variable d represents a dimension of a D-dimensional space. Therefore, the matrix factorization $\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}$ implies a transformation of both user- and item-vectors to the same D-dimensional space. A row vector $\mathbf{S}^{(1,1)}_{u\cdot} \in \mathbb{R}^{1\times D}$ is the representation of user u in this D-dimensional space and a column vector $\mathbf{S}^{(1,2)}_{\cdot i} \in \mathbb{R}^{D\times 1}$ is the representation of item i in this D-dimensional space.

Figure 2 visualizes the user-vector of user u and the item-vector of item i. Now, as a result of the factorization model, a recommendation score $\mathbf{S}_{ui}$ is computed as the dotproduct of the user-vector of u with the item-vector of i:

$$\begin{aligned} \mathbf{S}^{(1,1)}_{u\cdot} \cdot \mathbf{S}^{(1,2)}_{\cdot i} &= \sum_{d=1}^{D} \mathbf{S}^{(1,1)}_{ud}\mathbf{S}^{(1,2)}_{di} \\ &= ||\mathbf{S}^{(1,1)}_{u\cdot}|| \cdot ||\mathbf{S}^{(1,2)}_{\cdot i}|| \cdot \cos(\phi_{ui}) \\ &= ||\mathbf{S}^{(1,2)}_{\cdot i}|| \cdot \cos(\phi_{ui}), \end{aligned} \tag{6}$$

with ϕ_{ui} the angle between the two vectors, and $||\mathbf{S}^{(1,1)}_{u\cdot}|| = 1$ a probabilistic constraint of the model (Eq.5). Therefore, an item i will be recommended if its vector norm $||\mathbf{S}^{(1,2)}_{\cdot i}||$ is large and ϕ_{ui}, the angle of its vector with the user vector, is small. From this geometric interpretation we learn that the recommendation scores computed with the model in Equation 4 contain both a personalized factor $\cos(\phi_{ui})$ and a non-personalized, popularity based factor $||\mathbf{S}^{(1,2)}_{\cdot i}||$.

Many other authors adopted this two-factor model. However, they abandoned its probabilistic foundation by removing the constraints on the parameters (Eq.5) [21; 36; 37; 55; 73; 45; 52; 61; 16; 12]:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}. \tag{7}$$

Yet another interpretation of this two-factor model is that every hidden variable d represents a cluster containing both users and items. A large value $\mathbf{S}^{(1,1)}_{ud}$ means that user u has a large degree of membership in cluster d. Similarly, a large value $\mathbf{S}^{(1,2)}_{di}$ means that item i has a large degree of membership in cluster d. As such, pLSA can be interpreted as a soft clustering model. According to the interpretation, item i will be recommended to a user u if they have high degrees of membership in the same clusters.

Although much less common, hard clustering models for collaborative filtering also exist. Hofmann and Puzicha [20] proposed the model

$$p(i|u, e(u) = c, d(i) = d) = p(i)\phi(e,d), \tag{8}$$

in which $e(u)$ indicates the cluster user u belongs to and $d(i)$ indicates the cluster item i belongs to. Furthermore, $\phi(e,d)$ is the association value between the user-cluster e and the item-cluster d. This cluster association factor increases or decreases the probability that u likes i relative to the independence model $p(i|u) = p(i)$. As opposed to the aspect model, this model assumes E user-clusters only containing users and D item-clusters only containing items. Furthermore a user or item belongs to exactly one cluster. The factorization model corresponding to this approach is:

$$\begin{aligned} \mathbf{S} &= \mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}\mathbf{S}^{(1,3)}\mathbf{S}^{(1,4)}, \\ \mathbf{S}_{ui} &= p(i|u), \\ \mathbf{S}^{(1,1)}_{ue} &= \mathbb{I}(e(u) = e), \\ \mathbf{S}^{(1,2)}_{ed} &= \phi(e,d), \\ \mathbf{S}^{(1,3)}_{di} &= \mathbb{I}(d(i) = d), \\ \mathbf{S}^{(1,4)}_{ij} &= p(i) \cdot \mathbb{I}(i = j), \end{aligned}$$

in which $\mathbb{I}(\mathit{true}) = 1$, $\mathbb{I}(\mathit{false}) = 0$, and E and D are hyperparameters. The $|\mathcal{U}| \times E$ parameters in $\mathbf{S}^{(1,1)}$, $E \times D$ parameters in $\mathbf{S}^{(1,2)}$, $D \times |\mathcal{I}|$ parameters in $\mathbf{S}^{(1,3)}$ and the $|\mathcal{I}|$

parameters $\mathbf{S}^{(1,4)}$ need to be computed based on the data. Ungar and Foster [63] proposed a similar hard clustering method.

4.2 Explicitly Biased Models

In the above models, the recommendation score $\mathbf{S}_{ui}$ of an item i for a user u is the product of a personalized factor with an item-bias factor related to item-popularity. In Equation 6 the personalized factor is $\cos(\phi_{ui})$, and the bias factor is $||\mathbf{S}^{(1,2)}_{i}||$. In Equation 8 the personalized factor is $\phi(e,d)$, and the bias factor is $p(i)$. Other authors [28; 40; 24] proposed to model item- and user-biases as explicit terms instead of implicit factors. This results in the following factorization model:

$$\begin{aligned} \mathbf{S} &= \sigma\left(\mathbf{S}^{(1,1)} + \mathbf{S}^{(2,1)} + \mathbf{S}^{(3,1)}\mathbf{S}^{(3,2)}\right) \\ \mathbf{S}^{(1,1)}_{ui} &= b_u \\ \mathbf{S}^{(2,1)}_{ui} &= b_i, \end{aligned} \tag{9}$$

with σ the sigmoid link-function, and $\mathbf{S}^{(1,1)}, \mathbf{S}^{(2,1)} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ the user- and item-bias matrices in which all columns of $\mathbf{S}^{(1,1)}$ are identical and also all rows of $\mathbf{S}^{(2,1)}$ are identical. The $|\mathcal{U}|$ parameters in $\mathbf{S}^{(1,1)}$, $|\mathcal{I}|$ parameters in $\mathbf{S}^{(2,1)}$, $|\mathcal{U}|\cdot D$ parameters in $\mathbf{S}^{(3,1)}$, and $|\mathcal{I}|\cdot D$ parameters in $\mathbf{S}^{(3,2)}$ need to be computed based on the data. D is a hyperparameter of the model. The goal of explicitly modeling the bias terms is to make the interaction term $\mathbf{S}^{(3,1)}\mathbf{S}^{(3,2)}$ a pure personalization term. Although bias terms are commonly used for collaborative filtering with rating data, only a few works with collaborative filtering with binary, positive-only data use them.

4.3 Basic Neighborhood Models

Multiple authors proposed special cases of the basic two-factor factorization in Equation 7.

4.3.1 Item-based

In a first special case, $\mathbf{S}^{(1,1)} = \mathbf{R}$ [10; 54; 45; 2; 34]. In this case, the factorization model is given by

$$\mathbf{S} = \mathbf{R}\mathbf{S}^{(1,2)}. \tag{10}$$

Consequently, a user u is profiled by an $|\mathcal{I}|$-dimensional binary vector $\mathbf{R}_{u\cdot}$ and an item is profiled by an $|\mathcal{U}|$-dimensional real valued vector $\mathbf{S}^{(1,2)}_{\cdot i}$. This model is often interpreted as an *item-based neighborhood model* because the recommendation score $\mathbf{S}_{ui}$ of item i for user u is computed as

$$\mathbf{S}_{ui} = \mathbf{R}_{u\cdot}\cdot\mathbf{S}^{(1,2)}_{\cdot i} = \sum_{j\in\mathcal{I}}\mathbf{R}_{uj}\cdot\mathbf{S}^{(1,2)}_{ji} = \sum_{j\in I(u)}\mathbf{S}^{(1,2)}_{ji},$$

with $I(u)$ the known preferences of user u, and a parameter $\mathbf{S}^{(1,2)}_{ji}$ typically interpreted as the similarity between items j and i, i.e., $\mathbf{S}^{(1,2)}_{ji} = sim(j,i)$. Consequently, $\mathbf{S}^{(1,2)}$ is often called the item-similarity matrix. The SLIM method [34] adopts this model and additionally imposes the constraints

$$\mathbf{S}^{(1,2)}_{ji} \geq 0, \mathbf{S}^{(1,2)}_{ii} = 0. \tag{11}$$

The non-negativity is imposed to enhance interpretability. The zero-diagonal is imposed to avoid finding a trivial solution for the parameters in which every item is maximally similar to itself and has zero similarity with any other item.

This model is rooted in the intuition that good recommendations are similar to the known preferences of the user. Although they do not present it as such, Gori et al. [18] proposed ItemRank, a method that is based on an interesting extension of this intuition: good recommendations are similar to other good recommendations, with a bias towards the known preferences of the user. The factorization model corresponding to ItemRank is based on PageRank [35] and given by:

$$\mathbf{S} = \alpha\cdot\mathbf{S}\mathbf{S}^{(1,2)} + (1-\alpha)\cdot\mathbf{R}. \tag{12}$$

Because of the recursive nature of this model, $\mathbf{S}$ needs to be computed iteratively [18; 35].

4.3.2 User-based

Other authors proposed the symmetric counterpart of Equation 10 in which $\mathbf{S}^{(1,2)} = \mathbf{R}$ [48; 2; 3]. In this case, the factorization model is given by

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{R}, \tag{13}$$

which is often interpreted as a *user-based neighborhood model* because the recommendation score $\mathbf{S}_{ui}$ of item i for user u is computed as

$$\mathbf{S}_{ui} = \mathbf{S}^{(1,1)}_{u\cdot}\cdot\mathbf{R}_{\cdot i} = \sum_{v\in\mathcal{U}}\mathbf{S}^{(1,1)}_{uv}\cdot\mathbf{R}_{vi},$$

in which parameter $\mathbf{S}^{(1,1)}_{uv}$ is interpreted as the similarity between users u and v, i.e., $\mathbf{S}^{(1,1)}_{uv} = sim(u,v)$. $\mathbf{S}^{(1,1)}$ is often called the user-similarity matrix. The intuition behind this model is that good recommendations are preferred by similar users. Furthermore, Aiolli [2; 3] foresees the possibility to rescale the scores such that popular items get less importance. This changes the factorization model to

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{R}\mathbf{S}^{(1,3)}, \qquad \mathbf{S}^{(1,3)}_{ji} = \mathbb{I}(j=i)\cdot c(i)^{-(1-\beta)},$$

in which $\mathbf{S}^{(1,3)}$ is a diagonal rescaling matrix that rescales the item scores according to the item popularity $c(i)$, and $\beta\in[0,1]$ a hyperparameter. Additionally, Aiolli [3] imposes the constraint that $\mathbf{S}^{(1,1)}$ needs to be row normalized:

$$||\mathbf{S}^{(1,1)}_{u\cdot}|| = 1.$$

To the best of our knowledge, there exists no user-based counterpart for the ItemRank model of Equation 12.

4.3.3 Explicitly Biased

Furthermore, it is, obviously, possible to add explicit bias terms to neighborhood models. Wang et al. [68] proposed an item-based neighborhood model with one bias term:

$$\mathbf{S} = \mathbf{S}^{(1,1)} + \mathbf{R}\mathbf{S}^{(2,2)}, \qquad \mathbf{S}^{(1,1)}_{ui} = b_i,$$

and an analogous user-based model:

$$\mathbf{S} = \mathbf{S}^{(1,1)} + \mathbf{S}^{(2,1)}\mathbf{R}, \qquad \mathbf{S}^{(1,1)}_{ui} = b_i.$$

4.3.4 Unified

Moreover, Verstrepen and Goethals [66] showed that the item- and user-based neighborhood models are two incomplete instances of the same general neighborhood model. Consequently they propose *KUNN*, a complete instance of this general neighborhood model. The model they propose

can be written as a weighted sum of a user- and an item-based model, in which the weights depend on the user u and the item i for which a recommendation score is computed:

$$\begin{aligned} \mathbf{S} &= \left(\mathbf{S}^{(1,1)}\mathbf{R}\right)\mathbf{S}^{(1,3)} + \mathbf{S}^{(2,1)}\left(\mathbf{R}\mathbf{S}^{(2,3)}\right) \\ \mathbf{S}^{(1,3)}_{ij} &= \mathbb{I}(i=j)\cdot c(i)^{-1/2} \\ \mathbf{S}^{(2,1)}_{uv} &= \mathbb{I}(u=v)\cdot c(u)^{-1/2}. \end{aligned} \tag{14}$$

Finally, note that the above matrix factorization based descriptions of nearest neighbors methods imply that matrix factorization methods and neighborhood methods are not two separate approaches, but two perspectives on the same approach.

4.4 Factored Similarity Neighborhood Models

The item-similarity matrix $\mathbf{S}^{(1,2)}$ in Equation 10 contains $|\mathcal{I}|^2$ parameters, which is in practice often a very large number. Consequently, it can happen that the training data is not sufficient to accurately compute this many parameters. Furthermore, one often precomputes the item-similarity matrix $\mathbf{S}^{(1,2)}$ and performs the dotproducts $\mathbf{R}_{u\cdot}\cdot\mathbf{S}^{(1,2)}_{\cdot i}$ to compute the recommendation scores $\mathbf{S}_{ui}$ in real time. In this case, the dotproduct is between two $|\mathcal{I}|$-dimensional vectors, which is often prohibitive in real time, high traffic applications. One solution[2] is to factorize the similarity matrix, which leads to the following factorization model:

$$\mathbf{S} = \mathbf{R}\mathbf{S}^{(1,2)}{\mathbf{S}^{(1,2)}}^T,$$

in which every row of $\mathbf{S}^{(1,2)} \in \mathbb{R}^{|\mathcal{I}|\times D}$ represents a D-dimensional item-profile vector, with D a hyperparameter [71; 8]. In this case the item-similarity matrix is equal to

$$\mathbf{S}^{(1,2)}{\mathbf{S}^{(1,2)}}^T,$$

which means that the similarity $sim(i,j)$ between two items i and j is defined as the dotproduct of their respective profile vectors

$$\mathbf{S}^{(1,2)}_{i\cdot}\cdot{\mathbf{S}^{(1,2)}_{j\cdot}}^T.$$

This model only contains $|\mathcal{I}|\cdot D$ parameters instead of $|\mathcal{I}|^2$ which is much fewer since typically $D \ll |\mathcal{I}|$. Furthermore, by first precomputing the item vectors $\mathbf{S}^{(1,2)}$ and then precomputing the D-dimensional user-profile vectors given by $\mathbf{R}\mathbf{S}^{(1,2)}$, the real time computation of a score $\mathbf{S}_{ui}$ encompasses a dotproduct between a D-dimensional user-profile vector and a D-dimensional item-profile vector. Since $D \ll |\mathcal{I}|$, this dotproduct is much less expensive and can be more easily performed in real time. Furthermore, there is no trivial solution for the parameters of this model, as is the case for the non factored item-similarity model in Equation 10. Consequently, it is never required to impose the constraints from Equation 11. To avoid scaling problems when fitting parameters, Weston et al. [71] augment this model with a diagonal normalization factor matrix:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{R}\mathbf{S}^{(1,3)}{\mathbf{S}^{(1,3)}}^T \tag{15}$$

$$\mathbf{S}^{(1,1)}_{uv} = \mathbb{I}(u=v)\cdot c(u)^{-2/2}. \tag{16}$$

[2]Another solution is to enforce sparsity on the similarity matrix by means of the deviation function. This is discussed in Section 5.

A limitation of this model is that it implies that the similarity matrix is symmetric. This might hurt the model's accuracy in certain applications such as recommending tags for images. For an image of the Eiffel tower that is already tagged *Eiffel tower*, for example, the tag *Paris* is a reasonable recommendation. However, for an image of the Louvre already tagged *Paris*, the tag *Eiffel tower* is a bad recommendation. Paterek solved this problem for rating data by representing every item by two separate D-dimensional vectors [41]. One vector represents the item if it serves as evidence for computing recommendations, the other vector represents the item if it serves as a candidate recommendation. In this way, they can model also asymmetric similarities. This idea is not restricted to rating data, and for binary, positive-only data, it was adopted by Steck [58]:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{R}\mathbf{S}^{(1,3)}\mathbf{S}^{(1,4)} \tag{17}$$

$$\mathbf{S}^{(1,1)}_{uv} = \mathbb{I}(u=v)\cdot c(u)^{-1/2}. \tag{18}$$

Kabbur et al. also adopted this idea. However, similar to Equation 9, they also add bias terms:

$$\begin{aligned} \mathbf{S} &= \mathbf{S}^{(1,1)} + \mathbf{S}^{(2,1)} + \mathbf{S}^{(3,1)}\mathbf{R}\mathbf{S}^{(3,3)}\mathbf{S}^{(3,4)} \\ \mathbf{S}^{(1,1)}_{ui} &= b_u \\ \mathbf{S}^{(2,1)}_{ui} &= b_i \\ \mathbf{S}^{(3,1)}_{uv} &= \mathbb{I}(u=v)\cdot c(u)^{-\beta/2}, \end{aligned} \tag{19}$$

with $\mathbf{S}^{(3,3)} \in \mathbb{R}^{|\mathcal{I}|\times D}$ the matrix of item profiles when they serve as evidence, $\mathbf{S}^{(3,4)} \in \mathbb{R}^{D\times|\mathcal{I}|}$ the matrix of item profiles when they serve as candidates , and $\beta \in [0,1]$ and D hyperparameters.

4.5 Higher Order Neighborhood Models

The nearest neighbors methods discussed up to this point only consider pairwise interactions $sim(j,i)$ and/or $sim(u,v)$ and aggregate all the relevant ones for computing recommendations. Several authors [10; 31; 64; 50; 30; 33; 7] have proposed to incorporate also higher order interactions $sim(J,i)$ and/or $sim(u,V)$ with $J \subset \mathcal{I}$ and $V \subset \mathcal{U}$. Also in this case we can distinguish item-based approaches from user-based approaches.

4.5.1 Item-based

For the item-based approach, most authors [10; 31; 64; 30; 7] propose to replace the user-item matrix $\mathbf{R}$ in the pairwise model of Equation 10 by the user-itemset matrix $\mathbf{X}$:

$$\begin{aligned} \mathbf{S} &= \mathbf{X}\mathbf{S}^{(1,2)} \\ \mathbf{X}_{uJ} &= \prod_{j\in J}\mathbf{R}_{uj}, \end{aligned} \tag{20}$$

with $\mathbf{S}^{(1,2)} \in \mathbb{R}^{D\times|\mathcal{I}|}$ the itemset-item similarity matrix and $\mathbf{X} \in \{0,1\}^{|\mathcal{U}|\times D}$ the user-itemset matrix, in which $D \leq 2^{|\mathcal{I}|}$ is the result of an itemset selection procedure.

The HOSLIM method [7] adopts this model and additionally imposes the constraints in Equation 11.

The case in which $D = 2^{|\mathcal{I}|}$, and $\mathbf{S}^{(1,2)}$ is dense, is intractable. Tractable methods either limit $D \ll 2^{|\mathcal{I}|}$ or impose sparsity on $\mathbf{S}^{(1,2)}$ via the deviation function. While we discuss the latter in Section 5.11, there are multiple ways to do the former. Deshpande and Karypis [10] limit the number of itemsets by limiting the size of J. Alternatively, Christakopoulou and Karypis [7] only consider itemsets J that

were preferred by more than a minimal number of users. van Leeuwen and Puspitaningrum, on the other hand, limit the number of higher order itemsets by using an itemset selection algorithm based on the minimal description length principle [64]. Finally, Menezes et al. claim that it is in certain applications possible to compute all higher order interactions if one computes all higher order interactions on demand instead of in advance [31]. However, delaying the computation does not reduce its exponential complexity. Only if a large portion of the users requires recommendations on a very infrequent basis, computations for these users can be spread over a very long period of time and their approach might be feasible.

An alternative model for incorporating higher order interactions between items consists of finding the best association rule for making recommendations [50; 33]. This corresponds to the matrix factorization model

$$\mathbf{S} = \mathbf{X} \otimes \mathbf{S}^{(1,2)}, \qquad \mathbf{X}_{uJ} = \prod_{j \in J} \mathbf{R}_{uj},$$

with

$$(\mathbf{A} \otimes \mathbf{B})_{xy} = \max_{i=1\ldots m} \mathbf{A}_{xi}\mathbf{B}_{iy},$$

in which $\mathbf{A} \in \mathbb{R}^{n \times m}$ and $\mathbf{B} \in \mathbb{R}^{m \times k}$.

We did not find a convincing motivation for either of the two aggregation strategies. Moreover, multiple authors report that their attempt to incorporate higher order interactions heavily increased computational costs and did not significantly improve the results [10; 64].

4.5.2 User-based

Incorporating higher order interactions between users can be achieved be replacing the user-item matrix in Equation 13 by the userset-item matrix $\mathbf{Y}$ [30; 60]:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{Y}, \qquad \mathbf{Y}_{Vi} = \prod_{v \in V} \mathbf{R}_{vi}, \tag{21}$$

with $\mathbf{Y} \in \{0,1\}^{D \times |\mathcal{I}|}$ and, $\mathbf{S}^{(1,1)} \in \mathbb{R}^{|\mathcal{U}| \times D}$ the user-userset similarity matrix, in which $D \leq 2^{|\mathcal{U}|}$ is the result of a userset selection procedure [30; 60].

4.6 Multi-profile Models

When multiple users, e.g., members of the same family, share a single account, or when a user has multiple distinct tastes, the above matrix factorization models can be too limited because they aggregate all the distinct tastes of an account into one vector [67; 70; 26]. Therefore, Weston et al. [70] propose MaxMF, in which they model every account with multiple vectors instead of just one. Then, for every candidate recommendation, their model chooses the vector that maximizes the score of the candidate:

$$\mathbf{S}_{ui} = \max_{p=1\ldots P} \left(\mathbf{S}^{(1,1,p)}\mathbf{S}^{(1,2)}\right)_{ui}.$$

Kabbur and Karypis [26] argue that this approach worsens the performance for accounts with homogeneous taste or a low number of known preferences and therefore propose, NLMFi, an adapted version that combines a global account profile with multiple taste-specific account profiles:

$$\mathbf{S}_{ui} = \left(\mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}\right)_{ui} + \max_{p=1\ldots P} \left(\mathbf{S}^{(2,1,p)}\mathbf{S}^{(2,2)}\right)_{ui}.$$

Alternatively, they also propose NLMFs, a version in which $\mathbf{S}^{(2,2)} = \mathbf{S}^{(1,2)}$, i.e., the item profiles are shared between the global and the taste-specific terms:

$$\mathbf{S} = \left(\mathbf{S}^{(1,1)}\mathbf{S}^{(1,2)}\right)_{ui} + \max_{p=1\ldots P} \left(\mathbf{S}^{(2,1,p)}\mathbf{S}^{(1,2)}\right)_{ui}.$$

An important downside of the above two models is that P, the number of distinct account-profiles, is a hyperparameter that is the same for every account and cannot be too large (typically 2 or 3) to avoid an explosion of the computational cost and the number of parameters. DAMIB-Cover [67], on the other hand, starts from the item-based model in Equation 10, and efficiently considers $P = 2^{c(u)}$ different profiles for every account u. Specifically, every profile corresponds to a subset $S^{(p)}$ of the known preferences of u, $I(u)$. This results in the factorization model

$$\mathbf{S}_{ui} = \max_{p=1\ldots 2^{c(u)}} \left(\mathbf{R}\mathbf{S}^{(1,2,p)}\mathbf{S}^{(1,3)}\right)_{ui}$$

$$\mathbf{S}^{(1,2,p)}_{jk} = \frac{\mathbb{I}(j=k) \cdot |S^{(p)} \cap \{j\}|}{|S^{(p)}|^{\beta}},$$

with $\mathbf{S}^{(1,2,p)}$ a diagonal matrix that selects and rescales the known preferences of u that correspond to the subset $S^{(p)} \subseteq I(u)$, and $\beta \in [0,1]$ a hyperparameter.

5. DEVIATION FUNCTIONS

The factorization models described in Sec. 4 contain many parameters, i.e., the entries in the a priori unknown factor matrices, which we collectively denote as θ. These parameters need to be computed based on the training data $\mathbf{R}$. Computing all the parameters in a factorization model is done by minimizing the *deviation* between the training data $\mathbf{R}$ and the parameters θ of the factorization model. This deviation is measured by a deviation function $\mathcal{D}(\theta, \mathbf{R})$. Many deviation functions exist, and every deviation function mathematically expresses a different interpretation of the concept *deviation*.

The majority of deviation functions is non-convex in the parameters θ. Consequently, minimization algorithms can only compute parameters that correspond to a local minimum. The initialization of the parameters in the factorization model and the chosen hyperparameters of the minimization algorithm determine which local minimum will be found. If the value of the deviation function in this local minimum is not much higher than that of the global minimum, it is considered a good minimum.

5.1 Probabilistic Scores-based

Hofmann [19] proposes to compute the *optimal* parameters θ^* of the model as those that maximize the loglikelihood of the model, i.e., $\log p(\mathbf{R}|\theta)$, the logprobability of the known preferences given the model:

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \log p(\mathbf{R}_{ui}|\theta). \tag{22}$$

Furthermore, he models $p(\mathbf{R}_{ui} = 1|\theta)$ with $\mathbf{S}_{ui}$, i.e., he interprets the scores $\mathbf{S}$ as probabilities. This gives

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \log \mathbf{S}_{ui},$$

which is equivalent to minimizing the deviation function

$$\mathcal{D}(\theta, \mathbf{R}) = -\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \log \mathbf{S}_{ui}. \quad (23)$$

Recall that parameters θ are not directly visible in the right hand side of this equation but that scores $\mathbf{S}_{ui}$ are computed based on the factorization model which contains parameters θ, i.e., we use short notation $\mathbf{S}_{ui}$ instead of the full notation $\mathbf{S}_{ui}(\theta)$.

This deviation function was also adopted by Blei et al. in their approach called *latent dirichlet allocation (LDA)* [6].

Furthermore, note that Hofmann only maximizes the log-probability of the observed feedback and ignores the missing preferences. This is equivalent to the assumption that there is no information in the missing preferences, which implicitly corresponds to the assumption that feedback is *missing at random (MAR)* [56]. Clearly, this is not a realistic assumption, since negative feedback is missing by definition, which is obviously non random. Moreover, the number of items in collaborative filtering problems is typically very large, and only a very small subset of them will be preferred by a user. Consequently, the probability that a missing preference is actually not preferred is high. Hence, in reality, the feedback is *missing not at random (MNAR)*, and a good deviation function needs to account for this [56]. Furthermore, notice that Hofmann only maximizes the logprobability of the observed feedback and ignores the missing preferences. This is equivalent to the assumption that there is no information in the missing preferences, which implicitly corresponds to the assumption that feedback is *missing at random (MAR)* [56]. Clearly, this is not a realistic assumption, since negative feedback is missing by definition, which is obviously non random. Moreover, the number of items in collaborative filtering problems is typically very large, and only a very small subset of them will be preferred by a user. Consequently, the probability that a missing preference is actually not preferred is high. Hence, in reality, the feedback is *missing not at random (MNAR)*, and a good deviation function needs to account for this [56].

One approach is to assume, for the purposes of defining the deviation function, that *all* missing preferences are not preferred. This assumption is called *all missing are negative (AMAN)* [37]. Under this assumption, the parameters that maximize the loglikelihood of the model are computed as

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \log p(\mathbf{R}_{ui}|\theta).$$

For binary, positive-only data, one can model $p(\mathbf{R}_{ui}|\theta)$ as $\mathbf{S}_{ui}^{\mathbf{R}_{ui}} \cdot (1 - \mathbf{S}_{ui})^{(1-\mathbf{R}_{ui})}$. In this case, the parameters are computed as

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \left(\mathbf{R}_{ui} \log \mathbf{S}_{ui} + (1 - \mathbf{R}_{ui}) \log(1 - \mathbf{S}_{ui})\right).$$

While the AMAN assumption is more realistic than the MAR assumption, it adopts a conceptually flawed missing data model. Specifically, it assumes that all missing preferences are not preferred, which contradicts the goal of collaborative filtering: to find the missing preferences that are actually preferred. A better missing data model still assumes that all missing preferences are not preferred. However, it attaches a lower confidence to the assumption that a missing preference is not preferred, and a higher confidence to the assumption that an observed preference is indeed preferred. One possible way to apply this missing data model was proposed by Steck [56]. Although his original approach is more general, we give a specific simplified version that for binary, positive-only data:

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \big(\mathbf{R}_{ui} \log \mathbf{S}_{ui} + \alpha \cdot (1 - \mathbf{R}_{ui}) \log (1 - \mathbf{S}_{ui})\big), \quad (24)$$

in which the hyperparameter $\alpha < 1$ attaches a lower importance to the contributions that correspond to $\mathbf{R}_{ui} = 0$. Johnson [24] proposed a very similar computation, but does not motivate why he deviates from the theoretically well founded version of Steck. Furthermore, Steck adds regularization terms to avoid overfitting and finally proposes the deviation function

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \big(-\mathbf{R}_{ui} \log \mathbf{S}_{ui} - \alpha \cdot (1 - \mathbf{R}_{ui}) \cdot \log(1 - \mathbf{S}_{ui}) + \lambda \cdot \alpha \cdot (||\theta_u||_F^2 + ||\theta_i||_F^2)\big),$$

with $||\cdot||_F$ the Frobenius norm, λ a regularization hyperparameter, and θ_u, θ_i the vectors that group the model parameters related to user u and item i, respectively.

5.2 Basic Squared Error-based

Most deviation functions, however, abandon the interpretation that the scores $\mathbf{S}$ are probabilities. In this case, one can choose to model $p(\mathbf{R}_{ui}|\theta)$ with a normal distribution $\mathcal{N}(\mathbf{R}_{ui}|\mathbf{S}_{ui}, \sigma)$. By additionally adopting the AMAN assumption, the *optimal* parameters are computed as the ones that maximize the loglikelihood $\log p(\mathbf{R}|\theta)$:

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \log \mathcal{N}(\mathbf{R}_{ui}|\mathbf{S}_{ui}, \sigma),$$

which is equivalent to

$$\theta^* = \arg\max_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} -(\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 = \arg\min_{\theta} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} (\mathbf{R}_{ui} - \mathbf{S}_{ui})^2.$$

The Eckart-Young theorem [13] states that the scores matrix $\mathbf{S}$ that results from these parameters θ^*, is the same as that found by singular value decomposition (SVD) with the same dimensions of the training data matrix $\mathbf{R}$. As such, the theorem relates the above approach of minimizing the squared error between $\mathbf{R}$ and $\mathbf{S}$ to *latent semantic analysis (LSA)* [9] and the SVD based collaborative filtering methods [8; 49].

Alternatively, it is possible to compute the *optimal* parameters as those that maximize the logposterior $\log p(\theta|\mathbf{R})$, which relates to the loglikelihood $\log p(\mathbf{R}|\theta)$ as

$$p(\theta|\mathbf{R}) \propto p(\mathbf{R}|\theta) \cdot p(\theta).$$

When $p(\theta)$, the prior distribution of the parameters, is chosen to be a zero-mean, spherical normal distribution, maximizing the logposterior is equivalent to minimizing the de-

viation function [32]

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \big((\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 + \lambda_u \cdot ||\theta_u||_F^2 + \lambda_i \cdot ||\theta_i||_F^2 \big),$$

with λ_u, λ_i regularization hyperparameters. Hence, maximizing the logposterior instead of the loglikelihood is equivalent to adding a regularization term. This deviation function is adopted by the FISM and NLMF methods [27; 26].
In an alternative interpretation of this deviation function, $\mathbf{S}$ is a factorized approximation of $\mathbf{R}$ and the deviation function minimizes the squared error of the approximation. The regularization with the Frobenius norm is added to avoid overfitting. For the SLIM and HOSLIM methods, an alternative regularization term is proposed:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} (\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_R ||\mathbf{S}^{(t,f)}||_F^2 + \lambda_1 ||\mathbf{S}^{(t,f)}||_1,$$

with $||\cdot||_1$ the l_1-norm. Whereas the role of the Frobenius-norm is to avoid overfitting, the role of the l_1-norm is to introduce sparsity. The combined use of both norms is called elastic net regularization, which is known to implicitly group correlated items [34]. The sparsity induced by the l_1-norm regularization lowers the memory required for storing the model and allows to speed-up the computation of recommendations by means of the sparse dotproduct. Even more sparsity can be obtained by fixing a majority of the parameters to 0, based on a simple feature selection method. Ning and Karypis [34] empirically show that this significantly reduces runtimes, while only slightly reducing the accuracy.

5.3 Weighted Squared Error-based

These squared error based deviation functions can also be adapted to diverge from the AMAN assumption to a missing data model that attaches lower confidence to the missing preferences [21; 37; 56]:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{W}_{ui} (\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_{tf} ||\mathbf{S}^{(t,f)}||_F^2, \quad (25)$$

in which $\mathbf{W} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$ assigns weights to values in $\mathbf{R}$. The higher $\mathbf{W}_{ui}$, the higher the confidence about $\mathbf{R}_{ui}$. Hu et al. [21] provide two potential definitions of $\mathbf{W}$:

$$\mathbf{W}_{ui} = 1 + \beta \mathbf{R}_{ui},$$
$$\mathbf{W}_{ui} = 1 + \alpha \log(1 + \mathbf{R}_{ui}/\epsilon),$$

with α, β, ϵ hyperparameters. Clearly, this method is not limited to binary data, but works on positive-only data in general. We, however, only consider its use for binary, positive-only data. Equivalently, Steck [56] proposed:

$$\mathbf{W}_{ui} = \mathbf{R}_{ui} + (1 - \mathbf{R}_{ui}) \cdot \alpha,$$

with $\alpha < 1$.
Additionally, Steck [57] pointed out that a preference is more likely to show up in the training data if the item is more popular. To compensate for this bias, Steck proposes to weight the known preferences non-uniformly:

$$\mathbf{W}_{ui} = \mathbf{R}_{ui} \cdot \frac{C}{c(i)^\beta} + (1 - \mathbf{R}_{ui}) \cdot \alpha,$$

with C a constant, R the number of non-zeros in the training data $\mathbf{R}$, and $\beta \in [0, 1]$ a hyperparameter. Analogously, a preference is more likely to be missing in the training data if the item is less popular. To compensate for this bias, Steck proposes to weight the missing preferences non-uniformly:

$$\mathbf{W}_{ui} = \mathbf{R}_{ui} + (1 - \mathbf{R}_{ui}) \cdot C \cdot c(i)^\beta, \quad (26)$$

with C a constant. Steck proposed these two weighting strategies as alternatives to each other. However, we believe that they can be combined since they are the application of the same idea to the known and missing preferences respectively.
Although they provide less motivation, Pan et al. [37] arrive at similar weighting schemes. They propose $\mathbf{W}_{ui} = 1$ if $\mathbf{R}_{ui} = 1$ and give three possibilities for the case when $\mathbf{R}_{ui} = 0$:

$$\mathbf{W}_{ui} = \delta, \quad (27)$$
$$\mathbf{W}_{ui} = \alpha \cdot c(u), \quad (28)$$
$$\mathbf{W}_{ui} = \alpha \left(|\mathcal{U}| - c(i) \right), \quad (29)$$

with $\delta \in [0, 1]$ a uniform hyperparameter and α a hyperparameter such that $\mathbf{W}_{ui} \leq 1$ for all pairs (u, i) for which $\mathbf{R}_{ui} = 0$. In the first case, all missing preferences get the same weight. In the second case, a missing preference is more negative if the user already has many preferences. In the third case, a missing preference is less negative if the item is popular. Interestingly, the third weighting scheme is orthogonal to the one of Steck in Equation 26. Additionally, Pan et al. [37] propose a deviation function with an alternative regularization:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{W}_{ui} \big((\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 + \lambda \left(||\theta_u||_F^2 + ||\theta_i||_F^2 \right) \big). \quad (30)$$

Yao et al. [73] adopt a more complex missing data model that has a hyperparameter p, which indicates the overall likelihood that a missing preference is preferred. This translates into the deviation function:

$$\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \mathbf{W}_{ui} (1 - \mathbf{S}_{ui})^2 + \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} (1 - \mathbf{R}_{ui}) \mathbf{W}_{ui} (p - \mathbf{S}_{ui})^2 + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_{tf} ||\mathbf{S}^{(t,f)}||_F^2 \quad (31)$$

The special case with $p = 0$ reduces this deviation function to the one in Equation 25.
An even more complex missing data model and correspond-

ing deviation function was proposed by Sindhwani et al. [55]:

$$\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}\mathbf{R}_{ui}\mathbf{W}_{ui}(1-\mathbf{S}_{ui})^2 + \sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}(1-\mathbf{R}_{ui})\mathbf{W}_{ui}\left(\mathbf{P}_{ui}-\mathbf{S}_{ui}\right)^2 + \sum_{t=1}^{T}\sum_{f=1}^{F}\lambda_{tf}||\mathbf{S}^{(t,f)}||_F^2 - \lambda_H\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}(1-\mathbf{R}_{ui})H(\mathbf{P}_{ui}) \quad (32)$$

with $\mathbf{P}$ and $\mathbf{W}$ additional parameters of the model that need to be computed based on the data $\mathbf{R}$, together with all other parameters. The last term contains the entropy function H and serves as a regularizer for $\mathbf{P}$. Furthermore, they define the constraint

$$\frac{1}{|\mathcal{U}||\mathcal{I}|-|\mathcal{R}|}\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}\mathbf{P}_{ui}=p,$$

which expresses that the average probability that a missing value is actually one must be equal to the hyperparameter p. To reduce the computational cost, they fix $\mathbf{P}_{ui}=0$ for most (u,i)-pairs and randomly choose a few (u,i)-pairs for which $\mathbf{P}_{ui}$ is computed based on the data. It seems that this simplification completely offsets the modeling flexibility that was obtained by introducing $\mathbf{P}$. Additionally, they simplify $\mathbf{W}$ as the one-dimensional matrix factorization

$$\mathbf{W}_{ui}=\mathbf{V}_u\mathbf{V}_i.$$

A conceptual inconsistency of this deviation function is that although the recommendation score is given by $\mathbf{S}_{ui}$, $\mathbf{P}_{ui}$ could also be used. Hence, there exist two parameters for the same concept, which is, at best, ambiguous.

5.4 Maximum Margin-based

A disadvantage of the squared error-based deviation functions is their symmetry. For example, if $\mathbf{R}_{ui}=1$ and $\mathbf{S}_{ui}=0$, $(\mathbf{R}_{ui}-\mathbf{S}_{ui})^2=1$. This is desirable behavior because we want to penalize the model for predicting that a preference is not preferred. However if $\mathbf{S}_{ui}=2$, we obtain the same penalty: $(\mathbf{R}_{ui}-\mathbf{S}_{ui})^2=1$. This, on the other hand, is not desirable because we do not want to penalize the model for predicting that a preference will definitely be preferred.

A maximum margin-based deviation function does not suffer from this problem [36]:

$$\mathcal{D}\left(\theta,\mathbf{R}\right)=\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}\mathbf{W}_{ui}\cdot h\left(\tilde{\mathbf{R}}_{ui}\cdot\mathbf{S}_{ui}\right)+\lambda||\mathbf{S}||_{\Sigma}, \quad (33)$$

with $||.||_{\Sigma}$ the trace norm, λ a regularization hyperparameter, $h\left(\tilde{\mathbf{R}}_{ui}\cdot\mathbf{S}_{ui}\right)$ a smooth hinge loss given by Figure 3 [46], $\mathbf{W}$ given by one of the Equations 27-29 and the matrix $\tilde{\mathbf{R}}$ defined as

$$\begin{cases}\tilde{\mathbf{R}}_{ui}=1 & \text{if } \mathbf{R}_{ui}=1\\ \tilde{\mathbf{R}}_{ui}=-1 & \text{if } \mathbf{R}_{ui}=0.\end{cases}$$

This deviation function incorporates the confidence about the training data by means of $\mathbf{W}$ and the missing knowledge about the degree of preference by means of the hinge loss $h\left(\tilde{\mathbf{R}}_{ui}\cdot\mathbf{S}_{ui}\right)$. Since the degree of a preference $\mathbf{R}_{ui}=1$ is considered unknown, a value $\mathbf{S}_{ui}>1$ is not penalized if $\mathbf{R}_{ui}=1$.

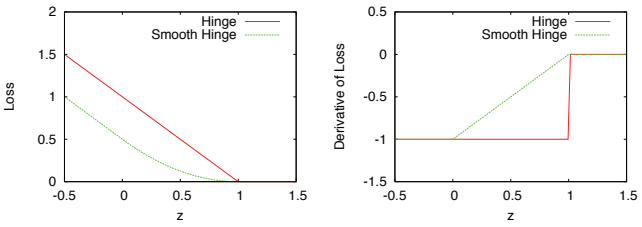

Figure 3: Shown are the loss function values $h(z)$ (left) and the gradients $dh(z)/dz$ (right) for the Hinge and Smooth Hinge. Note that the gradients are identical outside the region $z\in(0,1)$ [46].

5.5 Overall Ranking Error-based

The scores $\mathbf{S}$ computed by a collaborative filtering method are used to personally rank all items for every user. Therefore, one can argue that it is more natural to directly optimize the ranking of the items instead of their individual scores.

Rendle et al. [45], aim to maximize the area under the ROC curve (AUC), which is given by:

$$AUC=\frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|u|\cdot(|\mathcal{I}|-|u|)}\cdot\sum_{\mathbf{R}_{ui}=1}\sum_{\mathbf{R}_{uj}=0}\mathbb{I}(\mathbf{S}_{ui}>\mathbf{S}_{uj}). \quad (34)$$

If the AUC is higher, the pairwise rankings induced by the model $\mathbf{S}$ are more in line with the observed data $\mathbf{R}$. However, because $\mathbb{I}(\mathbf{S}_{ui}>\mathbf{S}_{uj})$ is non-differentiable, it is impossible to actually compute the parameters that (locally) maximize the AUC. Their solution is a deviation function, called the *Bayesian Personalized Ranking(BPR)-criterium*, which is a differentiable approximation of the negative AUC from which constant factors have been removed and to which a regularization term has been added:

$$\mathcal{D}\left(\theta,\mathbf{R}\right)=\sum_{u\in\mathcal{U}}\sum_{\mathbf{R}_{ui}=1}\sum_{\mathbf{R}_{uj}=0}-\log\sigma(\mathbf{S}_{ui}-\mathbf{S}_{uj}) + \sum_{t=1}^{T}\sum_{f=1}^{F}\lambda_{tf}||\mathbf{S}^{(t,f)}||_F^2, \quad (35)$$

with $\sigma(\cdot)$ the sigmoid function and λ_{tf} regularization hyperparameters. Since this approach explicitly accounts for the missing data, it corresponds to the MNAR assumption.

Pan and Chen claim that it is beneficial to relax the BPR deviation function by Rendle et al. to account for noise in the data [38; 39]. Specifically, they propose CoFiSet [38], which allows certain violations $\mathbf{S}_{ui}<\mathbf{S}_{uj}$ when $\mathbf{R}_{ui}>\mathbf{R}_{uj}$:

$$\mathcal{D}\left(\theta,\mathbf{R}\right)=\sum_{u\in\mathcal{U}}\sum_{I\subseteq I(u)}\sum_{J\subseteq\mathcal{I}\setminus I(u)}-\log\sigma\left(\frac{\sum_{i\in I}\mathbf{S}_{ui}}{|I|}-\frac{\sum_{j\in J}\mathbf{S}_{uj}}{|J|}\right)+\Gamma(\theta), \quad (36)$$

with $\Gamma(\theta)$ a regularization term that slightly deviates from the one proposed by Rendle et al. for no clear reason. Alternatively, they propose GBPR [39], which relaxes the BPR deviation function in a different way:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \Gamma(\theta) - \log \sigma^2 \left(\alpha \cdot \frac{\sum_{g \in \mathcal{G}_{u,i}} \mathbf{S}_{gi}}{|\mathcal{G}_{u,i}|} + (1-\alpha) \cdot \mathbf{S}_{ui} - \mathbf{S}_{uj} \right), \quad (37)$$

with $\mathcal{G}_{u,i}$ the union of $\{u\}$ with a random subset of $\{g \in \mathcal{U} \setminus \{u\} | \mathbf{R}_{gi} = 1\}$, and α a hyperparameter.
Furthermore, Kabbur et al. [27] also aim to maximize AUC with their method FISMauc. However, they propose to use a different differentiable approximation of AUC:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} ((\mathbf{R}_{ui} - \mathbf{R}_{uj}) - (\mathbf{S}_{ui} - \mathbf{S}_{uj}))^2 + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_{tf} ||\mathbf{S}^{(t,f)}||_F^2. \quad (38)$$

The same model without regularization was proposed by Töscher and Jahrer [62]:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} ((\mathbf{R}_{ui} - \mathbf{R}_{uj}) - (\mathbf{S}_{ui} - \mathbf{S}_{uj}))^2. \quad (39)$$

A similar deviation function was proposed by Takács and Tikk [61]:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \sum_{j \in \mathcal{I}} w(j) \cdot ((\mathbf{S}_{ui} - \mathbf{S}_{uj}) - (\mathbf{R}_{ui} - \mathbf{R}_{uj}))^2, \quad (40)$$

with $w(\cdot)$ a user-defined item weighting function. The simplest choice is $w(j) = 1$ for all j. An alternative proposed by Takács and Tikk is $w(j) = c(j)$. Another important difference is that this deviation function also minimizes the score-difference between the known preferences mutually.
Finally, it is remarkable that both Töscher and Jahrer, and Takács and Tikk, explicitly do not add a regularization term, whereas most other authors find that the regularization term is important for their model's performance.

5.6 Top of Ranking Error-based

Very often, only the N highest ranked items are shown to users. Therefore, Shi et al. [52] propose to improve the top of the ranking at the cost of worsening the overall ranking. Specifically, they propose the CLiMF method, which aims to minimize the *mean reciprocal rank (MRR)* instead of the AUC. The MRR is defined as

$$MRR = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} r_{>} \left(\max_{\mathbf{R}_{ui}=1} \mathbf{S}_{ui} \mid \mathbf{S}_{u\cdot} \right)^{-1},$$

in which $r_{>}(a|\mathbf{b})$ gives the rank of a among all numbers in $\mathbf{b}$ when ordered in descending order. Unfortunately, the non-smoothness of $r_{>}()$ and max makes the direct optimization of MRR unfeasible. Hence, Shi et al. derive a differentiable version of MRR. Yet, optimizing it is intractable. Therefore, they optimize a lower bound instead. After also adding regularization terms, their final deviation function is given by

$$\mathcal{D}(\theta, \mathbf{R}) = -\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui} \Big(\log \sigma(\mathbf{S}_{ui}) + \sum_{j \in \mathcal{I}} \log (1 - \mathbf{R}_{uj} \sigma(\mathbf{S}_{uj} - \mathbf{S}_{ui})) \Big) + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_{tf} ||\mathbf{S}^{(t,f)}||_F^2 \quad (41)$$

with λ a regularization constant and $\sigma(\cdot)$ the sigmoid function. A disadvantage of this deviation function is that it ignores all missing data, i.e., it corresponds to the MAR assumption.
An alternative for MRR is *mean average precision (MAP)*:

$$MAP = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{c(u)} \sum_{\mathbf{R})_{ui}=1} \frac{1}{r_{>}(\mathbf{S}_{ui}|\mathbf{S}_{u\cdot})} \sum_{\mathbf{R}_{uj}=1} \mathbb{I}(\mathbf{S}_{uj} \geq \mathbf{S}_{ui}),$$

which still emphasizes the top of the ranking, but less extremely than MRR. However, MAP is also non-smooth, preventing its direct optimization. For MAP, too, Shi et al. derived a differentiable version, called TFMAP [51]:

$$\mathcal{D}(\theta, \mathbf{R}) = -\sum_{u \in \mathcal{U}} \frac{1}{c(u)} \sum_{\mathbf{R})_{ui}=1} \sigma(\mathbf{S}_{ui}) \sum_{\mathbf{R}_{uj}=1} \sigma(\mathbf{S}_{uj} - \mathbf{S}_{ui}) + \lambda \cdot \sum_{t=1}^{T} \sum_{f=1}^{F} ||\mathbf{S}^{(t,f)}||_F^2, \quad (42)$$

with λ a regularization hyperparameter. Besides, the formulation of TFMAP in Equation 42 is a simplified version of the original, concieved for multi-context data, which is out of the scope of this survey.
Dhanjal et al. proposed yet another alternative [12]. They start from the definition of AUC in Equation 34, approximate the indicator function $\mathbb{I}(\mathbf{S}_{ui} - \mathbf{S}_{uj})$ by the squared hinge loss $(\max(0, 1 + \mathbf{S}_{uj} - \mathbf{S}_{ui}))^2$ and emphasize the deviation at the top of the ranking by means of the hyperbolic tangent function $\tanh(\cdot)$:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{\mathbf{R}_{ui}=1} \tanh \left(\sum_{\mathbf{R}_{uj}=0} (\max(0, 1 + \mathbf{S}_{uj} - \mathbf{S}_{ui}))^2 \right). \quad (43)$$

5.7 k-Order Statistic-based

On the one hand, the deviation functions in Equations 35-40 try to minimize the overall rank of the known preferences. On the other hand, the deviation functions in Equations 41 and 43 try to push one or a few known preferences as high as possible to the top of the item-ranking. Weston et al. [71] propose to minimize a trade-off between the above

two extremes:

$$\sum_{u\in\mathcal{U}}\sum_{\mathbf{R}_{ui}=1} w\left(\frac{r_>(\mathbf{S}_{ui}\mid\{\mathbf{S}_{ui}\mid\mathbf{R}_{ui}=1\})}{c(u)}\right) \cdot \sum_{\mathbf{R}_{uj}=0}\frac{\mathbb{I}(\mathbf{S}_{uj}+1\geq\mathbf{S}_{ui})}{r_>(\mathbf{S}_{uj}\mid\{\mathbf{S}_{uk}\mid\mathbf{R}_{uk}=0\})}, \quad (44)$$

with $w(\cdot)$ a function that weights the importance of the known preference as a function of their predicted rank among all known preferences. This weighting function is user-defined and determines the trade-off between the two extremes, i.e., minimizing the mean rank of the known preferences and minimizing the maximal rank of the known preferences. Because this deviation function is non-differentiable, Weston et al. propose the differentiable approximation

$$\mathcal{D}(\theta,\mathbf{R}) = \sum_{u\in\mathcal{U}}\sum_{\mathbf{R}_{ui}=1} w\left(\frac{r_>(\mathbf{S}_{ui}\mid\{\mathbf{S}_{ui}\mid\mathbf{R}_{ui}=1\})}{c(u)}\right) \cdot \sum_{\mathbf{R}_{uj}=0}\frac{\max(0,1+\mathbf{S}_{uj}-\mathbf{S}_{ui})}{N^{-1}\left(|\mathcal{I}|-c(u)\right)}, \quad (45)$$

where they replaced the indicator function by the hinge-loss and approximated the rank with $N^{-1}\left(|\mathcal{I}|-c(u)\right)$, in which N is the number of items k that were randomly sampled until $\mathbf{S}_{uk}+1>\mathbf{S}_{ui}$[3]. Furthermore, they use the simple weighting function

$$w\left(\frac{r_>(\mathbf{S}_{ui}\mid\{\mathbf{S}_{ui}\mid\mathbf{R}_{ui}=1\})}{|u|}\right) = \begin{cases}1 \textbf{ if } r_>(\mathbf{S}_{ui}\mid S\subseteq\{\mathbf{S}_{ui}\mid\mathbf{R}_{ui}=1\},|S|=K)=k \textbf{ and}\\ 0 \textbf{ otherwise }\end{cases},$$

i.e., from the set S that contains K randomly sampled known preferences, ranked by their predicted score, only the item at rank k is selected to contribute to the training error. When k is set low, the top of the ranking will be optimized at the cost of a worse mean rank. The opposite will hold when k is set high. The regularization is not done by adding a regularization term but by forcing the norm of the factor matrices to be below a maximum, which is a hyperparameter.
Alternatively, Weston et al. also propose a simplified deviation function:

$$\mathcal{D}(\theta,\mathbf{R}) = \sum_{u\in\mathcal{U}}\sum_{\mathbf{R}_{ui}=1} w\left(\frac{r_>(\mathbf{S}_{ui}\mid\{\mathbf{S}_{ui}\mid\mathbf{R}_{ui}=1\})}{c(u)}\right) \cdot \sum_{\mathbf{R}_{uj}=0}\max(0,1+\mathbf{S}_{uj}-\mathbf{S}_{ui}). \quad (46)$$

5.8 Rank Link Function-based

The ranking-based deviation functions discussed so far, are all tailor made differentiable approximations with respect to the recommendation scores of a certain ranking quality measure, like AUC, MRR or the k-th order statistic. Steck [58] proposes a more general approach that is applicable to any ranking quality measure that is differentiable with respect to the rankings. He demonstrates his method on two ranking quality measures: AUC and $nDCG$. For AUC_u, the AUC for user u, he does not use the formulation in Equation 34, but uses the equivalent

$$AUC_u = \frac{1}{c(u)\cdot(|\mathcal{I}|-c(u))} \cdot\left[(|\mathcal{I}|+1)c(u)-\binom{c(u)+1}{2}-\sum_{\mathbf{R}_u i=1} r_{ui}\right],$$

with r_{ui} the rank of item i in the recommendation list of user u. Second, $nDCG_u$ is defined as

$$nDCG_u = \frac{DCG}{DCG_{opt}}, \qquad DCG = \sum_{\mathbf{R}_{ui}=1}\frac{1}{\log(r_{ui}+1)},$$

with DCG_{opt} the DCG of the optimal ranking. In both cases, Steck proposes to relate the rank r_{ui} with the recommendation score $\mathbf{R}_{ui}$ by means of a link function

$$r_{ui} = \max\left\{1, |\mathcal{I}|\cdot\left(1-cdf(\hat{\mathbf{S}}_{ui})\right)\right\}, \quad (47)$$

with $\hat{\mathbf{S}}_{ui}=(\mathbf{S}_{ui}-\mu_u)/std_u$ the normalized recommendation score in which μ_u and std_u are the mean and standard deviation of the recommendation scores for user u, and cdf is the cumulative distribution of the normalized scores. However, to know cdf, he needs to assume a distribution for the normalized recommendation scores. He motivates that a zero-mean normal distribution of the recommendation scores is a reasonable assumption. Consequently, $cdf(\hat{\mathbf{S}}_{ui}) = probit(\hat{\mathbf{S}}_{ui})$. Furthermore, he proposes to approximate the probit function with the computationally more efficient sigmoid function or the even more efficient function

$$rankSE(\hat{\mathbf{S}}_{ui}) = [1-([1-\hat{\mathbf{S}}_{ui}]_+)^2]_+,$$

with $[x]_+ = max\{0,x\}$. In his *pointwise* approach, Steck uses Equation 47 to compute the ranks based on the recommendation scores. In his *listwise* approach, on the other hand, he finds the actual rank of a recommendation score by sorting the recommendation scores for every user, and uses Equation 47 only to compute the gradient of the rank with respect to the recommendation score during the minimization procedure. After adding regularizaton terms, he proposes the deviation function

$$\mathcal{D}(\theta,\mathbf{R}) = \sum_{u\in\mathcal{U}}\left(\sum_{\mathbf{R}_{ui}=1}\left(L(\mathbf{S}_{ui})+\lambda\cdot\left(||\theta_u||_F^2+||\theta_i||_F^2\right)\right) + \sum_{j\in\mathcal{I}}\gamma\cdot[\mathbf{S}_{uj}]_+^2\right), \quad (48)$$

with θ_u,θ_i the vectors that group all model parameters related to user u and item i, respectively, λ,γ regularization hyperparameters, and $L(\mathbf{S}_{ui})$ equal to $-AUC_u$ or $-nDCG_u$, which are a function of $\mathbf{S}_{ui}$ via r_{ui}. The last regularization term is introduced to enforce an approximately normal distribution on the scores.

5.9 Posterior KL-divergence-based

In our framework, the *optimal* parameters θ^* are computed as $\theta^* = \arg\min_\theta \mathcal{D}(\theta,\mathbf{R})$. However, we can consider this a special case of

$$\theta^* = \psi\left(\arg\min_\phi \mathcal{D}(\theta(\phi),\mathbf{R})\right),$$

[3]Weston et al. [69] provide a justification for this approximation.

in which ψ is chosen to be the identity function and the parameters θ are identical to the parameters ϕ, i.e., $\theta(\phi) = \phi$. Now, some authors [28; 40; 16] propose to choose $\psi()$ and ϕ differently.

Specifically, they model every parameter θ_j of the factorization model as a random variable with a parameterized posterior probability density function $p(\theta_j|\phi_j)$. Hence, finding the variables ϕ corresponds to finding the posterior distributions of the model parameters θ. Because it is intractable to find the true posterior distribution $p(\theta|\mathbf{R})$ of the parameters, they settle for a *mean-field* approximation $q(\theta|\phi)$, in which all variables are assumed to be independent.

Then, they define ψ as $\psi(\phi^*) = \mathbb{E}_{q(\theta|\phi^*)}[\theta]$, i.e., the point-estimate of the parameters θ equals their expected value under the mean-field approximation of their posterior distributions. Note that $\theta_j^* = \mathbb{E}_{q(\theta_j|\phi_j^*)}[\theta_j]$ because of the independence assumption.

If all parameters θ_j are assumed to be normally distributed with mean μ_j and variance σ_j^2 [28; 40], $\phi_j = (\mu_j, \sigma_j)$ and $\theta_j^* = \mathbb{E}_{q(\theta_j|\phi_j^*)}[\theta_j] = \mu_j^*$. If, on the other hand, all parameters θ_j are assumed to be gamma distributed with shape α_j and rate β_j [16], $\phi_j = (\alpha_j, \beta_j)$ and $\theta_j^* = \mathbb{E}_{q(\theta_j|\phi_j^*)}[\theta_j] = \alpha_j^*/\beta_j^*$. Furthermore, prior distributions are defined for all parameters θ. Typically, when this approach is adopted, the underlying assumptions are represented as a *graphical model* [5].

The parameters ϕ, and therefore the corresponding mean-field approximations of the posterior distribution of the parameters θ, can be inferred by defining the deviation function as the KL-divergence of the real (unknown) posterior distribution of the parameters, $p(\theta|\mathbf{R})$, from the modeled posterior distribution of the parameters, $q(\theta|\phi)$,

$$\mathcal{D}\left(\theta(\phi), \mathbf{R}\right) = D_{KL}\left(q(\theta|\phi) \| p(\theta|\mathbf{R})\right),$$

which can be solved despite the fact that $p(\theta|\mathbf{R})$ is unknown [25]. This approach goes by the name *variational inference* [25].

A *nonparametric* version of this approach also considers D, the number of latent dimensions in the simple two factor factorization model of Equation 7, as a parameter that depends on the data $\mathbf{R}$ instead of a hyperparameter, as most other methods do [17].

Note that certain solutions for latent Dirichlet allocation [6] also use variational inference techniques. However, in this case, variational inference is a part of the (variational) expectation-maximization algorithm for computing the parameters that optimize the negative log-likelihood of the model parameters, which serves as the deviation function. This is different from the methods discussed in this section, where the KL-divergence between the real and the approximate posterior is the one and only deviation function.

5.10 Convex

An intuitively appealing but non-convex deviation function is worthless if there exists no algorithm that can efficiently compute parameters that correspond to its good local minimum. Convex deviation functions on the other hand, have only one global minimum that can be computed with one of the well studied convex optimization algorithms. For this reason, it is worthwhile to pursue convex deviation functions.

Aiolli [3] proposes a convex deviation function based on the AUC (Eq. 34). For every individual user $u \in \mathcal{U}$, Aiolli starts from AUC_u, the AUC for u:

$$AUC_u = \frac{1}{c(u) \cdot (|\mathcal{I}| - c(u))} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \mathbb{I}(\mathbf{S}_{ui} > \mathbf{S}_{uj}).$$

Next, he proposes a lower bound on AUC_u:

$$AUC_u \geq \frac{1}{c(u) \cdot (|\mathcal{I}| - c(u))} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \frac{\mathbf{S}_{ui} - \mathbf{S}_{uj}}{2},$$

and interprets it as a weighted sum of margins $\frac{\mathbf{S}_{ui} - \mathbf{S}_{uj}}{2}$ between any known preferences and any absent feedback, in which every margin gets the same weight $\frac{1}{c(u) \cdot (|\mathcal{I}| - c(u))}$. Hence maximizing this lower bound on the AUC corresponds to maximizing the sum of margins between any known preference and any absent feedback in which every margin has the same weight. A problem with maximizing this sum is that very high margins on pairs that are easily ranked correctly can hide poor (negative) margins on pairs that are difficult to rank correctly. Aiolli proposes to replace the uniform weights with a weighting scheme that emphasizes the difficult pairs such that the total margin is the worst possible case, i.e., the lowest possible sum of weighted margins. Specifically, he proposes to solve for every user u the joint optimization problem

$$\theta^* = \arg\max_{\theta} \min_{\alpha_{u*}} \sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \alpha_{ui}\alpha_{uj}(\mathbf{S}_{ui} - \mathbf{S}_{uj}),$$

where for every user u, it holds that $\sum_{\mathbf{R}_{ui}=1} \alpha_{ui} = 1$ and $\sum_{\mathbf{R}_{uj}=0} \alpha_{uj} = 1$. To avoid overfitting of α, he adds two regularization terms:

$$\theta^* = \arg\max_{\theta} \min_{\alpha_{u*}} \Bigg(\sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \alpha_{ui}\alpha_{uj}(\mathbf{S}_{ui} - \mathbf{S}_{uj}) + \lambda_p \sum_{\mathbf{R}_{ui}=1} \alpha_{ui}^2 + \lambda_n \sum_{\mathbf{R}_{ui}=0} \alpha_{ui}^2 \Bigg),$$

with λ_p, λ_n regularization hyperparameters. The model parameters θ, on the other hand, are regularized by normalization constraints on the factor matrices. Solving the above maximization for every user, is equivalent to minimizing the deviation function

$$\mathcal{D}\left(\theta, \mathbf{R}\right) = \sum_{u \in \mathcal{U}} \Bigg(\max_{\alpha_{u*}} \Bigg(\sum_{\mathbf{R}_{ui}=1} \sum_{\mathbf{R}_{uj}=0} \alpha_{ui}\alpha_{uj}(\mathbf{S}_{uj} - \mathbf{S}_{ui}) - \lambda_p \sum_{\mathbf{R}_{ui}=1} \alpha_{ui}^2 - \lambda_n \sum_{\mathbf{R}_{ui}=0} \alpha_{ui}^2 \Bigg) \Bigg). \quad (49)$$

5.11 Analytically Solvable

Some deviation functions are not only convex, but also analytically solvable. This means that the parameters that minimize these deviation functions can be exactly computed from a formula and that no numerical optimization algorithm is required.

Traditionally, methods that adopt these deviation functions have been inappropriately called *neighborhood* or *memory-based* methods. First, although these methods adopt neighborhood-based factorization models, there are also neighborhood-based methods that adopt non-convex deviation functions, such as SLIM [34] and BPR-kNN [45], which were,

amongst others, discussed in Section 4. Second, a *memory-based* implementation of these methods, in which the necessary parameters of the factorization model are not precomputed, but computed in real time when they are required is conceptually possible, yet practically intractable in most cases. Instead, a *model-based* implementation of these methods, in which the factorization model is precomputed, is the best choice for the majority of applications.

5.11.1 Basic Neighborhood-based

A first set of analytically solvable deviation functions is tailored to the item-based neighborhood factorization models of Equation 10:

$$\mathbf{S} = \mathbf{R}\mathbf{S}^{(1,2)}.$$

As explained in Section 4, the factor matrix $\mathbf{S}^{(1,2)}$ can be interpreted as an item-similarity matrix. Consequently, these deviation functions compute every parameter in $\mathbf{S}^{(1,2)}$ as

$$\mathbf{S}^{(1,2)}_{ji} = sim(j,i),$$

with $sim(j,i)$ the similarity between items j and i according to some analytically computable similarity function. This is equivalent to

$$\mathbf{S}^{(1,2)}_{ji} - sim(j,i) = 0,$$

which is true for all (j,i)-pairs if and only if

$$\sum_{j\in\mathcal{I}}\sum_{i\in\mathcal{I}}\left(\mathbf{S}^{(1,2)}_{ji} - sim(j,i)\right)^2 = 0.$$

Hence, computing the factor matrix $\mathbf{S}^{(1,2)}_{ji}$ corresponds to minimizing the deviation function

$$\mathcal{D}\left(\theta,\mathbf{R}\right) = \sum_{j\in\mathcal{I}}\sum_{i\in\mathcal{I}}\left(\mathbf{S}^{(1,2)}_{ji} - sim(j,i)\right)^2.$$

In this case, the deviation function mathematically expresses the interpretation that $sim(j,i)$ is a good predictor for preferring i if j is also preferred. The key property that determines the analytical solvability of this deviation function is the absence of products of parameters. The non-convex deviation functions in Section 6.1, on the other hand, do contain products of parameters, which are contained in the term $\mathbf{S}_{ui}$. Consequently, they are harder to solve but allow richer parameter interactions.

A typical choice for $sim(j,i)$ is the cosine similarity [10]. The cosine similarity between two items j and i is given by:

$$cos(j,i) = \frac{\sum_{v\in\mathcal{U}}\mathbf{R}_{vj}\mathbf{R}_{vi}}{\sqrt{c(i)\cdot c(j)}}. \tag{50}$$

Another similarity measure is the conditional probability similarity measure [10], which is for two items i and j given by:

$$condProb(j,i) = \sum_{v\in\mathcal{U}}\frac{\mathbf{R}_{vi}\mathbf{R}_{vj}}{c(j)}. \tag{51}$$

Deshpande and Karypis also proposed an adapted version:

$$condProb^*(j,i) = \sum_{v\in\mathcal{U}}\frac{\mathbf{R}_{vi}\mathbf{R}_{vj}}{c(i)\cdot c(j)^\alpha\cdot c(v)}, \tag{52}$$

in which $\alpha\in[0,1]$ is a hyperparameter. They introduced the factor $1/c(j)^\alpha$ to avoid the recommendation of overly frequent items and the factor $1/c(v)$ to reduce the weight of i and j co-occurring in the preferences of v, if v has more preferences. Other similarity measures were proposed by Aiolli [2]:

$$sim(j,i) = \left(\sum_{v\in\mathcal{U}}\frac{\mathbf{R}_{vi}\mathbf{R}_{vj}}{c(j)^\alpha\cdot c(i)^{(1-\alpha)}}\right)^q,$$

with α, q hyperparameters, Gori et al. [18]:

$$sim(j,i) = \frac{\sum_{v\in\mathcal{U}}\mathbf{R}_{vj}\mathbf{R}_{vi}}{\sum_{k\in\mathcal{I}}\sum_{v\in\mathcal{U}}\mathbf{R}_{vj}\mathbf{R}_{vk}},$$

and Wang et al. [68]:

$$sim(j,i) = \log\left(1+\alpha\cdot\frac{\sum_{v\in\mathcal{U}}\mathbf{R}_{vj}\mathbf{R}_{vi}}{c(j)c(i)}\right),$$

with $\alpha\in\mathbb{R}_0^+$ a hyperparameter. Furthermore, Huang et al. show that $sim(j,i)$ can also be chosen from a number of similarity measures that are typically associated with link prediction [22]. Similarly, Bellogin et al. show that typical scoring functions used in information retrieval can also be used for $sim(j,i)$ [4].

It is common practice to introduce sparsity in $\mathbf{S}^{(1,2)}$ by defining

$$sim(j,i) = sim'(j,i)\cdot|KNN(j)\cap\{i\}|, \tag{53}$$

with $sim'(j,i)$ one of the similarity functions defined by Equations 50-52, $KNN(j)$ the set of items l that correspond to the k highest values $sim'(j,l)$, and k a hyperparameter. Motivated by a qualitative examination of their results, Sigurbjörnsson and Van Zwol [54] proposed additional adaptations:

$$sim(j,i) = s(j)\cdot d(i)\cdot r(j,i)\cdot sim'(j,i)\cdot|KNN(j)\cap\{i\}|,$$

with

$$s(j) = \frac{k_s}{k_s + |k_s - \log c(j)|}, \tag{54}$$

$$d(i) = \frac{k_d}{k_d + |k_d - \log c(i)|}, \tag{55}$$

$$r(j,i) = \frac{k_r}{k_r + (r-1)}, \tag{56}$$

in which i is the r-th most similar item to j and k_s, k_d and k_r are hyperparameters.

Finally, Desphande and Karypis [10] propose to normalize $sim(j,i)$ as

$$sim(j,i) = \frac{sim''(j,i)}{\sum_{l\in\mathcal{I}\setminus\{j\}}sim''(j,l)},$$

with $sim''(j,i)$ defined using Equation 53. Alternatively, Aiolli [2] proposes the normalization

$$sim(j,i) = \frac{sim''(j,i)}{\sum_{l\in\mathcal{I}\setminus\{i\}}sim''(l,i)^{2(1-\beta)}},$$

with β a hyperparameter.

A second set of analytically solvable deviation functions is tailored to the user-based neighborhood factorization model of Equation 13:

$$\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{R}.$$

In this case, the factor matrix $\mathbf{S}^{(1,1)}$ can be interpreted as a user-similarity matrix. Consequently, these deviation functions compute every parameter in $\mathbf{S}^{(1,1)}$ as

$$\mathbf{S}^{(1,1)}_{uv} = sim(u,v),$$

with $sim(u,v)$ the similarity between users u and v according to some analytically computable similarity function. In the same way as for the item-based case, computing the factor matrix $\mathbf{S}^{(1,1)}_{uv}$ corresponds to minimizing the deviation function

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{v \in \mathcal{U}} \left(\mathbf{S}^{(1,1)}_{uv} - sim(u,v) \right)^2.$$

In this case, the deviation function mathematically expresses the interpretation that users u and v for which $sim(u,v)$ is high, prefer the same items.
Sarwar et al. [48] propose

$$sim(u,v) = |KNN(u) \cap \{v\}|,$$

with $KNN(u)$ the set of users w that have the k highest cosine similarities $\cos(u,w)$ with user u, and k a hyperparameter. In this case, cosine similarity is defined as

$$cos(u,v) = \frac{\sum_{j \in \mathcal{I}} \mathbf{R}_{uj} \mathbf{R}_{vj}}{\sqrt{c(u) \cdot c(v)}}. \tag{57}$$

Alternatively, Aiolli [2] proposes

$$sim(u,v) = \left(\sum_{j \in \mathcal{I}} \frac{\mathbf{R}_{uj} \mathbf{R}_{vj}}{c(u)^{\alpha} \cdot c(v)^{(1-\alpha)}} \right)^q,$$

with α, q hyperparameters, and Wang et al. [68] propose

$$sim(u,v) = \log \left(1 + \alpha \cdot \frac{\sum_{j \in \mathcal{U}} \mathbf{R}_{uj} \mathbf{R}_{vj}}{c(u)c(v)} \right),$$

with α a hyperparameter.
The deviation function for the unified neighborhood based factorization model in Equation 14 is given by

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{v \in \mathcal{U}} \left(\mathbf{S}^{(1,1)}_{uv} - sim(u,v) \right)^2 + \sum_{j \in \mathcal{I}} \sum_{i \in \mathcal{I}} \left(\mathbf{S}^{(2,3)}_{ji} - sim(j,i) \right)^2.$$

However, $sim(j,i)$ and $sim(u,v)$ cannot be chosen arbitrarily. Verstrepen and Goethals [67] show that they need to satisfy certain constraints in order to render a well founded unification. Consequently, they propose KUNN, which corresponds to the following similarity definitions that satisfy the necessary constraints:

$$sim(u,v) = \sum_{j \in \mathcal{I}} \frac{\mathbf{R}_{uj} \mathbf{R}_{vj}}{\sqrt{c(u) \cdot c(v) \cdot c(j)}}$$

$$sim(i,j) = \sum_{v \in \mathcal{U}} \frac{\mathbf{R}_{vi} \mathbf{R}_{vj}}{\sqrt{c(i) \cdot c(j) \cdot c(v)}}.$$

5.11.2 Higher Order Neighborhood-based

A fourth set of analytically solvable deviation functions is tailored to the higher order itemset-based neighborhood factorization model of Equation 20:

$$\mathbf{S} = \mathbf{X}\mathbf{S}^{(1,2)}.$$

In this case, the deviation function is given by

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{j \in \mathcal{S}} \sum_{i \in \mathcal{I}} \left(\mathbf{S}^{(1,2)}_{ji} - sim(j,i) \right)^2.$$

with $\mathcal{S} \subseteq 2^{\mathcal{I}}$ the selected itemsets considered in the factorization model.
Deshpande and Karypis [10] propose to define $sim(j,i)$ similarly as for the pairwise interactions (Eq. 53). Alternatively, others [33; 48] proposed

$$sim(j,i) = sim'(j,i) \cdot \max\left(0, c(j \cup \{i\}) - f\right),$$

with f a hyperparameter.
Lin et al. [30] proposed yet another alternative:

$$sim(j,i) = sim'(j,i) \cdot |KNN_c(i) \cap \{j\}| \cdot \max\left(0, condProb(j,i) - c\right),$$

with $KNN_c(i)$ the set of items l that correspond to the k highest values $c(i,l)$, k a hyperparameter, *condProb* the conditional probability according to Equation 51 and c a hyperparameter. Furthermore, they define

$$sim'(j,i) = \frac{\left(\sum_{v \in \mathcal{U}} \mathbf{X}_{vi} \mathbf{X}_{vj} \right)^2}{c(j)}. \tag{58}$$

A fifth and final set of analytically solvable deviation functions is tailored to the higher order userset-based neighborhood factorization model of Equation 21: $\mathbf{S} = \mathbf{S}^{(1,1)}\mathbf{Y}$. In this case, the deviation function is given by $\mathcal{D}(\theta, \mathbf{R}) = \sum_{v \in \mathcal{S}} \sum_{u \in \mathcal{U}} \left(\mathbf{S}^{(1,1)}_{uv} - sim(v,u) \right)^2$, with $\mathcal{S} \subseteq 2^{\mathcal{U}}$ the selected usersets considered in the factorization model. Lin et al. [30] proposed to define

$$sim'(v,u) = \frac{\left(\sum_{j \in \mathcal{I}} \mathbf{Y}_{uj} \mathbf{Y}_{vj} \right)^2}{c(v)}. \tag{59}$$

Alternatively, Symeonidis et al. [60] propose

$$sim(v,u) = \frac{\sum_{j \in \mathcal{I}} \mathbf{Y}_{uj} \mathbf{Y}_{vj}}{c(v)} \cdot |v| \cdot |KNN(u)_{cp} \cap \{v\}|,$$

with $KNN(u)_{cp}$ the set of usersets w that correspond to the k highest values

$$\frac{\sum_{j \in \mathcal{I}} \mathbf{Y}_{uj} \mathbf{Y}_{wj}}{c(w)}.$$

6. MINIMIZATION ALGORITHMS

Efficiently computing the parameters that minimize a deviation function is often non trivial. Furthermore, there is a big difference between minimizing convex and non-convex deviation functions.

6.1 Non-convex Minimization

The two most popular families of minimization algorithms for non-convex deviation functions of collaborative filtering algorithms are *gradient descent* and *alternating least squares*. We discuss both in this section. Furthermore, we also briefly discuss a few other interesting approaches.

6.1.1 Gradient Descent

For deviation functions that assume preferences are missing at random, and consequently consider only the known

preferences [52], gradient descent (GD) is generally the numerical optimization algorithm of choice. In GD, the parameters θ are randomly initialized. Then, they are iteratively updated in the direction that reduces $\mathcal{D}(\theta, \mathbf{R})$:

$$\theta^{k+1} = \theta^k - \eta \nabla \mathcal{D}(\theta, \mathbf{R}),$$

with η a hyperparameter called the learning rate. The update step is larger if the absolute value of the gradient $\nabla \mathcal{D}(\theta, \mathbf{R})$ is larger. A version of GD that converges faster is Stochastic Gradient Descent (SGD). SGD uses the fact that

$$\nabla \mathcal{D}(\theta, \mathbf{R}) = \sum_{t=1}^{T} \nabla \mathcal{D}_t(\mathbf{S}, \mathbf{R}),$$

with T the number of terms in $\mathcal{D}(\mathbf{S}, \mathbf{R})$. Now, instead of computing $\nabla \mathcal{D}(\theta, \mathbf{R})$ in every iteration, only one term t is randomly sampled (with replacement) and the parameters θ are updated as

$$\theta^{k+1} = \theta^k - \eta \nabla \mathcal{D}_t(\mathbf{S}, \mathbf{R}).$$

Typically, a convergence criterium of choice is only reached after every term t is sampled multiple times on average.

However, when the deviation function assumes the missing feedback is missing not at random, the summation over the known preferences, $\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{R}_{ui}$, is replaced by a summation over all user item pairs, $\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}}$, and SGD needs to visit approximately 1000 times more terms. This makes the algorithm less attractive for deviation functions that assume the missing feedback is missing not at random.

To mitigate the large number of terms in the gradient, several authors propose to sample the terms in the gradient not uniformly but proportional to their impact on the parameters [75; 44; 76]. These approaches have not only been proven to speed up convergence, but also to improve the quality of the resulting parameters. Weston et al., on the other hand, sample for every known preference i, a number of non preferred items j until they encounter one for which $\mathbf{S}_{uj} + 1 > \mathbf{S}_{ui}$, i.e., it violates the hinge-loss approximation of the ground truth ranking. In this way, they ensure that every update significantly changes the parameters θ [71].

Additionally, due to recent advances in distributed computing infrastructure, parallel and distributed approaches have been proposed to speed up the convergence of SGD-style computations.

One of the classic works is the HogWild algorithm by Recht et al. [43]. In that work, the authors assume that the rating matrix is highly sparse and hence, for any two sampled ratings, their SGD updates will likely be non-conflicting (independent), since any two such updates are unlikely to share either the user or item vectors. As a result, HogWild drops the synchronization requirements, and lets each thread or processor update a random rating value. In the worst case of a conflict, there will be contention in writing out the results. The authors prove convergence of the algorithm under a simple assumption of rating matrix sparsity.

Gemulla et al. [15] present a distributed SGD algorithm (DSGD) in which the main assumption is that some blocks of the rating matrix are mutually independent and hence their variables can be updated in parallel. DSGD uniformly grids the rating matrix R into sub-matrices or blocks, such that they are independent with respect to the rows and columns. This method generates several configurations of the original rating matrix R, which are then fed to the SGD algorithm in sequence. In the actual algorithm, there are two nested for loops. The outer loop selects a configuration C of the independent blocks and sends it to the SGD scheduler. The latter, simply spawns as many parallel threads as the number of independent blocks in each configuration and applies SGD to each of them in parallel. Once the results are back, the next configuration is loaded to each processor after consolidating the results of the last iteration. This process continues until all the configurations are computed. Zhuang et al. [78] argue that both these methods suffer from two problems. The first is the issue of locking of threads, in which a thread that is executing a slower block needs to finish before the next round of assignments can begin. Second, since the elements of the rating matrix are accessed at random, it may result in high cache-miss and performance degradation. To solve both these problems, the authors propose a shared memory parallel algorithm called Fast Parallel SGD (FPSGD). There are two main features of the algorithm. To overcome the locking issue, the authors suggest continuous execution of SGD blocks by the scheduler. The only two conditions it needs to satisfy are (1) it has to be a free block, and (2) its number of past updates has to be smallest among all the free blocks (random if there is a tie). The second condition is necessary because otherwise all blocks will not get same chance of being updated. For dealing with memory discontinuity, the authors suggest updating each block in a fixed order of users and items. The randomness of the algorithm is introduced in selecting each block for the next update. Since FPSGD always selects the next block in a deterministic fashion, one simple strategy to select the next block in a random fashion is to split the original rating matrix R into many sub blocks (more than the number of available threads). Since many blocks will have the same update number, randomization is needed to select the next block for update. The authors call this method *partial randomization*. The results of experiments on a variety of datasets show that FPSGD achieves lower RMSE in a far smaller number of iterations.

A natural strategy to optimize the update rules in matrix factorization (MF) is to do a block-wise update and let each block be handled by a separate processor/computing unit. This is the strategy proposed by Yin et. al [74]. The authors first show that most of the loss functions used in MF are decomposable in the sense that they are sums over individual loss terms. Hence they can easily be parallelized. Given $A = WH$ as the model, the proposal is to split W and H into $W^{(I)}$ and $H^{(J)}$, such that the total loss can be written as the sum of the losses over indices I and J. Then the task is to develop a block-wise partition rule, in which blocks are updated independently when updating a factor matrix (by fixing the other factor matrix). Each block can be treated as one update unit. There are several ways in which the blocks can be updated. A straightforward way is to update update all blocks of H and then update all blocks of W. This approach can be referred to as concurrent block updates since each update happens concurrently. An alternate strategy is to update some blocks of H and W alternately. This method, called the frequent block-wise update, has the advantage of faster convergence. This happens due to the fact that updated block information are used in each alternating sequence and hence converges faster. The remainder of the paper presents a recipe for implementing these algorithms in MapReduce.

6.1.2 Alternating Least Squares

If the deviation function allows it, alternating least squares (ALS) becomes an interesting alternative to SGD when preferences are assumed to be missing not at random [29; 21]. In this respect, the deviation functions of Equations 25, 30, 31, and 48 are, amongst others, appealing because they can be minimized with a variant of the alternating least squares (ALS) method. Take for example the deviation function from Equation 25:

$$\mathcal{D}(\theta, \mathbf{R}) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbf{W}_{ui} (\mathbf{R}_{ui} - \mathbf{S}_{ui})^2 + \sum_{t=1}^{T} \sum_{f=1}^{F} \lambda_{tf} ||\mathbf{S}^{(t,f)}||_F^2,$$

combined with the basic two-factor factorization model from Equation 7:

$$\mathbf{S} = \mathbf{S}^{(1,1)} \mathbf{S}^{(1,2)}.$$

As most deviation functions, this deviation function is non-convex in the parameters θ and has therefore multiple local optima. However, if one temporarily fixes the parameters in $\mathbf{S}^{(1,1)}$, it becomes convex in $\mathbf{S}^{(1,2)}$ and we can analytically find updated values for $\mathbf{S}^{(1,2)}$ that minimize this convex function and are therefore guaranteed to reduce $\mathcal{D}(\theta, \mathbf{R})$. Subsequently, one can temporarily fix the parameters in $\mathbf{S}^{(1,2)}$ and in the same way compute updated values for $\mathbf{S}^{(1,1)}$ that are also guaranteed to reduce $\mathcal{D}(\theta, \mathbf{R})$. One can keep alternating between fixing $\mathbf{S}^{(1,1)}$ and $\mathbf{S}^{(1,2)}$ until a convergence criterium of choice is met. Hu et al. [21], Pan et al. [37] and Pan and Scholz [36] give detailed descriptions of different ALS variations. The version by Hu et al. contains optimizations for the case in which missing preferences are uniformly weighted. Pan and Scholz [36] describe optimizations that apply to a wider range of weighting schemes. Finally, Pilaszy et al. propose to further speed-up the computation by only approximately solving each convex ALS-step [42].

Additionally, ALS has the advantages that it does not require the tuning of a learning rate, that it can benefit from linear algebra packages such as Intel MKL, and that it needs relatively few iterations to converge. Furthermore, when the basic two-factor factorization of Equation 7 is used, every row of $\mathbf{S}^{(1,1)}$ and every column of $\mathbf{S}^{(1,2)}$ can be updated independently of all other rows or columns, respectively, which makes it fairly easy to massively parallelize the computation of the factor matrices [77].

6.1.3 Bagging

The maximum margin based deviation function in Equation 33 cannot be solved with ALS because it contains the hinge loss. Rennie and Srebro propose a conjugate gradients method for minimizing this function [46]. However, this method suffers from similar problems as SGD, related to the high number of terms in the loss function. Therefore, Pan and Scholz [36] propose a bagging approach. The essence of their bagging approach is that they do not explicitly weight every user-item pair for which $\mathbf{R}_{ui} = 0$, but sample from all these pairs instead. They create multiple samples, and compute multiple different solutions $\widetilde{\mathbf{S}}$ corresponding to their samples. These computations are also performed with the conjugate gradients method. They are, however, much less intensive since they only consider a small sample of the many user-item pairs for which $\mathbf{R}_{ui} = 0$. The different solutions $\widetilde{\mathbf{S}}$ are finally aggregated by simply taking their average.

6.1.4 Coordinate Descent

When an item-based neighborhood model is used in combination with a squared error-based deviation function, the user factors are fixed by definition, and the problem resembles a single ALS step. However, imposing the constraints in Equation 11 complicates the minimization [34]. Therefore, Ning and Karypis adopt cyclic coordinate descent and soft thresholding [14] for SLIM.

6.2 Convex Minimization

Aiolli [3] proposed the convex deviation function in Equation 49 and indicates that it can be solved with any algorithm for convex optimization.

The analytically solvable deviation functions are also convex. Moreover, minimizing them is equivalent to computing all the similarities involved in the model. Most works assume a brute force computation of the similarities. However, Verstrepen [65] recently proposed two methods that are an order of magnitude faster than the brute force computation.

7. RATING BASED METHODS

Interest in collaborative filtering on binary, positive-only data only recently increased. The majority of existing collaborative filtering research assumes rating data. In this case, the feedback of user u about item i, $\mathbf{R}_{ui}$, is an integer between B_l and B_h, with B_l and B_h the most negative and positive feedback, respectively. The most typical example of such data was provided in the context of the Netflix Prize with $B_l = 1$ and $B_h = 5$.

Technically, our case of binary, positive-only data is just a special case of rating data with $B_l = B_h = 1$. However, collaborative filtering methods for rating data are in general built on the implicit assumption that $B_l < B_h$, i.e., that both positive and negative feedback is available. Since this negative feedback is not available in our problem setting, it is not surprising that, in general, methods for rating data generate poor or even nonsensical results [21; 37; 56].

k-NN methods for rating data, for example, often use the Pearson correlation coefficient as a similarity measure. The Pearson correlation coefficient between users u and v is given by

$$pcc(u,v) = \frac{\sum_{\mathbf{R}_{uj}, \mathbf{R}_{vj} > 0} (\mathbf{R}_{uj} - \overline{\mathbf{R}}_u)(\mathbf{R}_{vj} - \overline{\mathbf{R}}_v)}{\sqrt{\sum_{\mathbf{R}_{uj}, \mathbf{R}_{vj} > 0} (\mathbf{R}_{uj} - \overline{\mathbf{R}}_u)^2} \sqrt{\sum_{\mathbf{R}_{uj}, \mathbf{R}_{vj} > 0} (\mathbf{R}_{vj} - \overline{\mathbf{R}}_v)^2}},$$

with $\overline{\mathbf{R}}_u$ and $\overline{\mathbf{R}}_v$ the average rating of u and v respectively. In our setting, with binary, positive-only data however, $\mathbf{R}_{uj}$ and $\mathbf{R}_{vj}$ are by definition always one or zero. Consequently, $\overline{\mathbf{R}}_u$ and $\overline{\mathbf{R}}_v$ are always one. Therefore, the Pearson correlation is always zero or undefined (zero divided by zero), making it a useless similarity measure for binary, positive-only data. Even if we would hack it by omitting the terms for mean centering, $-\overline{\mathbf{R}}_u$ and $-\overline{\mathbf{R}}_v$, it is still useless since it would always be equal to either one or zero.

Furthermore, when computing the score of user u for item i, user(item)-based k-NN methods for rating data typically find the k users (items) that are most similar to u (i) *and that have rated i (have been rated by u)* [11; 23]. On binary, positive-only data, this approach results in the nonsensical

result that $\mathbf{S}_{ui} = 1$ for every (u, i)-pair.
The matrix factorization methods for rating data are in general also not applicable to binary, positive-only data. Take for example a basic loss function for matrix factorization on rating data:

$$\min_{\mathbf{S}^{(1)},\mathbf{S}^{(2)}} \sum_{\mathbf{R}_{ui}>0} \left(\mathbf{R}_{ui} - \mathbf{S}^{(1)}_{u\cdot}\mathbf{S}^{(2)}_{\cdot i}\right)^2 + \lambda \left(||\mathbf{S}^{(1)}_{u\cdot}||^2_F + ||\mathbf{S}^{(2)}_{\cdot i}||^2_F\right),$$

which for binary, positive-only data simplifies to

$$\min_{\mathbf{S}^{(1)},\mathbf{S}^{(2)}} \sum_{\mathbf{R}_{ui}=1} \left(1 - \mathbf{S}^{(1)}_{u\cdot}\mathbf{S}^{(2)}_{\cdot i}\right)^2 + \lambda \left(||\mathbf{S}^{(1)}_{u\cdot}||^2_F + ||\mathbf{S}^{(2)}_{\cdot i}||^2_F\right).$$

The squared error term of this loss function is minimized when the rows and columns of $\mathbf{S}^{(1)}$ and $\mathbf{S}^{(2)}$, respectively, are all the same unit vector. This is obviously a nonsensical solution.
The matrix factorization method for rating data that uses singular value decomposition to factorize $\mathbf{R}$ also considers the entries where $\mathbf{R}_{ui} = 0$ and does not suffer from the above problem [49; 8]. Although this method does not result in nonsensical results, the performance has been shown inferior to methods specialized for binary, positive-only data [34; 56; 57].
In summary, although we cannot exclude the possibility that there exists a method for rating data that does perform well on binary, positive-only data, in general this is clearly not the case.

8. CONCLUSIONS

We have presented a comprehensive survey of collaborative filtering methods for binary, positive-only data. Its backbone is an innovative unified matrix factorization perspective on collaborative filtering methods, also those that are typically not associated with matrix factorization models such as nearest neighbors methods and association rule mining-based methods. From this perspective, a collaborative filtering algorithm consists of three building blocks: a matrix factorization model, a deviation function and a numerical minimization algorithm. By comparing methods along these three dimensions, we were able to highlight surprising commonalities and key differences.
An interesting direction for future work is to survey certain aspects that were not included in the scope of this survey. Examples are surveying the different strategies to deal with cold-start problems that are applicable to binary, positive-only data; and comparing the applicability of models and deviation functions for recomputation of models in real time upon receiving novel feedback.

9. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. 17(6):734–749, 2005.

[2] F. Aiolli. Efficient top-n recommendation for very large scale binary rated datasets. In *Proc. of the 7th ACM Conf. on Recommender Systems*, pages 273–280. ACM, 2013.

[3] F. Aiolli. Convex auc optimization for top-n recommendation with implicit feedback. In *Proc. of the 8th ACM Conf. on Recommender Systems*, pages 293–296. ACM, 2014.

[4] A. Bellogin, J. Wang, and P. Castells. Text retrieval methods for item ranking in collaborative filtering. In P. Clough, C. Foley, C. Gurrin, G. Jones, W. Kraaij, H. Lee, and V. Mudoch, editors, *Advances in Information Retrieval*, volume 6611, pages 301–306. Springer Berlin Heidelberg, 2011.

[5] C. M. Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *The Journal of machine Learning research*, 3:993–1022, 2003.

[7] E. Christakopoulou and G. Karypis. Hoslim: Higher-order sparse linear method for top-n recommender systems. In *Advances in Knowledge Discovery and Data Mining*, pages 38–49. Springer, 2014.

[8] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of the fourth ACM Conf. on Recommender Systems*, pages 39–46. ACM, 2010.

[9] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[10] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[11] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[12] C. Dhanjal, R. Gaudel, and S. Clémençon. Collaborative filtering with localised ranking. In *Proc. of the 29th AAAI Conf. on Artificial Intelligence*, 2015.

[13] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[14] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[15] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proc. of the 17th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 69–77, 2011.

[16] P. Gopalan, J. Hofman, and D. Blei. Scalable recommendation with hierarchical poisson factorization. In *Proc. of the 31st Conf. Annual Conf. on Uncertainty in Artificial Intelligence (UAI-15). AUAI Press*, 2015.

[17] P. Gopalan, F. J. Ruiz, R. Ranganath, and D. M. Blei. Bayesian nonparametric poisson factorization for recommendation systems. *Artificial Intelligence and Statistics (AISTATS)*, 33:275–283, 2014.

[18] M. Gori, A. Pucci, V. Roma, and I. Siena. Itemrank: A random-walk based scoring algorithm for recommender engines. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence*, volume 7, pages 2766–2771, 2007.

[19] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115, 2004.

[20] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence*, pages 688–693, 1999.

[21] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. of the Eighth IEEE Int. Conf. on Data Mining*, pages 263–272. IEEE, 2008.

[22] Z. Huang, X. Li, and H. Chen. Link prediction approach to collaborative filtering. In *Proc. of the 5th ACM/IEEE-CS joint Conf. on Digital libraries*, pages 141–142. ACM, 2005.

[23] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender systems: an introduction*. Cambridge University Press, 2010.

[24] C. Johnson. Logistic matrix factorization for implicit feedback data. In *Workshop on Distributed Machine Learning and Matrix Computations at the Twenty-eighth Annual Conf. on Neural Information Processing Systems (NIPS)*, 2014.

[25] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[26] S. Kabbur and G. Karypis. Nlmf: Nonlinear matrix factorization methods for top-n recommender systems. In *Workshop Proc. of the IEEE Int. Conf. on Data Mining*, pages 167–174. IEEE, 2014.

[27] S. Kabbur, X. Ning, and G. Karypis. Fism: factored item similarity models for top-n recommender systems. In *Proc. of the 19th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 659–667. ACM, 2013.

[28] N. Koenigstein, N. Nice, U. Paquet, and N. Schleyen. The xbox recommender system. In *Proc. of the sixth ACM Conf. on Recommender Systems*, pages 281–284. ACM, 2012.

[29] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[30] W. Lin, S. A. Alvarez, and C. Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data mining and knowledge discovery*, 6(1):83–105, 2002.

[31] G. V. Menezes, J. M. Almeida, F. Belém, M. A. Gonçalves, A. Lacerda, E. S. De Moura, G. L. Pappa, A. Veloso, and N. Ziviani. Demand-driven tag recommendation. In *Machine Learning and Knowledge Discovery in Databases*, pages 402–417. Springer, 2010.

[32] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.

[33] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Effective personalization based on association rule discovery from web usage data. In *Proc. of the 3rd Int. workshop on Web information and data management*, pages 9–15. ACM, 2001.

[34] X. Ning and G. Karypis. Slim: Sparse linear methods for top-n recommender systems. In *Proc. of the 11th IEEE Int. Conf. on Data Mining*, pages 497–506. IEEE, 2011.

[35] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[36] R. Pan and M. Scholz. Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering. In *Proc. of the 15th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 667–676. ACM, 2009.

[37] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Proc. of the Eighth IEEE Int. Conf. on Data Mining*, pages 502–511. IEEE, 2008.

[38] W. Pan and L. Chen. Cofiset: Collaborative filtering via learning pairwise preferences over item-sets. In *Proc. of the 13th SIAM Int. Conf. on Data Mining*, pages 180–188, 2013.

[39] W. Pan and L. Chen. Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *Proc. of the Twenty-Third Int. joint Conf. on Artificial Intelligence*, pages 2691–2697. AAAI Press, 2013.

[40] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *Proc. of the 22nd Int. Conf. on WWW*, pages 999–1008, 2013.

[41] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proc. of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[42] I. Pilászy, D. Zibriczky, and D. Tikk. Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proc. of the fourth ACM Conf. on Recommender Systems*, pages 71–78. ACM, 2010.

[43] B. Recht, C. Re, S. Wright, and F. Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24*, pages 693–701. 2011.

[44] S. Rendle and C. Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *Proc. of the 7th ACM Int. Conf. on Web search and data mining*, pages 273–282. ACM, 2014.

[45] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proc. of the Twenty-Fifth Conf. on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[46] J. D. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proc. of the 22nd Int. Conf. on Machine learning*, pages 713–719. ACM, 2005.

[47] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor. *Recommender Systems Handbook*. Springer, Boston, MA, 2011.

[48] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proc. of the 2nd ACM Conf. on Electronic commerce*, pages 158–167. ACM, 2000.

[49] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[50] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the 10th Int. Conf. on WWW*, pages 285–295. ACM, 2001.

[51] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, and N. Oliver. Tfmap: Optimizing map for top-n context-aware recommendation. In *Proc. of the 35th Int. ACM SIGIR Conf. on Research and development in information retrieval*, pages 155–164. ACM, 2012.

[52] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic. Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proc. of the sixth ACM Conf. on Recommender Systems*, pages 139–146. ACM, 2012.

[53] Y. Shi, M. Larson, and A. Hanjalic. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, 47(1):3, 2014.

[54] B. Sigurbjörnsson and R. Van Zwol. Flickr tag recommendation based on collective knowledge. In *The 17th Int. Conf. on WWW*, pages 327–336. ACM, 2008.

[55] V. Sindhwani, S. S. Bucak, J. Hu, and A. Mojsilovic. One-class matrix completion with low-density factorizations. In *Proc. of the 10th IEEE Int. Conf. on Data Mining*, pages 1055–1060. IEEE, 2010.

[56] H. Steck. Training and testing of recommender systems on data missing not at random. In *Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 713–722. ACM, 2010.

[57] H. Steck. Item popularity and recommendation accuracy. In *Proc. of the fifth ACM Conf. on Recommender Systems*, pages 125–132. ACM, 2011.

[58] H. Steck. Gaussian ranking by matrix factorization. In *Proc. of the 9th ACM Conf. on Recommender Systems*, pages 115–122. ACM, 2015.

[59] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

[60] P. Symeonidis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos. Nearest-biclusters collaborative filtering based on constant and coherent values. *Inf. Retr.*, 11(1):51–75, 2008.

[61] G. Takács and D. Tikk. Alternating least squares for personalized ranking. In *Proc. of the sixth ACM Conf. on Recommender Systems*, pages 83–90. ACM, 2012.

[62] A. Töscher and M. Jahrer. Collaborative filtering ensemble for ranking. *Journal of Machine Learning Research W&CP: Proc. of KDD Cup 2011*, 18:61–74, 2012.

[63] L. H. Ungar and D. P. Foster. Clustering methods for collaborative filtering. In *AAAI workshop on recommendation systems*, volume 1, pages 114–129, 1998.

[64] M. van Leeuwen and D. Puspitaningrum. Improving tag recommendation using few associations. In *Advances in Intelligent Data Analysis XI*, pages 184–194. Springer, 2012.

[65] K. Verstrepen. *Collaborative Filtering with Binary, Positive-Only Data*. PhD thesis, University of Antwerp, 2015.

[66] K. Verstrepen and B. Goethals. Unifying nearest neighbors collaborative filtering. In *Proc. of the 8th ACM Conf. on Recommender Systems*, pages 177–184. ACM, 2014.

[67] K. Verstrepen and B. Goethals. Top-n recommendation for shared accounts. In *Proc. of the 9th ACM Conf. on Recommender Systems*, pages 59–66. ACM, 2015.

[68] J. Wang, A. P. De Vries, and M. J. Reinders. A user-item relevance model for log-based collaborative filtering. In *Advances in Information Retrieval*, pages 37–48. Springer, 2006.

[69] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proc. of the 22th Interantional Joint Conf. on Artifical Intelligence*, volume 11, pages 2764–2770, 2011.

[70] J. Weston, R. J. Weiss, and H. Yee. Nonlinear latent factorization by embedding multiple user interests. In *Proc. of the 7th ACM Conf. on Recommender Systems*, pages 65–68. ACM, 2013.

[71] J. Weston, H. Yee, and R. J. Weiss. Learning to rank recommendations with the k-order statistic loss. In *Proc. of the 7th ACM Conf. on Recommender Systems*, pages 245–248. ACM, 2013.

[72] X. Yang, Y. Guo, Y. Liu, and H. Steck. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1–10, 2014.

[73] Y. Yao, H. Tong, G. Yan, F. Xu, X. Zhang, B. K. Szymanski, and J. Lu. Dual-regularized one-class collaborative filtering. In *Proc. of the 23rd ACM Int. Conf. on Information and Knowledge Management*, pages 759–768. ACM, 2014.

[74] J. Yin, L. Gao, and Z. M. Zhang. *Machine Learning and Knowledge Discovery in Databases: European Conf., ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proc., Part III*, chapter Scalable Nonnegative Matrix Factorization with Block-wise Updates, pages 337–352. 2014.

[75] W. Zhang, T. Chen, J. Wang, and Y. Yu. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proc. of the 36th Int. ACM SIGIR Conf. on Research and development in information retrieval*, pages 785–788. ACM, 2013.

[76] H. Zhong, W. Pan, C. Xu, Z. Yin, and Z. Ming. Adaptive pairwise preference learning for collaborative recommendation with implicit feedbacks. In *Proc. of the 23rd ACM Int. Conf. on Conf. on Information and Knowledge Management*, pages 1999–2002. ACM, 2014.

[77] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Algorithmic Aspects in Information and Management*, pages 337–348. Springer, 2008.

[78] Y. Zhuang, W.-S. Chin, Y.-C. Juan, and C.-J. Lin. A fast parallel sgd for matrix factorization in shared memory systems. In *Proc. of the 7th ACM Conf. on Recommender Systems*, pages 249–256, 2013.

Fake News Detection on Social Media: A Data Mining Perspective

Kai Shu[†], Amy Sliva[‡], Suhang Wang[†], Jiliang Tang [♮], and Huan Liu[†]
[†]Computer Science & Engineering, Arizona State University, Tempe, AZ, USA
[‡]Charles River Analytics, Cambridge, MA, USA
[♮]Computer Science & Engineering, Michigan State University, East Lansing, MI, USA
[†]{kai.shu,suhang.wang,huan.liu}@asu.edu,
[‡]asliva@cra.com, [♮] tangjili@msu.edu

ABSTRACT

Social media for news consumption is a double-edged sword. On the one hand, its low cost, easy access, and rapid dissemination of information lead people to seek out and consume news from social media. On the other hand, it enables the wide spread of "fake news", i.e., low quality news with intentionally false information. The extensive spread of fake news has the potential for extremely negative impacts on individuals and society. Therefore, fake news detection on social media has recently become an emerging research that is attracting tremendous attention. Fake news detection on social media presents unique characteristics and challenges that make existing detection algorithms from traditional news media ineffective or not applicable. First, fake news is intentionally written to mislead readers to believe false information, which makes it difficult and nontrivial to detect based on news content; therefore, we need to include auxiliary information, such as user social engagements on social media, to help make a determination. Second, exploiting this auxiliary information is challenging in and of itself as users' social engagements with fake news produce data that is big, incomplete, unstructured, and noisy. Because the issue of fake news detection on social media is both challenging and relevant, we conducted this survey to further facilitate research on the problem. In this survey, we present a comprehensive review of detecting fake news on social media, including fake news characterizations on psychology and social theories, existing algorithms from a data mining perspective, evaluation metrics and representative datasets. We also discuss related research areas, open problems, and future research directions for fake news detection on social media.

1. INTRODUCTION

As an increasing amount of our lives is spent interacting online through social media platforms, more and more people tend to seek out and consume news from social media rather than traditional news organizations. The reasons for this change in consumption behaviors are inherent in the nature of these social media platforms: (i) it is often more timely and less expensive to consume news on social media compared with traditional news media, such as newspapers or television; and (ii) it is easier to further share, comment on, and discuss the news with friends or other readers on social media. For example, 62 percent of U.S. adults get news on social media in 2016, while in 2012, only 49 percent reported seeing news on social media[1]. It was also found that social media now outperforms television as the major news source[2]. Despite the advantages provided by social media, the quality of news on social media is lower than traditional news organizations. However, because it is cheap to provide news online and much faster and easier to disseminate through social media, large volumes of fake news, i.e., those news articles with intentionally false information, are produced online for a variety of purposes, such as financial and political gain. It was estimated that over 1 million tweets are related to fake news "Pizzagate"[3] by the end of the presidential election. Given the prevalence of this new phenomenon, "Fake news" was even named the word of the year by the Macquarie dictionary in 2016.

The extensive spread of fake news can have a serious negative impact on individuals and society. First, fake news can break the authenticity balance of the news ecosystem. For example, it is evident that the most popular fake news was even more widely spread on Facebook than the most popular authentic mainstream news during the U.S. 2016 president election[4]. Second, fake news intentionally persuades consumers to accept biased or false beliefs. Fake news is usually manipulated by propagandists to convey political messages or influence. For example, some report shows that Russia has created fake accounts and social bots to spread false stories[5]. Third, fake news changes the way people interpret and respond to real news. For example, some fake news was just created to trigger people's distrust and make them confused, impeding their abilities to differentiate what is true from what is not[6]. To help mitigate the negative effects caused by fake news–both to benefit the public and the news ecosystem–It's critical that we develop methods to automatically detect fake news on social media.

[1]http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/
[2]http://www.bbc.com/news/uk-36528256
[3]https://en.wikipedia.org/wiki/Pizzagate_conspiracy_theory
[4]https://www.buzzfeed.com/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook?utm_term=.nrg0WA1VP0#.gjJyKapW5y
[5]http://time.com/4783932/inside-russia-social-media-war-america/
[6]https://www.nytimes.com/2016/11/28/opinion/fake-news-and-the-internet-shell-game.html?_r=0

Detecting fake news on social media poses several new and challenging research problems. Though fake news itself is not a new problem–nations or groups have been using the news media to execute propaganda or influence operations for centuries–the rise of web-generated news on social media makes fake news a more powerful force that challenges traditional journalistic norms. There are several characteristics of this problem that make it uniquely challenging for automated detection. First, fake news is intentionally written to mislead readers, which makes it nontrivial to detect simply based on news content. The content of fake news is rather diverse in terms of topics, styles and media platforms, and fake news attempts to distort truth with diverse linguistic styles while simultaneously mocking true news. For example, fake news may cite true evidence within the incorrect context to support a non-factual claim [22]. Thus, existing hand-crafted and data-specific textual features are generally not sufficient for fake news detection. Other auxiliary information must also be applied to improve detection, such as knowledge base and user social engagements. Second, exploiting this auxiliary information actually leads to another critical challenge: the quality of the data itself. Fake news is usually related to newly emerging, time-critical events, which may not have been properly verified by existing knowledge bases due to the lack of corroborating evidence or claims. In addition, users' social engagements with fake news produce data that is big, incomplete, unstructured, and noisy [79]. Effective methods to differentiate credible users, extract useful post features and exploit network interactions are an open area of research and need further investigations.

In this article, we present an overview of fake news detection and discuss promising research directions. The key motivations of this survey are summarized as follows:

- Fake news on social media has been occurring for several years; however, there is no agreed upon definition of the term "fake news". To better guide the future directions of fake news detection research, appropriate clarifications are necessary.
- Social media has proved to be a powerful source for fake news dissemination. There are some emerging patterns that can be utilized for fake news detection in social media. A review on existing fake news detection methods under various social media scenarios can provide a basic understanding on the state-of-the-art fake news detection methods.
- Fake news detection on social media is still in the early age of development, and there are still many challenging issues that need further investigations. It is necessary to discuss potential research directions that can improve fake news detection and mitigation capabilities.

To facilitate research in fake news detection on social media, in this survey we will review two aspects of the fake news detection problem: *characterization* and *detection*. As shown in Figure 1, we will first describe the background of the fake news detection problem using theories and properties from psychology and social studies; then we present the detection approaches. Our major contributions of this survey are summarized as follows:

- We discuss the narrow and broad definitions of fake news that cover most existing definitions in the literature and further present the unique characteristics of fake news on social media and its implications compared with the traditional media;
- We give an overview of existing fake news detection methods with a principled way to group representative methods into different categories; and
- We discuss several open issues and provide future directions of fake news detection in social media.

The remainder of this survey is organized as follows. In Section 2, we present the definition of fake news and characterize it by comparing different theories and properties in both traditional and social media. In Section 3, we continue to formally define the fake news detection problem and summarize the methods to detect fake news. In Section 4, we discuss the datasets and evaluation metrics used by existing methods. We briefly introduce areas related to fake news detection on social media in Section 5. Finally, we discuss the open issues and future directions in Section 6 and conclude this survey in Section 7.

2. FAKE NEWS CHARACTERIZATION

In this section, we introduce the basic social and psychological theories related to fake news and discuss more advanced patterns introduced by social media. Specifically, we first discuss various definitions of fake news and differentiate related concepts that are usually misunderstood as fake news. We then describe different aspects of fake news on traditional media and the new patterns found on social media.

2.1 Definitions of Fake News

Fake news has existed for a very long time, nearly the same amount of time as news began to circulate widely after the printing press was invented in 1439[7]. However, there is no agreed definition of the term "fake news". Therefore, we first discuss and compare some widely used definitions of fake news in the existing literature, and provide our definition of fake news that will be used for the remainder of this survey.

A narrow definition of fake news is news articles that are intentionally and verifiably false and could mislead readers [2]. There are two key features of this definition: *authenticity* and *intent*. First, fake news includes false information that can be verified as such. Second, fake news is created with dishonest intention to mislead consumers. This definition has been widely adopted in recent studies [57; 17; 62; 41].

Broader definitions of fake news focus on the either authenticity or intent of the news content. Some papers regard satire news as fake news since the contents are false even though satire is often entertainment-oriented and reveals its own deceptiveness to the consumers [67; 4; 37; 9]. Other literature directly treats deceptive news as fake news [66], which includes serious fabrications, hoaxes, and satires.

In this article, we use the narrow definition of fake news. Formally, we state this definition as follows,

Definition 1 (Fake News) *Fake news is a news article that is intentionally and verifiably false.*

[7]http://www.politico.com/magazine/story/2016/12/fake-news-history-long-violent-214535

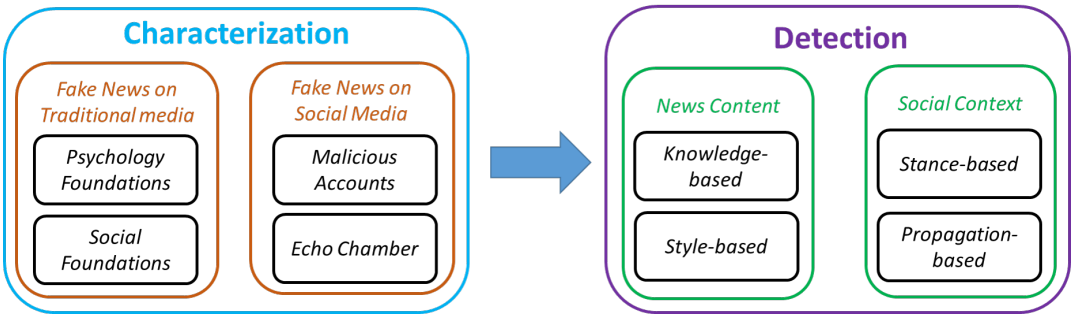

Figure 1: Fake news on social media: from characterization to detection.

The reasons for choosing this narrow definition are threefolds. First, the underlying intent of fake news provides both theoretical and practical value that enables a deeper understanding and analysis of this topic. Second, any techniques for truth verification that apply to the narrow conception of fake news can also be applied to under the broader definition. Third, this definition is able to eliminate the ambiguities between fake news and related concepts that are not considered in this article. The following concepts are not fake news according to our definition: (1) satire news with proper context, which has no intent to mislead or deceive consumers and is unlikely to be mis-perceived as factual; (2) rumors that did not originate from news events; (3) conspiracy theories, which are difficult verify as true or false; (4) misinformation that is created unintentionally; and (5) hoaxes that are only motivated by fun or to scam targeted individuals.

2.2 Fake News on Traditional News Media

Fake news itself is not a new problem. The media ecology of fake news has been changing over time from newsprint to radio/television and, recently, online news and social media. We denote "traditional fake news" as the fake news problem before social media had important effects on its production and dissemination. Next, we will describe several psychological and social science foundations that describe the impact of fake news at both the individual and social information ecosystem levels.

Psychological Foundations of Fake News. Humans are naturally not very good at differentiating between real and fake news. There are several psychological and cognitive theories that can explain this phenomenon and the influential power of fake news. Traditional fake news mainly targets consumers by exploiting their *individual vulnerabilities*. There are two major factors which make consumers naturally vulnerable to fake news: (i) *Naïve Realism*: consumers tend to believe that their perceptions of reality are the only accurate views, while others who disagree are regarded as uninformed, irrational, or biased [92]; and (ii) *Confirmation Bias*: consumers prefer to receive information that confirms their existing views [58]. Due to these cognitive biases inherent in human nature, fake news can often be perceived as real by consumers. Moreover, once the misperception is formed, it is very hard to correct it. Psychology studies shows that correction of false information (e.g., fake news) by the presentation of true, factual information is not only unhelpful to reduce misperceptions, but sometimes may even increase the misperceptions, especially among ideological groups [59].

Social Foundations of the Fake News Ecosystem. Considering the entire news consumption ecosystem, we can also describe some of the social dynamics that contribute to the proliferation of fake news. Prospect theory describes decision making as a process by which people make choices based on the relative gains and losses as compared to their current state [39; 81]. This desire for maximizing the reward of a decision applies to social gains as well, for instance, continued acceptance by others in a user's immediate social network. As described by social identity theory [76; 77] and normative influence theory [3; 40], this preference for social acceptance and affirmation is essential to a person's identity and self-esteem, making users likely to choose "socially safe" options when consuming and disseminating news information, following the norms established in the community even if the news being shared is fake news.

This rational theory of fake news interactions can be modeled from an economic game theoretical perspective [26] by formulating the news generation and consumption cycle as a two-player strategy game. For explaining fake news, we assume there are two kinds of key players in the information ecosystem: *publisher* and *consumer*. The process of news publishing is modeled as a mapping from original signal s to resultant news report a with an effect of distortion bias b, i.e., $s \xrightarrow{b} a$, where $b = [-1, 0, 1]$ indicates $[left, no, right]$ biases take effects on news publishing process. Intuitively, this is capturing the degree to which a news article may be biased or distorted to produce fake news. The utility for the publisher stems from two perspectives: (i) *short-term utility:* the incentive to maximize profit, which is positively correlated with the number of consumers reached; (ii) *long-term utility:* their reputation in terms of news authenticity. Utility of consumers consists of two parts: (i) *information utility*: obtaining true and unbiased information (usually extra investment cost needed); (ii) *psychology utility*: receiving news that satisfies their prior opinions and social needs, e.g., confirmation bias and prospect theory. Both publisher and consumer try to maximize their overall utilities in this strategy game of the news consumption process. We can capture the fact that fake news happens when the *short-term utility* dominates a publisher's overall utility and *psychology utility* dominates the consumer's overall utility, and an equilibrium is maintained. This explains the social dynamics that lead to an information ecosystem where fake news can thrive.

2.3 Fake News on Social Media

In this subsection, we will discuss some unique characteristics of fake news on social media. Specifically, we will highlight the key features of fake news that are enabled by social media. Note that the aforementioned characteristics of traditional fake news are also applicable to social media.

Malicious Accounts on Social Media for Propaganda. While many users on social media are legitimate, social media users may also be malicious, and in some cases are not even real humans. The low cost of creating social media accounts also encourages malicious user accounts, such as social bots, cyborg users, and trolls. A social bot refers to a social media account that is controlled by a computer algorithm to automatically produce content and interact with humans (or other bot users) on social media [23]. Social bots can become malicious entities designed specifically with the purpose to do harm, such as manipulating and spreading fake news on social media. Studies shows that social bots distorted the 2016 U.S. presidential election online discussions on a large scale [6], and that around 19 million bot accounts tweeted in support of either Trump or Clinton in the week leading up to election day[8]. Trolls, real human users who aim to disrupt online communities and provoke consumers into an emotional response, are also playing an important role in spreading fake news on social media. For example, evidence suggests that there were 1,000 paid Russian trolls spreading fake news on Hillary Clinton[9]. Trolling behaviors are highly affected by people's mood and the context of online discussions, which enables the easy dissemination of fake news among otherwise "normal" online communities [14]. The effect of trolling is to trigger people's inner negative emotions, such as anger and fear, resulting in doubt, distrust, and irrational behavior. Finally, cyborg users can spread fake news in a way that blends automated activities with human input. Usually cyborg accounts are registered by human as a camouflage and set automated programs to perform activities in social media. The easy switch of functionalities between human and bot offers cyborg users unique opportunities to spread fake news [15]. In a nutshell, these highly active and partisan malicious accounts on social media become the powerful sources and proliferation of fake news.

Echo Chamber Effect. Social media provides a new paradigm of information creation and consumption for users. The information seeking and consumption process are changing from a mediated form (e.g., by journalists) to a more disinter-mediated way [19]. Consumers are selectively exposed to certain kinds of news because of the way news feed appear on their homepage in social media, amplifying the psychological challenges to dispelling fake news identified above. For example, users on Facebook always follow like-minded people and thus receive news that promote their favored existing narratives [65]. Therefore, users on social media tend to form groups containing like-minded people where they then polarize their opinions, resulting in an *echo chamber* effect. The echo chamber effect facilitates the process by which people consume and believe fake news due to the following psychological factors [60]: (1) *social credibility*, which means people are more likely to perceive a source as credible if others perceive the source is credible, especially when there is not enough information available to access the truthfulness of the source; and (2) *frequency heuristic*, which means that consumers may naturally favor information they hear frequently, even if it is fake news. Studies have shown that increased exposure to an idea is enough to generate a positive opinion of it [100; 101], and in echo chambers, users continue to share and consume the same information. As a result, this echo chamber effect creates segmented, homogeneous communities with a very limited information ecosystem. Research shows that the homogeneous communities become the primary driver of information diffusion that further strengthens polarization [18].

[8]http://comprop.oii.ox.ac.uk/2016/11/18/resource-for-understanding-political-bots/

[9]http://www.huffingtonpost.com/entry/russian-trolls-fake-news_us_58dde6bae4b08194e3b8d5c4

3. FAKE NEWS DETECTION

In the previous section, we introduced the conceptual characterization of traditional fake news and fake news in social media. Based on this characterization, we further explore the problem definition and proposed approaches for fake news detection.

3.1 Problem Definition

In this subsection, we present the details of mathematical formulation of fake news detection on social media. Specifically, we will introduce the definition of key components of fake news and then present the formal definition of fake news detection. The basic notations are defined below,

- Let a refer to a *News Article*. It consists of two major components: *Publisher* and *Content*. Publisher $\vec{p_a}$ includes a set of profile features to describe the original author, such as name, domain, age, among other attributes. Content $\vec{c_a}$ consists of a set of attributes that represent the news article and includes headline, text, image, etc.
- We also define *Social News Engagements* as a set of tuples $\mathcal{E} = \{e_{it}\}$ to represent the process of how news spread over time among n users $\mathcal{U} = \{u_1, u_2, ..., u_n\}$ and their corresponding posts $\mathcal{P} = \{p_1, p_2, ..., p_n\}$ on social media regarding news article a. Each engagement $e_{it} = \{u_i, p_i, t\}$ represents that a user u_i spreads news article a using p_i at time t. Note that we set $t = Null$ if the article a does not have any engagement yet and thus u_i represents the publisher.

Definition 2 (Fake News Detection) *Given the social news engagements $\mathcal{E}$ among n users for news article a, the task of fake news detection is to predict whether the news article a is a fake news piece or not, i.e., $\mathcal{F} : \mathcal{E} \rightarrow \{0, 1\}$ such that,*

$$\mathcal{F}(a) = \begin{cases} 1, & \text{if } a \text{ is a piece of fake news,} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where $\mathcal{F}$ is the prediction function we want to learn.

Note that we define fake news detection as a binary classification problem for the following reason: fake news is essentially a *distortion bias* on information manipulated by the publisher. According to previous research about media bias

theory [26], distortion bias is usually modeled as a binary classification problem.

Next, we propose a general data mining framework for fake news detection which includes two phases: (i) feature extraction and (ii) model construction. The feature extraction phase aims to represent news content and related auxiliary information in a formal mathematical structure, and model construction phase further builds machine learning models to better differentiate fake news and real news based on the feature representations.

3.2 Feature Extraction

Fake news detection on traditional news media mainly relies on news content, while in social media, extra social context auxiliary information can be used to as additional information to help detect fake news. Thus, we will present the details of how to extract and represent useful features from *news content* and *social context*.

3.2.1 News Content Features

News content features $\vec{c_a}$ describe the meta information related to a piece of news. A list of representative news content attributes are listed below:

- **Source:** Author or publisher of the news article
- **Headline:** Short title text that aims to catch the attention of readers and describes the main topic of the article
- **Body Text:** Main text that elaborates the details of the news story; there is usually a major claim that is specifically highlighted and that shapes the angle of the publisher
- **Image/Video:** Part of the body content of a news article that provides visual cues to frame the story

Based on these raw content attributes, different kinds of feature representations can be built to extract discriminative characteristics of fake news. Typically, the news content we are looking at will mostly be *linguistic-based* and *visual-based*, described in more detail below.

Linguistic-based: Since fake news pieces are intentionally created for financial or political gain rather than to report objective claims, they often contain opinionated and inflammatory language, crafted as "clickbait" (i.e., to entice users to click on the link to read the full article) or to incite confusion [13]. Thus, it is reasonable to exploit linguistic features that capture the different writing styles and sensational headlines to detect fake news. Linguistic-based features are extracted from the text content in terms of document organizations from different levels, such as characters, words, sentences, and documents. In order to capture the different aspects of fake news and real news, existing work utilized both common linguistic features and domain-specific linguistic features. Common linguistic features are often used to represent documents for various tasks in natural language processing. Typical common linguistic features are: (i) *lexical features*, including character-level and word-level features, such as total words, characters per word, frequency of large words, and unique words; (ii) *syntactic features*, including sentence-level features, such as frequency of function words and phrases (i.e., "n-grams" and bag-of-words approaches [24]) or punctuation and parts-of-speech (POS) tagging. Domain-specific linguistic features, which are specifically aligned to news domain, such as quoted words, external links, number of graphs, and the average length of graphs, etc [62]. Moreover, other features can be specifically designed to capture the deceptive cues in writing styles to differentiate fake news, such as lying-detection features [1].

Visual-based: Visual cues have been shown to be an important manipulator for fake news propaganda[10]. As we have characterized, fake news exploits the individual vulnerabilities of people and thus often relies on sensational or even fake images to provoke anger or other emotional response of consumers. Visual-based features are extracted from visual elements (e.g. images and videos) to capture the different characteristics for fake news. Faking images were identified based on various user-level and tweet-level hand-crafted features using classification framework [28]. Recently, various *visual* and *statistical* features has been extracted for news verification [38]. Visual features include clarity score, coherence score, similarity distribution histogram, diversity score, and clustering score. Statistical features include count, image ratio, multi-image ratio, hot image ratio, long image ratio, etc.

3.2.2 Social Context Features

In addition to features related directly to the content of the news articles, additional social context features can also be derived from the user-driven social engagements of news consumption on social media platform. Social engagements represent the news proliferation process over time, which provides useful auxiliary information to infer the veracity of news articles. Note that few papers exist in the literature that detect fake news using social context features. However, because we believe this is a critical aspect of successful fake news detection, we introduce a set of common features utilized in similar research areas, such as rumor veracity classification on social media. Generally, there are three major aspects of the social media context that we want to represent: users, generated posts, and networks. Below, we investigate how we can extract and represent social context features from these three aspects to support fake news detection.

User-based: As we mentioned in Section 2.3, fake news pieces are likely to be created and spread by non-human accounts, such as social bots or cyborgs. Thus, capturing users' profiles and characteristics by user-based features can provide useful information for fake news detection. User-based features represent the characteristics of those users who have interactions with the news on social media. These features can be categorized across different levels: *individual level* and *group level*. Individual level features are extracted to infer the credibility and reliability for each user using various aspects of user demographics, such as registration age, number of followers/followees, number of tweets the user has authored, etc [11]. Group level user features capture overall characteristics of groups of users related to the news [99]. The assumption is that the spreaders of fake news

[10]https://www.wired.com/2016/12/photos-fuel-spread-fake-news/

and real news may form different communities with unique characteristics that can be depicted by group level features. Commonly used group level features come from aggregating (e.g., averaging and weighting) individual level features, such as 'percentage of verified users' and 'average number of followers' [49; 42].

Post-based: People express their emotions or opinions towards fake news through social media posts, such as skeptical opinions, sensational reactions, etc. Thus, it is reasonable to extract post-based features to help find potential fake news via reactions from the general public as expressed in posts. Post-based features focus on identifying useful information to infer the veracity of news from various aspects of relevant social media posts. These features can be categorized as *post level*, *group level*, and *temporal level*. Post level features generate feature values for each post. The aforementioned linguistic-based features and some embedding approaches [69] for news content can also be applied for each post. Specifically, there are unique features for posts that represent the social response from general public, such as *stance*, *topic*, and *credibility*. Stance features (or viewpoints) indicate the users' opinions towards the news, such as supporting, denying, etc [37]. Topic features can be extracted using topic models, such as latent Dirichlet allocation (LDA) [49]. Credibility features for posts assess the degree of reliability [11]. Group level features aim to aggregate the feature values for all relevant posts for specific news articles by using "wisdom of crowds". For example, the average credibility scores are used to evaluate the credibility of news [37]. A more comprehensive list of group-level post features can also be found in [11]. Temporal level features consider the temporal variations of post level feature values [49]. Unsupervised embedding methods, such as recurrent neural network (RNN), are utilized to capture the changes in posts over time [69; 48]. Based on the shape of this time series for various metrics of relevant posts (e.g, number of posts), mathematical features can be computed, such as SpikeM parameters [42].

Network-based: Users form different networks on social media in terms of interests, topics, and relations. As mentioned before, fake news dissemination processes tend to form an echo chamber cycle, highlighting the value of extracting network-based features to represent these types of network patterns for fake news detection. Network-based features are extracted via constructing specific networks among the users who published related social media posts. Different types of networks can be constructed. The *stance network* can be built with nodes indicating all the tweets relevant to the news and the edge indicating the weights of similarity of stances [37; 75]. Another type of network is the *co-occurrence network*, which is built based on the user engagements by counting whether those users write posts relevant to the same news articles [69]. In addition, the *friendship network* indicates the following/followee structure of users who post related tweets [42]. An extension of this friendship network is the *diffusion network*, which tracks the trajectory of the spread of news [42], where nodes represent the users and edges represent the information diffusion paths among them. That is, a diffusion path between two users u_i and u_j exists if and only if (1) u_j follows u_i, and (2) u_j posts about a given news only after u_i does so. After these networks are properly built, existing network metrics can be applied as feature representations. For example, degree and clustering coefficient have been used to characterize the diffusion network [42] and friendship network [42]. Other approaches learn the latent node embedding features by using SVD [69] or network propagation algorithms [37].

3.3 Model Construction

In the previous section, we introduced features extracted from different sources, i.e., news content and social context, for fake news detection. In this section, we discuss the details of the model construction process for several existing approaches. Specifically we categorize existing methods based on their main input sources as: *News Content Models* and *Social Context Models*.

3.3.1 News Content Models

In this subsection, we focus on news content models, which mainly rely on news content features and existing factual sources to classify fake news. Specifically, existing approaches can be categorized as *Knowledge-based* and *Style-based*.

Knowledge-based: Since fake news attempts to spread false claims in news content, the most straightforward means of detecting it is to check the truthfulness of major claims in a news article to decide the news veracity. Knowledge-based approaches aim to use external sources to fact-check proposed claims in news content. The goal of fact-checking is to assign a truth value to a claim in a particular context [83]. Fact-checking has attracted increasing attention, and many efforts have been made to develop a feasible automated fact-checking system. Existing fact-checking approaches can be categorized as *expert-oriented*, *crowdsourcing-oriented*, and *computational-oriented*.

- *Expert-oriented* fact-checking heavily relies on human domain experts to investigate relevant data and documents to construct the verdicts of claim veracity, for example PolitiFact[11], Snopes[12], etc. However, expert-oriented fact-checking is an intellectually demanding and time-consuming process, which limits the potential for high efficiency and scalability.

- *Crowdsourcing-oriented* fact-checking exploits the "wisdom of crowd" to enable normal people to annotate news content; these annotations are then aggregated to produce an overall assessment of the news veracity. For example, Fiskkit[13] allows users to discuss and annotate the accuracy of specific parts of a news article. As another example, an anti-fake news bot named "For real" is a public account in the instant communication mobile application LINE[14], which allows people to report suspicious news content which is then further checked by editors.

- *Computational-oriented* fact-checking aims to provide an automatic scalable system to classify true and false claims. Previous computational-oriented fact checking methods try to solve two majors issues: (i) identifying

[11]http://www.politifact.com/
[12]http://www.snopes.com/
[13]http://fiskkit.com
[14]https://grants.g0v.tw/projects/588fa7b382223f001e022944

check-worthy claims and (ii) discriminating the veracity of fact claims. To identify check-worthy claims, factual claims in news content are extracted that convey key statements and viewpoints, facilitating the subsequent fact-checking process [31]. Fact-checking for specific claims largely relies on *external resources* to determine the truthfulness of a particular claim. Two typical external sources include the *open web* and structured *knowledge graph*. Open web sources are utilized as references that can be compared with given claims in terms of both the consistency and frequency [5; 50]. Knowledge graphs are integrated from the linked open data as a structured network topology, such as DBpedia and Google Relation Extraction Corpus. Fact-checking using a knowledge graph aims to check whether the claims in news content can be inferred from existing facts in the knowledge graph [98; 16; 72].

Style-based: Fake news publishers often have malicious intent to spread distorted and misleading information and influence large communities of consumers, requiring particular writing styles necessary to appeal to and persuade a wide scope of consumers that is not seen in true news articles. Style-based approaches try to detect fake news by capturing the *manipulators* in the writing style of news content. There are mainly two typical categories of style-based methods: *Deception-oriented* and *Objectivity-oriented*.

- *Deception-oriented* stylometric methods capture the deceptive statements or claims from news content. The motivation of deception detection originates from forensic psychology (i.e., Undeutsch Hypothesis) [82] and various forensic tools including Criteria-based Content Analysis [84] and Scientific-based Content Analysis [45] have been developed. More recently, advanced natural language processing models are applied to spot deception phases from the following perspectives: *Deep syntax* and *Rhetorical structure*. Deep syntax models have been implemented using probabilistic context free grammers (PCFG), with which sentences can be transformed into rules that describe the syntax structure. Based on the PCFG, different rules can be developed for deception detection, such as unlexicalized/ lexicalized production rules and grandparent rules [22]. Rhetorical structure theory can be utilized to capture the differences between deceptive and truthful sentences [68]. Deep network models, such as convolutional neural networks (CNN), have also been applied to classify fake news veracity [90].
- *Objectivity-oriented* approaches capture style signals that can indicate a decreased objectivity of news content and thus the potential to mislead consumers, such as hyperpartisan styles and yellow-journalism. Hyperpartisan styles represent extreme behavior in favor of a particular political party, which often correlates with a strong motivation to create fake news. Linguistic-based features can be applied to detect hyperpartisan articles [62]. Yellow-journalism represents those articles that do not contain well-researched news, but instead rely on eye-catching headlines (i.e., clickbait) with a propensity for exaggeration, sensationalization, scare-mongering, etc. Often, news titles will summarize the major viewpoints of the article that the author wants to convey, and thus misleading and deceptive clickbait titles can serve as a good indicator for recognizing fake news articles [13].

3.3.2 *Social Context Models*

The nature of social media provides researchers with additional resources to supplement and enhance News Content Models. Social context models include relevant user social engagements in the analysis, capturing this auxiliary information from a variety of perspectives. We can classify existing approaches for social context modeling into two categories: *Stance-based* and *Propagation-based*. Note that very few existing fake news detection approaches have utilized social context models. Thus, we also introduce similar methods for rumor detection using social media, which have potential application for fake news detection.

Stance-based: Stance-based approaches utilize users' viewpoints from relevant post contents to infer the veracity of original news articles. The stance of users' posts can be represented either *explicitly* or *implicitly*. Explicit stances are direct expressions of emotion or opinion, such as the "thumbs up" and "thumbs down" reactions expressed in Facebook. Implicit stances can be automatically extracted from social media posts. Stance detection is the task of automatically determining from a post whether the user is in favor of, neutral toward, or against some target entity, event, or idea [53]. Previous stance classification methods mainly rely on hand-crafted linguistic or embedding features on individual posts to predict stances [53; 64]. Topic model methods, such as latent dirichlet allocation (LDA) can be applied to learn latent stance from topics [37]. Using these methods, we can infer the news veracity based on the stance values of relevant posts. Tacchini *et al.* proposed to construct a bipartite network of user and Facebook posts using the "like" stance information [75]; based on this network, a semi-supervised probabilistic model was used to predict the likelihood of Facebook posts being hoaxes. Jin *et al.* explored topic models to learn latent viewpoint values and further exploited these viewpoints to learn the credibility of relevant posts and news content [37].

Propagation-based: Propagation-based approaches for fake news detection reason about the *interrelations* of relevant social media posts to predict news credibility. The basic assumption is that the credibility of a news event is highly related to the credibilities of relevant social media posts. Both *homogeneous* and *heterogeneous* credibility networks can be built for propagation process. Homogeneous credibility networks consist of a single type of entities, such as post or event [37]. Heterogeneous credibility networks involve different types of entities, such as posts, sub-events, and events [36; 29]. Gupta *et al.* proposed a PageRank-like credibility propagation algorithm by encoding users' credibilities and tweets' implications on a three layer user-tweet-event heterogeneous information network. Jin *et al.* proposed to include news aspects (i.e., latent sub-events), build a three-layer hierarchical network, and utilize a graph optimization framework to infer event credibilities. Recently, the conflicting viewpoint relationships are included to build a homogeneous credibility network among tweets and guide the process to evaluate their credibilities [37].

4. ASSESSING DETECTION EFFICACY

In this section, we discuss how to assess the performance of algorithms for fake news detection. We focus on the available datasets and evaluation metrics for this task.

4.1 Datasets

Online news can be collected from different sources, such as news agency homepages, search engines, and social media websites. However, manually determining the veracity of news is a challenging task, usually requiring annotators with domain expertise who performs careful analysis of claims and additional evidence, context, and reports from authoritative sources. Generally, news data with annotations can be gathered in the following ways: *Expert journalists*, *Fact-checking websites*, *Industry detectors*, and *Crowdsourced workers*. However, there are no agreed upon benchmark datasets for the fake news detection problem. Some publicly available datasets are listed below:

- *BuzzFeedNews*[15]: This dataset comprises a complete sample of news published in Facebook from 9 news agencies over a week close to the 2016 U.S. election from September 19 to 23 and September 26 and 27. Every post and the linked article were fact-checked claim-by-claim by 5 BuzzFeed journalists. This dataset is further enriched in [62] by adding the linked articles, attached media, and relevant metadata. It contains 1,627 articles–826 mainstream, 356 left-wing, and 545 right-wing articles.
- *LIAR*[16]: This dataset is collected from fact-checking website PolitiFact through its API [90]. It includes 12,836 human-labeled short statements, which are sampled from various contexts, such as news releases, TV or radio interviews, campaign speeches, etc. The labels for news truthfulness are fine-grained multiple classes: pants-fire, false, barely-true, half-true, mostly true, and true.
- *BS Detector*[17]: This dataset is collected from a browser extension called BS detector developed for checking news veracity[18]. It searches all links on a given webpage for references to unreliable sources by checking against a manually complied list of domains. The labels are the outputs of BS detector, rather than human annotators.
- *CREDBANK*[19]: This is a large scale crowdsourced dataset of approximately 60 million tweets that cover 96 days starting from October 2015. All the tweets are broken down to be related to over 1,000 news events, with each event assessed for credibilities by 30 annotators from Amazon Mechanical Turk [52].

In Table 1, we compare these public fake news detection datasets, highlighting the features that can be extracted from each dataset. We can see that no existing public dataset can provide all possible features of interest. In addition, these datasets also have specific limitation that make them challenging to use for fake news detection. BuzzFeedNews only contains headlines and text for each news piece and covers news articles from very few news agencies. LIAR includes mostly short statements, rather than the entire news content. Further, these statements are collected from various speakers, rather than news publishers, and may include some claims that are not fake news. BS Detector data is collected and annotated by using a developed news veracity checking tool. As the labels have not been properly validated by human experts, any model trained on this data is really learning the parameters of BS Detector, rather than expert-annotated ground truth fake news. Finally, CREDBANK was originally collected for tweet credibility assessment, so the tweets in this dataset are not really the social engagements for specific news articles.

To address the disadvantages of existing fake news detection datasets, we have an ongoing project to develop a usable dataset for fake news detection on social media. This dataset, called *FakeNewsNet*[20], includes all mentioned news content and social context features with reliable ground truth fake news labels.

4.2 Evaluation Metrics

To evaluate the performance of algorithms for fake news detection problem, various evaluation metrics have been used. In this subsection, we review the most widely used metrics for fake news detection. Most existing approaches consider the fake news problem as a classification problem that predicts whether a news article is fake or not:

- True Positive (**TP**): when predicted fake news pieces are actually annotated as fake news;
- True Negative (**TN**): when predicted true news pieces are actually annotated as true news;
- False Negative (**FN**): when predicted true news pieces are actually annotated as fake news;
- False Positive (**FP**): when predicted fake news pieces are actually annotated as true news.

By formulating this as a classification problem, we can define following metrics,

$$Precision = \frac{|TP|}{|TP| + |FP|} \quad (2)$$

$$Recall = \frac{|TP|}{|TP| + |FN|} \quad (3)$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (4)$$

$$Accuracy = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \quad (5)$$

These metrics are commonly used in the machine learning community and enable us to evaluate the performance of a classifier from different perspectives. Specifically, accuracy measures the similarity between predicted fake news and real fake news. Precision measures the fraction of all detected fake news that are annotated as fake news, addressing the important problem of identifying which news is fake. However, because fake news datasets are often skewed, a high precision can be easily achieved by making fewer positive

[15]https://github.com/BuzzFeedNews/2016-10-facebook-fact-check/tree/master/data
[16]https://www.cs.ucsb.edu/ william/data/liar_dataset.zip
[17]https://www.kaggle.com/mrisdal/fake-news
[18]https://github.com/bs-detector/bs-detector
[19]http://compsocial.github.io/CREDBANK-data/
[20]https://github.com/KaiDMML/FakeNewsNet

Table 1: Comparison of Fake News Detection Datasets.

Dataset \ Features	News Content		Social Context		
	Linguistic	**Visual**	**User**	**Post**	**Network**
BuzzFeedNews	✓				
LIAR	✓				
BS Detector	✓				
CREDBANK	✓		✓	✓	✓

predictions. Thus, recall is used to measure the sensitivity, or the fraction of annotated fake news articles that are predicted to be fake news. F1 is used to combine precision and recall, which can provide an overall prediction performance for fake news detection. Note that for *Precision*, *Recall*, F_1, and *Accuracy*, the higher the value, the better the performance.

The *Receiver Operating Characteristics* (ROC) curve provides a way of comparing the performance of classifiers by looking at the trade-off in the *False Positive Rate* (FPR) and the *True Positive Rate* (TPR). To draw the ROC curve, we plot the FPR on the x axis and and TPR along the y axis. The ROC curve compares the performance of different classifiers by changing class distributions via a threshold. TPR and FPR are defined as follows (note that TPR is the same as recall defined above):

$$TPR = \frac{|TP|}{|TP| + |FN|} \quad (6)$$

$$FPR = \frac{|FP|}{|FP| + |TN|} \quad (7)$$

Based on the ROC curve, we can compute the Area Under the Curve (AUC) value, which measures the overall performance of how likely the classifier is to rank the fake news higher than any true news. Based on [30], AUC is defined as below.

$$AUC = \frac{\sum(n_0 + n_1 + 1 - r_i) - n_0(n_0 + 1)/2}{n_0 n_1} \quad (8)$$

where r_i is the rank of i_{th} fake news piece and n_0 (n_1) is the number of fake (true) news pieces. It is worth mentioning that AUC is more statistically consistent and more discriminating than accuracy [47], and it is usually applied in an imbalanced classification problem, such as fake news classification, where the number of ground truth fake news articles and and true news articles have a very imbalanced distribution.

5. RELATED AREAS

In this section, we further discuss areas that are related to the problem of fake news detection. We aim to point out the differences between these areas and fake news detection by briefly explaining the task goals and highlighting some popular methods.

5.1 Rumor Classification

A rumor can usually be defined as "a piece of circulating information whose veracity status is yet to be verified at the time of spreading" [102]. The function of a rumor is to make sense of an *ambiguous* situation, and the truthfulness value could be *true*, *false* or *unverified*. Previous approaches for rumor analysis focus on four subtasks: rumor detection, rumor tracking, stance classification, and veracity classification [102]. Specifically, rumor detection aims to classify a piece of information as rumor or non-rumor [96; 70]; rumor tracking aims to collect and filter posts discussing specific rumors; rumor stance classification determines how each relevant post is oriented with respect to the rumor's veracity; veracity classification attempts to predict the actual truth value of the rumor. The most related task to fake news detection is the rumor veracity classification. Rumor veracity classification relies heavily on the other subtasks, requiring the stances or opinions can be extracted from relevant posts. These posts are considered as important sensors for determining the veracity of the rumor. Different from rumors, which may include long-term rumors, such as conspiracy theories, as well as short-term emerging rumors, fake news refers to information related specifically to public news events that can be verified as false.

5.2 Truth Discovery

Truth discovery is the problem of detecting true facts from multiple conflicting sources [46]. Truth discovery methods do not explore the fact claims directly, but rely on a collection of contradicting sources that record the properties of objects to determine the truth value. Truth discovery aims to determine the *source credibility* and *object truthfulness* at the same time. The fake news detection problem can benefit from various aspects of truth discovery approaches under different scenarios. First, the credibility of different news outlets can be modeled to infer the truthfulness of reported news. Second, relevant social media posts can also be modeled as social response sources to better determine the truthfulness of claims [56; 93]. However, there are some other issues that must be considered to apply truth discovery to fake news detection in social media scenarios. First, most existing truth discovery methods focus on handling *structured* input in the form of Subject-Predicate-Object (SPO) tuples, while social media data is highly unstructured and noisy. Second, truth discovery methods can not be well applied when a fake news article is newly launched and published by only a few news outlets because at that point there is not enough social media posts relevant to it to serve as additional sources.

5.3 Clickbait Detection

Clickbait is a term commonly used to describe eye-catching and teaser headlines in online media. Clickbait headlines create a so-called "curiosity gap", increasing the likelihood that reader will click the target link to satisfy their curiosity. Existing clickbait detection approaches utilize various linguistic features extracted from teaser messages, linked webpages, and tweet meta information [12; 8; 63]. Different types of clickbait are categorized, and some of them are highly correlated with non-factual claims [7]. The underly-

ing motivation of clickbait is usually for click-through rates and the resultant advertising revenue. Thus, the body text of clickbait articles are often informally organized and poorly reasoned. This discrepancy has been used by researchers to identify the inconsistency between headlines and news contents in an attempt to detect fake news articles[21]. Even though not all fake news may include clickbait headlines, specific clickbait headlines could serve as an important indicator, and various features can be utilized to help detect fake news.

5.4 Spammer and Bot Detection

Spammer detection on social media, which aims to capture malicious users that coordinate among themselves to launch various attacks, such as spreading ads, disseminating pornography, delivering viruses, and phishing [44], has recently attracted wide attention. Existing approaches for social spammer detection mainly rely on extracting features from user activities and social network information [35; 95; 33; 34]. In addition, the rise of social bots has also increased the circulation of false information as they automatically retweet posts without verifying the facts [23]. The major challenge brought by social bots is that they can give a false impression that information is highly popular and endorsed by many people, which enables the echo chamber effect for the propagation of fake news. Previous approaches for bot detection are based on social network information, crowdsourcing, and discriminative features [23; 55; 54]. Thus, both spammer and social bots could provide insights about target specific malicious social media accounts that can be used for fake news detection.

6. OPEN ISSUES AND FUTURE RESEARCH

In this section, we present some open issues in fake news detection and future research directions. Fake news detection on social media is a newly emerging research area, so we aim to point out promising research directions from a data mining perspective. Specifically, as shown in Figure 2, we outline the research directions in four categories: *Data-oriented*, *Feature-oriented*, *Model-oriented* and *Application-oriented*.

Data-oriented: Data-oriented fake news research is focusing on different kinds of data characteristics, such as : *dataset*, *temporal* and *psychological*. From a dataset perspective, we demonstrated that there is no existing benchmark dataset that includes resources to extract all relevant features. A promising direction is to create a comprehensive and large-scale fake news benchmark dataset, which can be used by researchers to facilitate further research in this area. From a temporal perspective, fake news dissemination on social media demonstrates unique temporal patterns different from true news. Along this line, one interesting problem is to perform *early fake news detection*, which aims to give early alerts of fake news during the dissemination process. For example, this approach could look at only social media posts within some time delay of the original post as sources for news verification [37]. Detecting fake news early can help prevent further propagation on social media. From a psychological perspective, different aspects of fake news have been qualitatively explored in the social psychology literature [92; 58; 59], but quantitative studies to verify these psychological factors are rather limited. For example, the echo chamber effect plays an important role for fake news spreading in social media. Then how to capture echo chamber effects and how to utilize the pattern for fake news detection in social media could be an interesting investigation. Moreover, *intention detection* from news data is promising but limited as most existing fake news research focus on detecting the authenticity but ignore the intent aspect of fake news. Intention detection is very challenging as the intention is often explicitly unavailable. Thus. it's worth to explore how to use data mining methods to validate and capture psychology intentions.

Feature-oriented: Feature-oriented fake news research aims to determine effective features for detecting fake news from multiple data sources. We have demonstrated that there are two major data sources: *news content* and *social context*. From a news content perspective, we introduced linguistic-based and visual-based techniques to extract features from text information. Note that linguistic-based features have been widely studied for general NLP tasks, such as text classification and clustering, and specific applications such as author identification [32] and deception detection [22], but the underlying characteristics of fake news have not been fully understood. Moreover, embedding techniques, such as word embedding and deep neural networks, are attracting much attention for textual feature extraction, and has the potential to learn better representations [90; 87; 88]. In addition, *visual features* extracted from images are also shown to be important indicators for fake news [38]. However, very limited research has been done to exploit effective visual features, including traditional local and global features [61] and newly emerging deep network-based features [43; 89; 85], for the fake news detection problem. Recently, it has been shown that advanced tools can manipulate video footage of public figures [80], synthesize high quality videos [74], etc. Thus, it becomes much more challenging and important to differentiate real and fake visual content, and more advanced visual-based features are needed for this research. From a social context perspective, we introduced user-based, post-based, and network-based features. Existing user-based features mainly focus on general user profiles, rather than differentiating account types separately and extracting user-specific features. Post-based features can be represented using other techniques, such as convolutional neural networks (CNN) [69], to better capture people's opinions and reactions toward fake news. Images in social media posts can also be utilized to better understand users' sentiments [91] toward news events. Network-based features are extracted to represent how different types of networks are constructed. It is important to extend this preliminary work to explore (i) how other networks can be constructed in terms of different aspects of relationships among relevant users and posts; and (ii) other advanced methods of network representations, such as network embedding [78; 86].

Model-oriented: Model-oriented fake news research opens the door to building more effective and practical models for fake news detection. Most previously mentioned approaches focus on extracting various features, incorporating theses features into supervised classification models, such as naïve Bayes, decision tree, logistic regression, k nearest neighbor

[21] http://www.fakenewschallenge.org/

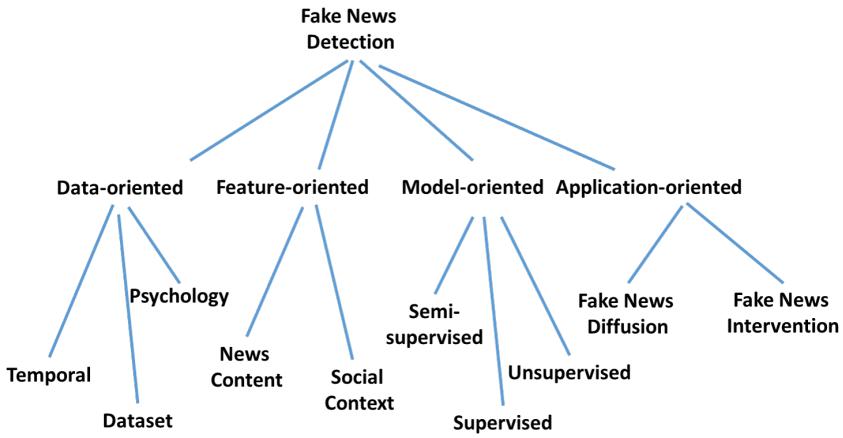

Figure 2: Future directions and open issues for fake news detection on social media.

(KNN), and support vector machines (SVM), and then selecting the classifier that performs the best [62; 75; 1]. More research can be done to build more complex and effective models and to better utilize extracted features, such as *aggregation methods*, *probabilistic methods*, *ensemble methods*, or *projection methods* [73]. Specifically, we think there is some promising research in the following directions. First, aggregation methods combine different feature representations into a weighted form and optimize the feature weights. Second, since fake news may commonly mix true statements with false claims, it may make more sense to predict the likelihood of fake news instead of producing a binary value; probabilistic models predict a probabilistic distribution of class labels (i.e., fake news versus true news) by assuming a generative model that pulls from the same distribution as the original feature space [25]. Third, one of the major challenges for fake news detection is the fact that each feature, such as source credibility, news content style, or social response, has some limitations to directly predict fake news on its own. Ensemble methods build a conjunction of several weak classifiers to learn a strong classifier that is more successful than any individual classifier alone; ensembles have been widely applied to various applications in the machine learning literature [20]. It may be beneficial to build ensemble models as news content and social context features each have supplementary information that has the potential to boost fake news detection performance. Finally, fake news content or social context information may be noisy in the raw feature space; projection methods refer to approaches that lean projection functions to map between original feature spaces (e.g., news content features and social context features) and the latent feature spaces that may be more useful for classification.

Moreover, most existing approaches are *supervised*, which requires a pre-annotated fake news ground truth dataset to train a model. However, obtaining a reliable fake news dataset is very time and labor intensive, as the process often requires expert annotators to perform careful analysis of claims and additional evidence, context, and reports from authoritative sources. Thus, it is also important to consider scenarios where limited or no labeled fake news pieces are available in which *semi-supervised* or *unsupervised* models can be applied. While the models created by supervised classification methods may be more accurate given a well-curated ground truth dataset for training, unsupervised models can be more practical because unlabeled datasets are easier to obtain.

Application-oriented: Application-oriented fake news research encompass research that goes into other areas beyond fake news detection. We propose two major directions along these lines: *fake news diffusion* and *fake news intervention*. Fake news diffusion characterizes the diffusion paths and patterns of fake news on social media sites. Some early research has shown that true information and misinformation follow different patterns when propagating in online social networks [18; 51]. Similarly, the diffusion of fake news in social media demonstrates its own characteristics that need further investigation, such as *social dimensions*, *life cycle*, *spreader identification*, etc. Social dimensions refer to the heterogeneity and weak dependency of social connections within different social communities. Users' perceptions of fake news pieces are highly affected by their like-minded friends in social media (i.e., echo chambers), while the *degree* differs along different social dimensions. Thus, it is worth exploring why and how different social dimensions play a role in spreading fake news in terms of different topics, such as political, education, sports, etc. The fake news diffusion process also has different stages in terms of people's attentions and reactions as time goes by, resulting in a unique life cycle. Research has shown that breaking news and in-depth news demonstrate different life cycles in social media [10]. Studying the life cycle of fake news will provide deeper understanding of how particular stories "go viral" from normal public discourse. Tracking the life cycle of fake news on social media requires recording essential trajectories of fake news diffusion in general [71], as well as further investigations of the process for specific fake news pieces, such as graph-based models and evolution-based models [27]. In addition, identifying key spreaders of fake news is crucial to mitigate the diffusion scope in social media. Note that key spreaders can be categorized in two ways, i.e., *stance* and *authenticity*. Along the stance dimensions, spreaders can either be (i) *clarifiers*, who propose skeptical and opposing viewpoints towards fake news and try to clarify them; or (ii) *persuaders*, who spread fake news with supporting opinions

to persuade others to believe it. In this sense, it is important to explore how to detect clarifiers and persuaders and better use them to control the dissemination of fake news. From an authenticity perspective, spreaders could be either *human*, *bot*, or *cyborg*. Social bots have been used to intentionally spread fake news in social media, which motivates further research to better characterize and detect malicious accounts designed for propaganda.

Finally, we also propose further research into fake news intervention, which aims to reduce the effects of fake news by *proactive* intervention methods that minimize the spread scope or *reactive* intervention methods after fake news goes viral. Proactive fake news intervention methods try to (i) remove malicious accounts that spread fake news or fake news itself to *isolate* it from future consumers; (ii) *immunize* users with true news to change the belief of users that may already have been affected by fake news. There is recent research that attempts to use content-based immunization and network-based immunization methods in misinformation intervention [94; 97]. One approach uses a multivariate Hawkes process to model both true news and fake news and mitigate the spreading of fake news in real-time [21]. The aforementioned spreader detection techniques can also be applied to target certain users (e.g., persuaders) in social media to stop spreading fake news, or other users (e.g. clarifiers) to maximize the spread of corresponding true news.

7. CONCLUSION

With the increasing popularity of social media, more and more people consume news from social media instead of traditional news media. However, social media has also been used to spread fake news, which has strong negative impacts on individual users and broader society. In this article, we explored the fake news problem by reviewing existing literature in two phases: characterization and detection. In the characterization phase, we introduced the basic concepts and principles of fake news in both traditional media and social media. In the detection phase, we reviewed existing fake news detection approaches from a data mining perspective, including feature extraction and model construction. We also further discussed the datasets, evaluation metrics, and promising future directions in fake news detection research and expand the field to other applications.

8. ACKNOWLEDGEMENTS

This material is based upon work supported by, or in part by, the ONR grant N00014-16-1-2257.

9. REFERENCES

[1] Sadia Afroz, Michael Brennan, and Rachel Greenstadt. Detecting hoaxes, frauds, and deception in writing style online. In *ISSP'12*.

[2] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. Technical report, National Bureau of Economic Research, 2017.

[3] Solomon E Asch and H Guetzkow. Effects of group pressure upon the modification and distortion of judgments. *Groups, leadership, and men*, pages 222–236, 1951.

[4] Meital Balmas. When fake news becomes real: Combined exposure to multiple news sources and political attitudes of inefficacy, alienation, and cynicism. *Communication Research*, 41(3):430–454, 2014.

[5] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *IJCAI'07*.

[6] Alessandro Bessi and Emilio Ferrara. Social bots distort the 2016 us presidential election online discussion. *First Monday*, 21(11), 2016.

[7] Prakhar Biyani, Kostas Tsioutsiouliklis, and John Blackmer. " 8 amazing secrets for getting more clicks": Detecting clickbaits in news streams using article informality. In *AAAI'16*.

[8] Jonas Nygaard Blom and Kenneth Reinecke Hansen. Click bait: Forward-reference as lure in online news headlines. *Journal of Pragmatics*, 76:87–100, 2015.

[9] Paul R Brewer, Dannagal Goldthwaite Young, and Michelle Morreale. The impact of real news about fake news: Intertextual processes and political satire. *International Journal of Public Opinion Research*, 25(3):323–343, 2013.

[10] Carlos Castillo, Mohammed El-Haddad, Jürgen Pfeffer, and Matt Stempeck. Characterizing the life cycle of online news stories using social media reactions. In *CSCW'14*.

[11] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *WWW'11*.

[12] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. Stop clickbait: Detecting and preventing clickbaits in online news media. In *ASONAM'16*.

[13] Yimin Chen, Niall J Conroy, and Victoria L Rubin. Misleading online content: Recognizing clickbait as false news. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*, pages 15–19. ACM, 2015.

[14] Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. Anyone can become a troll: Causes of trolling behavior in online discussions. In *CSCW '17*.

[15] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *IEEE Transactions on Dependable and Secure Computing*, 9(6):811–824, 2012.

[16] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PloS one*, 10(6):e0128193, 2015.

[17] Niall J Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: Methods for finding fake news. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

[18] Michela Del Vicario, Alessandro Bessi, Fabiana Zollo, Fabio Petroni, Antonio Scala, Guido Caldarelli, H Eu-

gene Stanley, and Walter Quattrociocchi. The spreading of misinformation online. *Proceedings of the National Academy of Sciences*, 113(3):554–559, 2016.

[19] Michela Del Vicario, Gianna Vivaldo, Alessandro Bessi, Fabiana Zollo, Antonio Scala, Guido Caldarelli, and Walter Quattrociocchi. Echo chambers: Emotional contagion and group polarization on facebook. *Scientific Reports*, 6, 2016.

[20] Thomas G Dietterich et al. Ensemble methods in machine learning. *Multiple classifier systems*, 1857:1–15, 2000.

[21] Mehrdad Farajtabar, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Khalil, Shuang Li, Le Song, and Hongyuan Zha. Fake news mitigation via point process based intervention. *arXiv preprint arXiv:1703.07823*, 2017.

[22] Song Feng, Ritwik Banerjee, and Yejin Choi. Syntactic stylometry for deception detection. In *ACL'12*.

[23] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. The rise of social bots. *Communications of the ACM*, 59(7):96–104, 2016.

[24] Johannes Fürnkranz. A study using n-gram features for text categorization. *Austrian Research Institute for Artifical Intelligence*, 3(1998):1–10, 1998.

[25] Ashutosh Garg and Dan Roth. Understanding probabilistic classifiers. *ECML'01*.

[26] Matthew Gentzkow, Jesse M Shapiro, and Daniel F Stone. Media bias in the marketplace: Theory. Technical report, National Bureau of Economic Research, 2014.

[27] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A Zighed. Information diffusion in online social networks: A survey. *ACM Sigmod Record*, 42(2):17–28, 2013.

[28] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *WWW'13*.

[29] Manish Gupta, Peixiang Zhao, and Jiawei Han. Evaluating event credibility on twitter. In *PSDM'12*.

[30] David J Hand and Robert J Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 2001.

[31] Naeemul Hassan, Chengkai Li, and Mark Tremayne. Detecting check-worthy factual claims in presidential debates. In *CIKM'15*.

[32] John Houvardas and Efstathios Stamatatos. N-gram feature selection for authorship identification. *Artificial Intelligence: Methodology, Systems, and Applications*, pages 77–86, 2006.

[33] Xia Hu, Jiliang Tang, Huiji Gao, and Huan Liu. Social spammer detection with sentiment information. In *ICDM'14*.

[34] Xia Hu, Jiliang Tang, and Huan Liu. Online social spammer detection. In *AAAI'14*, pages 59–65, 2014.

[35] Xia Hu, Jiliang Tang, Yanchao Zhang, and Huan Liu. Social spammer detection in microblogging. In *IJCAI'13*.

[36] Zhiwei Jin, Juan Cao, Yu-Gang Jiang, and Yongdong Zhang. News credibility evaluation on microblog with a hierarchical propagation model. In *ICDM'14*.

[37] Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. News verification by exploiting conflicting social viewpoints in microblogs. In *AAAI'16*.

[38] Zhiwei Jin, Juan Cao, Yongdong Zhang, Jianshe Zhou, and Qi Tian. Novel visual and statistical image features for microblogs news verification. *IEEE Transactions on Multimedia*, 19(3):598–608, 2017.

[39] Daniel Kahneman and Amos Tversky. Prospect theory: An analysis of decision under risk. *Econometrica: Journal of the econometric society*, pages 263–291, 1979.

[40] Jean-Noel Kapferer. *Rumors: Uses, Interpretation and Necessity*. Routledge, 2017.

[41] David O Klein and Joshua R Wueller. Fake news: A legal perspective. 2017.

[42] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *ICDM'13*, pages 1103–1108. IEEE, 2013.

[43] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[44] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: social honeypots+ machine learning. In *SIGIR'10*.

[45] Tony Lesce. Scan: Deception detection by scientific content analysis. *Law and Order*, 38(8):3–6, 1990.

[46] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. A survey on truth discovery. *ACM Sigkdd Explorations Newsletter*, 17(2):1–16, 2016.

[47] Charles X Ling, Jin Huang, and Harry Zhang. Auc: a statistically consistent and more discriminating measure than accuracy.

[48] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. Detecting rumors from microblogs with recurrent neural networks.

[49] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblogging websites. In *CIKM'15*.

[50] Amr Magdy and Nayer Wanas. Web-based statistical fact checking of textual documents. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 103–110. ACM, 2010.

[51] Filippo Menczer. The spread of misinformation in social media. In *WWW'16*.

[52] Tanushree Mitra and Eric Gilbert. Credbank: A large-scale social media corpus with associated credibility annotations. In *ICWSM'15*.

[53] Saif M Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko. Stance and sentiment in tweets.

ACM Transactions on Internet Technology (TOIT), 17(3):26, 2017.

[54] Fred Morstatter, Harsh Dani, Justin Sampson, and Huan Liu. Can one tamper with the sample api?: Toward neutralizing bias from spam and bot content. In *WWW'16*.

[55] Fred Morstatter, Liang Wu, Tahora H Nazer, Kathleen M Carley, and Huan Liu. A new approach to bot detection: Striking the balance between precision and recall. In *ASONAM'16*.

[56] Subhabrata Mukherjee and Gerhard Weikum. Leveraging joint interactions for credibility analysis in news communities. In *CIKM'15*.

[57] Eni Mustafaraj and Panagiotis Takis Metaxas. The fake news spreading plague: Was it preventable? *arXiv preprint arXiv:1703.06988*, 2017.

[58] Raymond S Nickerson. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175, 1998.

[59] Brendan Nyhan and Jason Reifler. When corrections fail: The persistence of political misperceptions. *Political Behavior*, 32(2):303–330, 2010.

[60] Christopher Paul and Miriam Matthews. The russian firehose of falsehood propaganda model.

[61] Dong ping Tian et al. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 8(4):385–396, 2013.

[62] Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*, 2017.

[63] Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. Clickbait detection. In *European Conference on Information Retrieval*, pages 810–817. Springer, 2016.

[64] Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. Rumor has it: Identifying misinformation in microblogs. In *EMNLP'11*.

[65] Walter Quattrociocchi, Antonio Scala, and Cass R Sunstein. Echo chambers on facebook. 2016.

[66] Victoria L Rubin, Yimin Chen, and Niall J Conroy. Deception detection for news: three types of fakes. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

[67] Victoria L Rubin, Niall J Conroy, Yimin Chen, and Sarah Cornwell. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of NAACL-HLT*, pages 7–17, 2016.

[68] Victoria L Rubin and Tatiana Lukoianova. Truth and deception at the rhetorical structure level. *Journal of the Association for Information Science and Technology*, 66(5):905–917, 2015.

[69] Natali Ruchansky, Sungyong Seo, and Yan Liu. Csi: A hybrid deep model for fake news. *arXiv preprint arXiv:1703.06959*, 2017.

[70] Justin Sampson, Fred Morstatter, Liang Wu, and Huan Liu. Leveraging the implicit structure within social media for emergent rumor detection. In *CIKM'15*.

[71] Chengcheng Shao, Giovanni Luca Ciampaglia, Alessandro Flammini, and Filippo Menczer. Hoaxy: A platform for tracking online misinformation. In *WWW'16*.

[72] Baoxu Shi and Tim Weninger. Fact checking in heterogeneous information networks. In *WWW'16*.

[73] Kai Shu, Suhang Wang, Jiliang Tang, Reza Zafarani, and Huan Liu. User identity linkage across online social networks: A review. *ACM SIGKDD Explorations Newsletter*, 18(2):5–17, 2017.

[74] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):95, 2017.

[75] Eugenio Tacchini, Gabriele Ballarin, Marco L Della Vedova, Stefano Moret, and Luca de Alfaro. Some like it hoax: Automated fake news detection in social networks. *arXiv preprint arXiv:1704.07506*, 2017.

[76] Henri Tajfel and John C Turner. An integrative theory of intergroup conflict. *The social psychology of intergroup relations*, 33(47):74, 1979.

[77] Henri Tajfel and John C Turner. The social identity theory of intergroup behavior. 2004.

[78] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *WWW'15*.

[79] Jiliang Tang, Yi Chang, and Huan Liu. Mining social media with social theories: a survey. *ACM SIGKDD Explorations Newsletter*, 15(2):20–29, 2014.

[80] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR'16*.

[81] Amos Tversky and Daniel Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty*, 5(4):297–323, 1992.

[82] Udo Undeutsch. Beurteilung der glaubhaftigkeit von aussagen. *Handbuch der psychologie*, 11:26–181, 1967.

[83] Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. *ACL'14*.

[84] Aldert Vrij. Criteria-based content analysis: A qualitative review of the first 37 studies. *Psychology, Public Policy, and Law*, 11(1):3, 2005.

[85] Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. Attributed signed network embedding. In *CIKM'17*.

[86] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. Signed network embedding in social media. In *SDM'17*.

[87] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. Linked document embedding for classification. In *CIKM'16*.

[88] Suhang Wang, Jiliang Tang, Fred Morstatter, and Huan Liu. Paired restricted boltzmann machine for linked data. In *CIKM'16*.

[89] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *WWW'17*.

[90] William Yang Wang. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

[91] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. Unsupervised sentiment analysis for social media images. In *IJCAI*, pages 2378–2379, 2015.

[92] Andrew Ward, L Ross, E Reed, E Turiel, and T Brown. Naive realism in everyday life: Implications for social conflict and misunderstanding. *Values and knowledge*, pages 103–135, 1997.

[93] Gerhard Weikum. What computers should know, shouldn't know, and shouldn't believe. In *WWW'17*.

[94] L Wu, F Morstatter, X Hu, and H Liu. Chapter 5: Mining misinformation in social media, 2016.

[95] Liang Wu, Xia Hu, Fred Morstatter, and Huan Liu. Adaptive spammer detection with sparse group modeling. In *ICWSM'17*.

[96] Liang Wu, Jundong Li, Xia Hu, and Huan Liu. Gleaning wisdom from the past: Early detection of emerging rumors in social media. In *SDM'17*.

[97] Liang Wu, Fred Morstatter, Xia Hu, and Huan Liu. Mining misinformation in social media. *Big Data in Complex and Social Networks*, pages 123–152, 2016.

[98] You Wu, Pankaj K Agarwal, Chengkai Li, Jun Yang, and Cong Yu. Toward computational fact-checking. *Proceedings of the VLDB Endowment*, 7(7):589–600, 2014.

[99] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 13. ACM, 2012.

[100] Robert B Zajonc. Attitudinal effects of mere exposure. *Journal of personality and social psychology*, 9(2p2):1, 1968.

[101] Robert B Zajonc. Mere exposure: A gateway to the subliminal. *Current directions in psychological science*, 10(6):224–228, 2001.

[102] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. Detection and resolution of rumours in social media: A survey. *arXiv preprint arXiv:1704.00656*, 2017.

Common pitfalls in training and evaluating recommender systems

Hung-Hsuan Chen[†], Chu-An Chung[‡], Hsin-Chien Huang[‡], Wen Tsui[‡]
[†]Computer Science and Information Engineering, National Central University
[‡]Computational Intelligence Technology Center, Industrial Technology Research Institute
hhchen@ncu.edu.tw, {JudyChung, hchuang, arvin}@itri.org.tw

ABSTRACT

This paper formally presents four common pitfalls in training and evaluating recommendation algorithms for information systems. Specifically, we show that it could be problematic to separate the server logs into training and test data for model generation and model evaluation if the training and the test data are selected improperly. In addition, we show that click through rate – a common metric to measure and compare the performance of different recommendation algorithms – may not be a good measurement of profitability – the income a recommendation module brings to a website. Moreover, we demonstrate that evaluating recommendation revenue may not be a straightforward task as it first looks. Unfortunately, these pitfalls appeared in many previous studies on recommender systems and information systems. We explicitly explain these problems and propose methods to address them. We conducted experiments to support our claims. Finally, we review previous papers and competitions that may suffer from these problems.

1. INTRODUCTION

A recommender system suggests items that may interest a user. Recommender systems are mostly adopted by the E-Commerce (EC) providers, such as Amazon [19] and Walmart [14]. Researchers have proposed many recommendation algorithms from various perspectives, such as leveraging on the text similarity between products, utilizing users' clicking or purchasing behaviors, or a combination of both. These methods include Collaborative Filtering, Matrix Factorization, language modeling, and their variations and mixed versions [16; 19; 21; 22; 24].

To evaluate the proposed recommendation algorithms, researchers have applied or invented assorted metrics, such as the click through rate (CTR), the Mean Absolute Error (MAE), Normalized Mean Absolute Error (NMAE), and the Root Mean Square Error (RMSE) between the predicted user rating and the true user rating, the Discounted Cumulative Gain (DCG) and Kendall's Tau of the recommended item list, and the traditional Information Retrieval metrics like Precision and Recall [14].

Recommender systems can be applied to various application domains, such as movie recommendation [7], music recommendation [11], collaborator recommendation [8], expert recommendation [9], etc. Recently, industrial companies and research labs, such as Criteo, Netflix, and YOOCHOOSE, have organized recommendation competitions that attract thousands of participants. Typically, the organizer provides the datasets for training and locks away the test data for the final evaluation [6; 7; 15].

Instead of proposing another recommendation algorithm or evaluation metric, this paper emphasizes on four common pitfalls of developing and evaluating recommendation modules for information systems. Probably influenced by the machine learning, data mining, and information retrieval fields, the researchers of recommender systems usually split the available dataset (e.g., the logs of clickstreams) into the training and the test data, which are used to generate the recommendation model and evaluate the performance of the model respectively. Although such a procedure works well in many machine learning studies and applications, applying this procedure to recommender systems could be problematic, as we will explain later in this paper. In addition, the typical evaluation metrics – click through rate – may be problematic if used carelessly. Unfortunately, many of these pitfalls appear in previous research papers and competitions, as we will illustrate later. We discuss these issues and propose possible ways to fix or bypass them in this paper.

The rest of the paper is organized as follows. In Section 2, we present the typical approach to separate the training and the test data of the studies on recommender systems. We discuss two issues of generating and evaluating the recommendation methods in Section 3 and Section 4. In Section 5 and Section 6, we show two issues regarding click through rate and recommendation revenue. In Section 7, we review previous works, including (1) the typical recommendation metrics and (2) previous competitions and publications that fall into several pitfalls illustrated in this paper. Finally, we discuss the discoveries and address future work in Section 8.

2. A TYPICAL PROCEDURE OF PREPARING TRAINING AND TEST DATASETS

Figure 1 shows a possible procedure of generating the training and the test data when studying recommendation modules of an information system, e.g., an EC website. Initially, an EC website probably had no recommendation module. At a certain time, the engineers of the website decided to include a recommendation model. To train the model, the engineers used the available logs to extract (x_i, y_i) as the training instance. Here, $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,\ell})$ is the context features ($i = 1, 2, \ldots, n$; ℓ: the number of features; n: the number of training instances). The context features could be, for example, the user's gender, location, education,

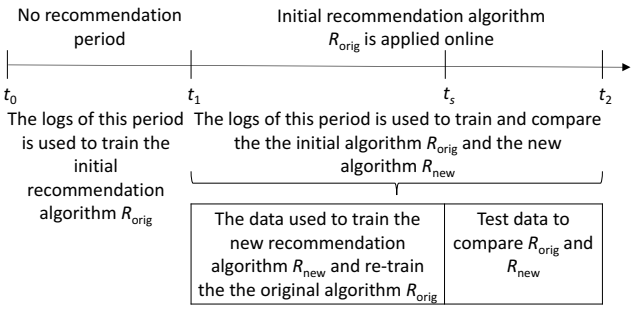

Figure 1: An illustration of a possible procedure to generate the training and the test data for developing and evaluating recommender systems. The initial online recommendation algorithm model R_{orig} is trained based on users' behaviors in the "no recommendation" period (from t_0 to t_1). To train a new model R_{new} and test whether the new recommendation method is better than R_{orig}, we split the log data between t_1 and t_2 into training and test data, as many research papers and competitions did. Both R_{orig} and R_{new} are trained based on the training data (the logs between t_1 and t_s) and compared based on the test data (the logs between t_s and t_2) using the specified metric (e.g., click through rate or order rate).

annual salary, the current browsing item's category, price, color, the current date, the current day of the week, etc. The y_i is the corresponding item that should be recommended to the user under the context features x_i. Since the website has no recommendation module initially, y_i could only be inferred based on intuitions, e.g., a user's next clicked or purchased item given x_i. During the test (recommendation) phase, the recommendation module utilizes the given context features as clues to output the products that may interest the users.

For example, if the engineers believe that a good recommender system should recommend products that are popular and related to the current browsing item, they may propose CategoryTP – recommending the top popular (e.g., most viewed) items the in the category c, the current browsing products' category. In this case, the context feature list contains only one feature $x_{i,1}$ whose value is c. To obtain the top popular items of each category, the view count of each product during the "no recommendation period" (see Figure 1) is calculated.

Suppose at another time, the engineers of the EC website proposed another recommendation algorithm R_{new}. To compare R_{new} with the original one R_{orig}, the engineers split the logs between t_1 and t_2 (the period where the original recommendation module was employed) into the training data (logs between t_1 and t_s) and the test data (logs between t_s and t_2). The engineers trained R_{new} the new model and re-trained R_{orig} the original model based on the training data, and compared their performance based on the test data using the specified metrics, such as the click through rate, the recommendation order rate, or the recommendation revenue.

Many research papers and competitions applied this procedure or similar procedures to generate the training and the test data, as will be illustrated in Section 7. Unfortunately, such a training and evaluation procedure is problematic, as demonstrated in the following sections.

Table 1: The percentage of the clicks resulted from the in-page direct links. We hide the actual numbers of clicks due to business sensitivity.

Day	Day 1	Day 2
Direct link Percentage	19.3150%	21.2812%

3. ISSUE 1: TRAINED MODEL COULD BE BIASED TOWARD HIGHLY REACHABLE PRODUCTS

This section shows that training a recommendation module based on the logs of the clickstreams may be problematic, if the training data are poorly selected. We will start by explaining the problem and proposing possible ways to bypass the problem. We will conduct experiments to support our claim.

3.1 The core problem

Many studies or competitions employ the clickstreams as the training data for the proposed recommendation algorithm [23]. Specifically, given that a user's current context feature as $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,\ell})$ and a user's next visited product as p_j, several researchers suggested to treat the recommendation task as a machine learning task in which (x_i, p_j) is a positive training instance. To generate the negative training instances, one possible approach is randomly sample a product p_k ($p_k \neq p_j$) from all products and treat (x_i, p_k) as a negative example [12].

If we generate the training dataset by this manner, the distribution of the training data (x_i, p_j) is affected by unstable factors, such as the presentation of the pages. Specifically, assume that we apply a Markov Chain recommendation model in which each node represents a product and $w_{i,j}$ the weight of a directed edge from node i to node j represents the transition probability from product p_i to p_j based on the log of the clickstreams. If the product page of p_i contains manifest direct-links to the product p_j, many users are likely to click p_j after browsing p_i. Thus, the information $p_i \rightarrow p_j$ (or more formally, $(x_i = (x_{i,1} = p_i), p_j)$ as a positive training instance) may become a strong positive signal simply because it is extremely easy to reach p_j from p_i. Unfortunately, the links included in the product pages are sometimes decided arbitrarily by few persons, e.g., the marketing executives or the engineers. As a result, the linked products may be little relevant to the current browsing product. For example, EC websites may aggressively exhibit the sponsored or advertised products, as demonstrated in Figure 2, even though the sponsored or the advertised products may not be directly relevant to the browsing product. As a result, the recommendation algorithms are likely to output the highly reachable products, which is highly influenced by the layout of the product pages.

3.2 Selecting proper training data

Based on the discussion above, we can see that a product's reachability highly influences the distribution of the train-

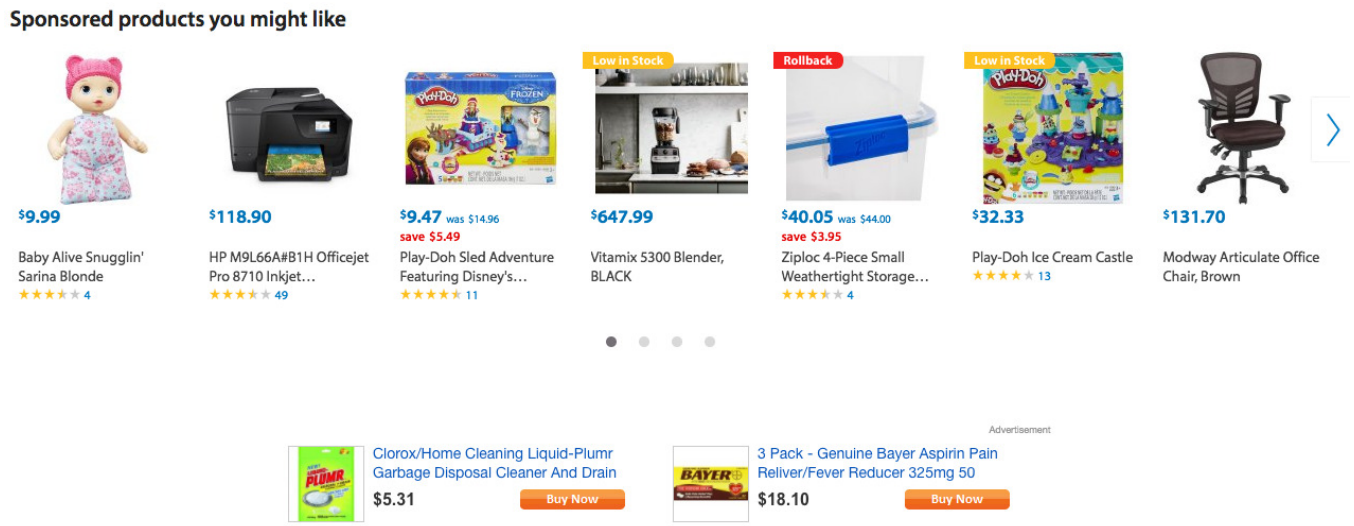

Figure 2: A snapshot of the product page on `https://www.walmart.com/`. The main product of this page is a prepaid smartphone. The sponsored products and the list of the advertised products on this page are of little relevance to a smart phone.

ing data. We illustrate two examples here. First, if we use the Markov Chain model as the training algorithm, the learned rules are highly influenced by the "layout" of the product pages (e.g., which product pages have direct links to which other product pages). Second, if we use popularity-based methods (e.g., obtaining the category of the current browsing product and recommending the top viewed products of the category), the recommendation algorithms are again likely to recommend the highly reachable products, such as the products that have many incoming links. Both cases are undesirable, because once we change the link targets, the distribution of the training data could be very different, which may lead to a different set of learned rules.

We analyzed the logs on two continuous days of a large EC website in Southeast Asia[1]. We found that about 20% of user clicks are resulted from the in-page direct links, as demonstrated in Table 1. This suggests that by rearranging the layout of the pages or the link targets in the pages, approximately 1/5 of the positive training instances are likely to be different.

To neutralize the effect that the highly reachable products are more likely to be treated as the positive training instances, we should weaken the weights of the positive instances in which next clicked product is highly reachable. In other words, if many product pages have direct links to the product p_j, we should assign a small weight to the positive training instance (x_i, p_j). Another possible method is to include the positive training instance (x_i, p_j) only when such a behavior is *spontaneous*, i.e., such a click was not influenced by the layout of the page. If we push the idea to the limit, we may include the information that p_i leads to p_j only when 1) the page of p_i contains no direct links to p_j, and 2) the p_j is not included in the recommendation list of the page of p_i. Thus, the positive signal that $p_i \to p_j$ is less likely to be affected by the layouts of the pages.

[1]Due to business sensitivity, we are not allowed to share the name of the company.

3.3 Experiment

To support the above claims, we simulated the typical method to generate the training data and the test data. This section presents the detailed settings, which is very close to what we introduced in Section 2. We present and discuss the results at the end of this section.

3.3.1 Experiment setting

We simulated an EC website that contains $1,000$ products $p_0, p_1, \ldots, p_{999}$. Each product is randomly assigned to one out of the ten product categories $c_0, c_1, \ldots, c_9$. We generated the similarity score between every pair of the products by the following rule: if two products p_i and p_j belong to different product categories, $s(p_i, p_j)$ the similarity score between them is sampled from a uniform distribution between 0.01 and 0.35. Otherwise, we sample $s(p_i, p_j)$ from a uniform distribution between 0.3 and 0.9.

The rest simulation follows the illustration of Figure 1. Initially (during t_0 and t_1), the website has no recommendation module. Each product page (p_i) contains direct links to 5 randomly selected products $(p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(5)})$, which we called "promoted products". We assume that when a user reads the page of product p_i, whether she/he will continue to view another product follows a Poisson distribution with a fixed λ. If she/he decides to continue, for 80% of the time the user would click on the promoted products and 20% of the time the user would visit a product p_j with probability proportional to $s(p_i, p_j)$ the similarity between p_i and p_j.

At t_1, the first recommendation module is online. We assume that if a user decides to stay on the site, then for 40% of the time she/he would click on one of the ten items returned by the online recommender system, 40% of the time the user would click on the promoted products, and for 20% of the time she/he would click the next item p_j proportional to the similarity score between p_i and p_j.

We generated 1,000,000 sessions that browse the products based on the above rules. We set the initial recommendation model to a simple approach CategoryTP (Categorical

Top Popular): when a user is viewing the product p_i whose product category is c_j, we recommend the top-10 popular products in the category c_j.

Now suppose we would like to try other recommendation algorithms. To compare the new algorithms with CategoryTP, we take the logs between t_1 and t_2 and separate the first 700,000 sessions as the training data and the last 300,000 sessions as the test data. We train each recommendation module (including the initial recommendation module CategoryTP) by two types training data: 1) train-all – all the 700,000 training logs, and 2) train-sel – if a user views p_i followed by p_j, the information is included in the training data only when p_j is neither included in the promoted products nor the recommendation list.

We compared the CategoryTP with the following methods: (1) TotalTP (Total Top Popular) : obtain the items that are mostly viewed during the training period and always recommend these items during the test period; (2) MC (Markov Chain): build a table to record the transition probability from one product to another during the training period and make recommendation in the test period based on the table, i.e., recommendation based on the most likely transferred product from the current browsing product; (3) ICF-U2I (Item-based Collaborative Filtering based on the User-to-Item matrix): during the training period, generate a user-to-item matrix in which the entry (i, j) represents the number of times a user i views a product j, and compute the cosine distance between every pair of the columns (which represents product features). During the test period, the algorithm recommends the columns that are most similar to the current browsing product; (4) ICF-I2I: (Item-based Collaborative Filtering based on the Item-to-Item matrix): during the training period, generate an item-to-item matrix in which the entry (i, j) represents the number of times item i and item j are viewed by the same user and compute the cosine distance between every pair of the columns (which represents product features). During the test period, the recommendation policy is the same as ICF-U2I, i.e., recommending the columns that are most similar to the current browsing product; (5) NMF-U2I: during the training period, generate a user-to-item matrix as explained in ICF-U2I and perform non-negative matrix factorization on the matrix to get the hidden vector of each item. Next, the algorithm calculates the cosine distance between every pair of these vectors. The recommendation policy during the test period is the same as ICF-U2I; (6) NMF-I2I: during the training period, generate an item-to-item matrix as explained in ICF-I2I and perform the non-negative matrix factorization on the matrix, as explained in NMF-U2I. The algorithm calculates the cosine distance and recommends items based on the distances during the test period, as explained in NMF-U2I.

3.3.2 Results

We show the average percentage of the promoted products appearing in the top-10 recommended items given the user is browsing a specified product. As demonstrated in Table 2, when using train-all (all the available training data) as the training data, several algorithms recommend many of the "promoted products". As a result, we seem to learn the "layout" of the product page (i.e., the direct links from p_i to p_j) instead of the intrinsic relatedness of between products. On the other hand, when we only utilize the train-sel (the data that match the rules specified in Section 3.2) as the training data, the ratio of the promoted products appearing in the recommendation list is much lower. In order not to let the layout of the product pages influence the learned rules, we probably should use the train-sel as the training data, or perhaps decrease the weights of the highly reachable products.

3.4 Lessons learned

The common wisdom that the clickstream represents a user could be problematic because the clickstreams are highly influenced by the reachability of the products and the layouts of the product pages. The items that occupy many spaces are more likely to be clicked and reached. As a result, training a recommender system based on the clickstreams are likely to learn (1) the "layout" of the pages, and (2) the recommendation rules of the online recommender system. Ideally, we should keep only the spontaneous clicks in the log to, at least partially, solve or bypass this issue.

4. ISSUE 2: THE ONLINE RECOMMENDATION ALGORITHM AFFECTS THE DISTRIBUTION OF THE TEST DATA

This section shows that evaluating the recommendation algorithms based on the logs of the clickstreams may be problematic if the test data are poorly selected. Like the previous section, we will start by explaining the problem and propose possible ways to bypass the problem. We will conduct experiments to support our claim.

4.1 The core problem

If we follow the procedure in Section 2 to generate the test data, the click through rate of the new proposed recommendation algorithm R_{new} is very likely to be worse than the online algorithm R_{orig}. The fundamental problem of such a setting is that, if the suggested product list L_{new} recommended by the new recommendation module R_{new} is very different from the online recommendation module's list L_{orig}, the online users have no chances to click on the products that appear only in L_{new} but not in L_{orig}. As a result, the recommendation modules that suggest products close to the online module tend to perform better.

4.2 Selecting proper test data

In several studies, researchers utilized the clickstreams as the ground truth for evaluating recommender systems, as we will demonstrate in Section 7. Specifically, given the current context feature x_i and the user's next visited product p_j, we treat a recommendation to be successful if the recommended list contains p_j. As we explained above, if a product p_k appears in L_{new} but not in L_{orig}, the product p_k is less likely to be clicked even though p_k might be a great recommendation given the context feature x_i.

If we use the entire clickstream logs as the ground truth for evaluation, the online recommendation algorithm is always favored, since the next prouct p_j is clicked probably because it is included in the recommendation list returned by the online recommendation module and naturally has a higher chance to be clicked. Instead of using the entire clickstreams, we probably should include the instance (x_i, p_j) to the test dataset only when the click happens spontaneously. Similar to the discussion in Section 3.2, if we push the concept to the limit, we should only include (x_i, p_j) to the test dataset

Table 2: The table shows the ratio of the number of the promoted products that appear in the recommendation list when using train-all or train-sel as the training data (each recommendation algorithm returns the top-10 results).

method	MC	CategoryTP	TotalTP	ICF-U2I	ICF-I2I	NMF-U2I	NMF-I2I
train-all	100.00%	1.48%	1.84%	93.22%	1.40%	1.48%	1.34%
train-sel	1.08%	0.86%	0.98%	14.46%	1.28%	1.32%	1.24%

if 1) p_j is not directly linked from the current browsing page, and 2) p_j is not included in the recommendation list given x_i as the input of the online recommendation algorithm.

4.3 Experiment

This section demonstrates that the online recommendation algorithm highly influences users' clicks.

4.3.1 Experiment setting

The entire simulation process is very similar to Section 3.3. Specifically, we first use the CategoryTP as the online recommendation algorithm and report the click through rate of various recommendation algorithms based on two test datasets: (1) test-all – the entire clickstream logs, and (2) test-sel – the clickstreams that match the conditions described in the last section. Next, we change the online recommendation algorithm to TotalTP and repeat the same experiment.

4.3.2 Results

Table 3 and Table 4 show the experimental results when the online recommendation algorithms are CategoryTP and TotalTP respectively. For each compared method in each table, we show four results: (1) training by train-all and evaluating by test-all; (2) training by train-all and evaluating by test-sel; (3) training by train-sel and evaluating by test-all; (4) training by train-sel and evaluating by test-sel.

If we use train-all for training and test-all for evaluation, the recommendation algorithm that is the same as the online recommendation algorithm always yields great results. For example, when we use CategoryTP as the online recommendation algorithm, the click through rates of (offline) CategoryTP and TotalTP are 37.61% and 5.41% respectively. However, when we switch the online algorithm to TotalTP, the click through rate of CategoryTP drops to 5.28%, and the click through rate of (offline) TotalTP increases to 40.80%. This demonstrates the issue we discussed in Section 4.1 – the online recommendation algorithm and the ones similar to the online recommendation algorithm tend to outperform the others if the training and the test datasets are selected poorly. Since several previous studies simply use all the available test dataset for evaluation, the results are likely to bias toward the online algorithm. As a result, their reported results are questionable.

When we use train-all for training and test-sel for evaluation, the models probably partially learned the recommendation rules of the online recommendation algorithm. If the online algorithm is a mediocre algorithm (e.g., TotalTP), the proposed algorithms are likely to learn from bad examples. Thus, the models are probably not generic enough to perform well in the general cases. As demonstrated in Table 4, all the (train-all, test-sel) are worse than the (train-sel, test-sel) results given the same proposed recommendation algorithm. On the other hand, if the online recommendation algorithm mostly offers great recommendations, even a mediocre algorithm may learn from good examples (obtained from train-all). However, the good results do not result from the ability of the model itself, but from the good training data. As a result, the reported numbers may still be questionable.

When we use train-sel as the training data and test-all as the test data, the evaluation ground truth is affected by the online recommendation algorithm and the link structure between the product pages. As demonstrated, when using CategoryTP as the online algorithm, the click through rates of (offline) CategoryTP and TotalTP are 4.36% and 0.71% respectively. However, when using TotalTP as the online algorithm, the click through rate of CategoryTP drops to 0.86% and the (offline) TotalTP increases to 1.47%. Indeed, the online algorithm affects the distribution of the test dataset.

Finally, we believe that the proper training dataset and test dataset should be train-sel and test-sel respectively. In this case, the online training model does not affect the generation of the training data. Thus, the proposed methods will not learn from the examples that are affected by the online learning algorithm. Meanwhile, the online training algorithm does not affect the generation of the test data either. Thus, we may evaluate the proposed method based on an unbiased test dataset.

4.4 Lessons learned

Previous studies sometimes use all the available test data as the ground truth for evaluation. Unfortunately, such an evaluation process inevitably favors the algorithms that suggest products close to the online recommendation algorithm. We should carefully select the test dataset to perform a fairer evaluation.

5. ISSUE 3: CLICK THROUGH RATES ARE MEDIOCRE PROXY TO THE RECOMMENDATION REVENUES

5.1 Core problem

Several academic studies on recommender systems exploit the click through rate to compare different recommendation algorithms. This metric is user-centric, i.e., it measures a user's satisfaction about a recommendation. Click through rate is popular, probably because the industry hesitates to share or release revenue-related information. As a result, researchers mostly can only study users' feedback and satisfaction on a recommender system and hope that boosting user-centric measures (e.g., click through rate) will increase the business-centric measures (e.g., recommendation revenue). Unfortunately, such a surmise was not carefully validated.

Table 3: The click through rates when the online recommendation algorithm is CategoryTP.

	CategoryTP		TotalTP		MC		ICF-U2I		ICF-I2I		NMF-U2I		NMF-I2I	
	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel
test-all	37.61%	4.36%	5.41%	0.71%	58.89%	4.05%	46.88%	17.02%	28.37%	21.79%	16.54%	5.78%	8.78%	16.19%
test-sel	2.76%	2.75%	1.02%	1.03%	1.88%	2.86%	1.60%	3.03%	2.38%	2.66%	2.54%	1.74%	2.52%	2.73%

Table 4: The click through rates when the online recommendation algorithm is TotalTP.

	CategoryTP		TotalTP		MC		ICF-U2I		ICF-I2I		NMF-U2I		NMF-I2I	
	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel	train-all	train-sel
test-all	5.28%	0.86%	40.80%	1.47%	60.40%	1.22%	48.61%	16.40%	23.85%	10.77%	13.29%	0.80%	1.53%	4.17%
test-sel	2.66%	2.89%	0.93%	1.18%	1.01%	2.98%	1.04%	2.62%	1.00%	2.07%	1.08%	2.39%	1.12%	2.03%

Figure 3: The relationship between the click through rate and the recommendation revenue (from 2014/9/2 to 2015/9/13, each point represents one date). We exclude the tick marks because we are not allowed to report their exact values. The correlation of determination is only 0.089, suggesting that they have a weak positive relationship.

5.2 Experiment

5.2.1 *Experiment setting*

We selected two measures to represent the user-centric and the business-centric metrics: the click through rate and the recommendation revenue. The click through rate is the proportion that a recommendation is clicked. This metric is probably the most typical user-centric measure. The recommendation revenue is the income contributed by the recommendation module.

We collected a year-long log (2014/9/2 - 2015/9/13) from a large EC website in Southeast Asia. For each day, we calculated the click through rate and the recommendation revenue. Note that during the year of the study we launched and retired different recommendation algorithms (e.g., Matrix Factorization, Markov Chain, CategoryTP, a mixture of several methods, etc.) on different pages (e.g., the main page, the product category page, and the product page). As a result, we measure the click through rate and the recommendation revenue across many different settings.

5.2.2 *Experiment results*

Figure 3 presents the relationship between the click through rate of the recommendations and the recommendation revenue. The correlation of determination (R2) between the click through rate and recommendation revenue is only 0.089, suggesting that they have a very weak correlation. Therefore, it could be improper to simply pursuing click through rate and hope that increasing click through rate will boost the revenue.

5.3 Lessons learned

Based on the result, comparing different recommendation modules purely based on the user-centric metrics, such as the click through rate, may fail to capture the business owner's satisfaction. Unfortunately, studies on recommender systems mostly perform comparisons based on the user-centric metrics. As a result, even if a recommendation algorithm attracts many clicks, we cannot assure this algorithm will bring a large amount of revenue to the website.

6. ISSUE 4: EVALUATING RECOMMENDATION REVENUE IS NOT STRAIGHTFORWARD

EC companies build recommendation modules in the hope that these modules will discover users' purchasing intentions and eventually boost the revenue. It is reported that a large portion of an EC website's revenue comes from recommendation [20]. In this section, we show that evaluating recommendation revenue is not as straightforward as it first looks.

6.1 The core problem

We ask a more fundamental and probably more challeng-

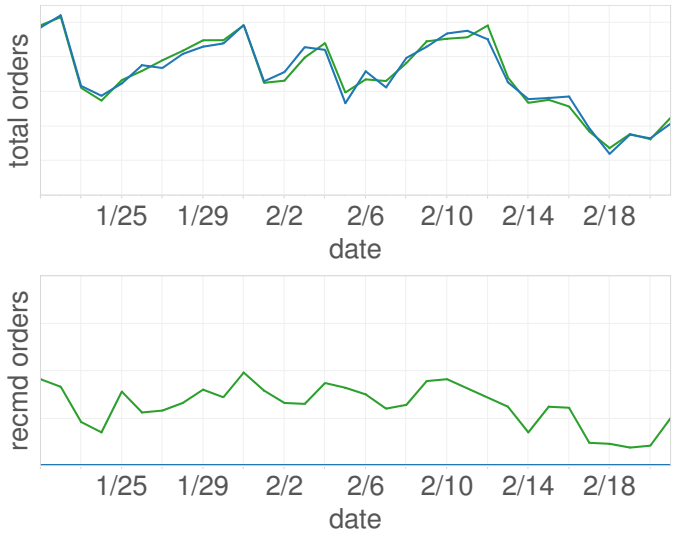

Figure 4: A comparison of the number of the total purchases in two cases: displaying the recommendation panel or not. Green line: the channel with a recommendation panel; blue line: the channel without a recommendation panel. Due to business sensitivity, the tick labels of the y-axes are removed. It appears that the recommendation panel brings little *extra* revenue, if any, to the channel.

ing question: does a recommendation module bring *extra* revenue (or *extra* orders) to an EC website? Although we can measure the number of purchases contributed by a recommendation module, we cannot tell whether these buyers would still purchase these (or possibly similar) items without the recommendation module. It is possible that the recommendation modules are served as a convenient tool for users to locate the desired items, but even without the recommendation module, the users can still discover these items through another user interface provided by the website.

6.2 Experiment

6.2.1 *Experiment setting*

To answer the question, we propose to conduct experiments based on A/B testing to control the appearance of the recommendation module. Specifically, we collaborated with a large EC website in Southeast Asia. We directed 5% of users to a channel that displays no recommendation on the product page (channel 1), and another 5% of users to a normal channel that displays recommendations as usual (channel 2). If the *total* purchases of channel 1 is very similar to the total purchases of channel 2, then the recommendation module brings no *extra* purchases to the website. In this case, the recommendation module probably simply provides another way for the users to discover the items of their interests.

6.2.2 *Experiment results*

Figure 4 shows the total purchases and the recommended purchases of the two channels. We use blue line to display users' orders in channel 1 (the no recommendation channel) and the green line for channel 2 (the normal channel). The lower sub-figure of Figure 4 shows that users indeed purchase recommended items. This seems to suggest that recommendation is helpful in increasing the number of sales. However, if we compare the total number of purchases of each channel (the upper sub-figure of Figure 4), we see no obvious benefit of having the recommendation panel. It appears that, for most of the times, the users who purchases recommended items still purchase items even without the recommendation module.

6.3 Lessons learned

Based on the result, we suspect that, although a recommendation module may help users quickly discover their needs, these users, even without the recommendations, may still be able to locate the desired products by other processes, e.g., by querying through the search bar, or by navigating through the hierarchical table of contents. Thus, it is not clear whether a recommendation module brings extra purchases, or simply re-direct users from other purchasing processes to recommendation. In an extreme case, an EC company can fill in the entire page with recommendations and claim that nearly 100% of their revenue comes directly from recommendations. Apparently, such a claim is misleading. To proper evaluate the extra revenue contributed by a recommender system, we still need to leverage on A/B testing. Unfortunately, it is very difficult for the researchers in academia to perform A/B testing in practice.

7. RELATED WORKS

7.1 Common metrics to evaluate recommender systems

Most studies on recommender systems compare different algorithms based on user-centric metrics, which can be categorized into the following types: accuracy-based, diversity-based, and surprisal-based.

Typically, we would like a recommendation module to accurately predict a user's preference on items. Thus, accuracy is a natural choice to measure recommendation modules. A simple way to measure accuracy is click through rate – in what percentage a user clicks a recommendation [10]. However, such a metric may favor the algorithms that tend to recommend popular items, because recommendation accuracy usually declines towards the long tail [25]. In addition to accuracy, researchers have found that diversity plays an important role in improving users' satisfaction about recommendation [5]. To quantify "diversity", one can measure the average dissimilarity between all pairs of recommended items based on these items' attributes, such as their brands, prices, or taxonomy. Diversity and accuracy are usually a trade-off. One can easily increase diversity by recommending unrelated items, but this usually sacrifices the accuracy. Surprisal, sometimes known as novelty, is a type of user-centric metric that is relevant to diversity. Previous studies have inconsistent definitions about surprisal. As a result, the terms "diversity", "surprisal", "serendipity", and "coverage" are sometimes interchangeable. Here, we define surprisal as the unexpectedness of a recommendation. Thus, one can define the surprisal as the inverse of an item's popularity. Such an idea is illustrated in [27] with a simple modification.

Several studies aimed to optimize a combination of the above metrics because very often we can expect a tradeoff between these metrics. Instead of proposing another metric or trying

to maximize these metrics, we show that many studies probably incorrectly reported these metrics. We also show that user-centric metrics may be inconsistent with the business-centric metrics, such as recommendation revenue.

There are studies aimed at unbiased offline evaluation for recommender systems based on contextual-bandit [18; 17]. These methods mostly assume the researchers can control the online recommendation module to explore the uncertain cases. Unfortunately, most researchers have no access to a live large-scale information system. They mostly rely on the datasets released by the large companies.

7.2 Reviewing Previous Competitions and Publications

We review previous competitions and publications that may fall into some of these pitfalls.

RecSys Challenge 2015 [6] offered the clickstreams of sessions from a big retailer which utilizes the recommender system service provided by YOOCHOOSE. Some of these sessions include the purchase events. The goal of the challenge is to predict the purchased item (if any) given the clickstream of a new session. Such a dataset is indeed valuable resource for the researchers in academia. However, using the clickstreams as the training data may generate the models that tend to recommend the highly reachable items, as discussed in Section 3. Additionally, the challenge utilized users' logs to perform offline evaluation, which may also be problematic, as explained in Section 4. The similar problems appear in several challenges, such as the RecSys Challenge 2016 [26], the Display Advertising Challenge provided by Criteo Labs in 2014 [3], the Click-Through Prediction Challenges provided by Avazu in 2015 [2] and by Outbrain in 2017 [4]. The studies that utilized these datasets for model generation and evaluation may also suffer from similar problems. Recently, some competitions started to use (or partially use) the online events for evaluation. For example, the RecSys Challenge 2017 consists of two stages – the offline and the online phase. During the online evaluation phase, the recommendations proposed by each team are rolled out to the live system. As a result, the real users have chances to interact with the recommended items [1].

Many studies on recommender systems aimed at predicting users' ratings to items or increase the click through rate. However, clicking (browsing) and buying could be very different behaviors. As shown in [13], this two types of behaviors can be classified by a supervised learner. Thus, simply pursuing the click through rate may not necessarily increase the business runner's revenue. For example, given a list of recommended items, a user may be encouraged to continue browsing these items instead of purchasing. Unfortunately, since the revenue related information is usually business sensitive, the business runners may not want to share such information with outsiders.

It is reported that 35% of Amazon's product sales are from recommendation [20]. However, how to derive such a number was unspecified. If, for example, an EC website fills most of the pages with recommendations, it is not surprising that most of the clicks and the purchases are directly or indirectly from the recommendations. To ensure the (extra) purchases or the (extra) revenue coming from a recommendation module, we believe that one of the most reliable tool is A/B testing. Unfortunately, applying A/B testing requires to have a large platform with many users. This is not always available for researchers, especially those in universities.

8. DISCUSSION AND FUTURE WORK

This paper shows four pitfalls that may occur in several studies and competitions of recommender systems. Specifically, the first two issues are due to the biased data collection of the training and the test datasets. These two pitfalls should be concerned not only by the researchers and practitioners of the recommender systems but also the data scientists in general. However, we mainly focus on the field of recommender systems in this paper because, unlike some machine learning problems in which the distribution of the training and the test datasets are affected only to a small extent by the data collecting approach, the distribution of the training and test dataset of recommender systems are highly influenced by the way they are collected. The third issue is regarding the proper selection of the evaluation metrics. Again, every data scientist should aware of the issue, but such a problem is less discussed in the field of recommender systems due to its nature – the revenue related information is usually business sensitive and protected. Finally, the fourth issue is even less addressed: even if the recommender systems indeed bring purchases, the purchases may not be the *extra* purchases. As a result, the recommender systems may improve customers' user experience, but may not bring immediate extra revenue to a business runner.

For future work, we would like to build an open platform in which researchers may register and plug their recommendation algorithm onto the platform. The platform redirects traffic into different registered algorithms in the backend. Thus, every recommendation algorithm is served online. The researchers do not need to worry about the offline evaluation issues. Such a platform may, at least partially, address the second and the fourth issue discussed in this paper. We are currently developing such a system and negotiating with several EC companies to try the system. We hope to open the system as an arena so that researchers and practitioners may compare their recommendation algorithms in a more realistic environment.

9. ACKNOWLEDGEMENTS

We appreciate partial financial support from the Ministry of Science Technology (MOST 105-2218-E-008-015) and the Industrial Technology Research Institute (ITRI 106-W100-21A2). We are grateful to the National Center for High-performance Computing for computer time and facilities.

10. REFERENCES

[1] ACM RecSys Challenge 2017. `http://2017.recsyschallenge.com/`. Accessed: 2017-07-14.

[2] Click-through rate prediction. `https://www.kaggle.com/c/avazu-ctr-prediction`. Accessed: 2017-07-14.

[3] Display advertising challenge. `https://www.kaggle.com/c/criteo-display-ad-challenge`. Accessed: 2017-07-14.

[4] Outbrain click prediction. `https://www.kaggle.com/c/outbrain-click-prediction`. Accessed: 2017-07-14.

[5] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *Proceedings of the second ACM international conference on web search and data mining*, pages 5–14. ACM, 2009.

[6] D. Ben-Shimon, A. Tsikinovsky, M. Friedmann, B. Shapira, L. Rokach, and J. Hoerle. Recsys challenge 2015 and the yoochoose dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 357–358. ACM, 2015.

[7] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[8] H.-H. Chen, L. Gou, X. Zhang, and C. L. Giles. CollabSeer: a search engine for collaboration discovery. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*, pages 231–240. ACM, 2011.

[9] H.-H. Chen, I. Ororbia, G. Alexander, and C. L. Giles. ExpertSeer: a Keyphrase Based Expert Recommender for Digital Libraries. *arXiv preprint arXiv:1511.02058*, 2015.

[10] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[11] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *Advances in neural information processing systems*, pages 385–392, 2008.

[12] Y. Goldberg and O. Levy. word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[13] Q. Guo and E. Agichtein. Ready to buy or just browsing?: detecting web searcher goals from interaction data. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 130–137. ACM, 2010.

[14] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004.

[15] Y. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin. Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 43–50. ACM, 2016.

[16] Y. Koren, R. Bell, C. Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[17] L. Li, S. Chen, J. Kleban, and A. Gupta. Counterfactual estimation and optimization of click metrics in search engines: A case study. In *Proceedings of the 24th International Conference on World Wide Web*, pages 929–934. ACM, 2015.

[18] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306. ACM, 2011.

[19] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[20] I. MacKenzie. How retailers can keep up with consumers. `http://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers`. Accessed: 2017-07-14.

[21] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[22] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[23] P. Romov and E. Sokolov. Recsys challenge 2015: ensemble learning with categorical features. In *Proceedings of the 2015 International ACM Recommender Systems Challenge*, page 1. ACM, 2015.

[24] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[25] H. Steck. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 125–132. ACM, 2011.

[26] W. Xiao, X. Xu, K. Liang, J. Mao, and J. Wang. Job recommendation with hawkes process: an effective solution for recsys challenge 2016. In *Proceedings of the Recommender Systems Challenge*, page 11. ACM, 2016.

[27] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515, 2010.

Intelligent Disaster Response via Social Media Analysis - A Survey

Tahora H. Nazer*, Guoliang Xue*, Yusheng Ji†, and Huan Liu*
*Arizona State University, USA
†National Institute of Informatics, Japan
{tahora.nazer,xue,huan.liu}@asu.edu and kei@nii.ac.jp

ABSTRACT

The success of a disaster relief and response process is largely dependent on timely and accurate information regarding the status of the disaster, the surrounding environment, and the affected people. This information is primarily provided by first responders on-site and can be enhanced by the first-hand reports posted in real-time on social media. Many tools and methods have been developed to automate disaster relief by extracting, analyzing, and visualizing actionable information from social media. However, these methods are not well integrated in the relief and response processes and the relation between the two requires exposition for further advancement. In this survey, we review the new frontier of intelligent disaster relief and response using social media, show stages of disasters which are reflected on social media, establish a connection between proposed methods based on social media and relief efforts by first responders, and outline pressing challenges and future research directions.

1. INTRODUCTION

Social media is a new way of communication in the course of disasters. A major difference between social media and traditional sources is the possibility of receiving feedback from the affected people. Responders such as Red Cross can benefit this two-way communication channel to inform people and also gain insight by monitoring their posts. Twenty million tweets after Hurricane Sandy (2012) and eight million tweets after the Boston Marathon Bombings (2013) [29] have been published on Twitter. This swarm of posts can provide valuable insight and help with the disaster management when the functioning of a community is disrupted due to severe fatalities and infrastructural damage [1; 12].

Two types of insight can be obtained from social media in the course of disasters. The "big picture" is an estimate of the scope of the disaster: area, casualties, and failed structures. "Insightful information" is more detailed and is available when more data is available on social media. Locations that need food, medical supplies, or blankets are examples of insightful information [71].

One challenge associated with acquiring insight via social media is processing enormous amount of information in a timely manner. After Japan earthquake and tsunami (2011), 1,200 tweets were published every minute from Tokyo [2] and after Hurricane Sandy (2012), the peak rate of 16,000 tweets per minute has been reported [60]. This amount of data is too large to be manually processed by emergency responders. Some of the proposed methods to overcome this issue are presented in Section 3.1.

Another challenge is the wide-spread of unwanted content such as daily chatter, spam, and rumor in social media. Among the 8 million tweets related to Boston Marathon Bombings (2013), 29% were found to be rumors and 51% to be generic opinions and comments [29]. Moreover, exploiting bots has worsened this issue. Large number of bots can be generated in a short period of time and be used to spread spam, deviate the conversation of real users, and help with the virality of rumors. We mention some of the solutions to this challenge in Section 3.2.

Disasters have eight socio-temporal stages: Pre-disaster, Warning, Threat, Impact, Inventory, Rescue, Remedy, and Recovery [76]. The volume of social media posts varies in each stage; majority of users start posting after the disaster onsets and the frequency decreases when the disaster reaches its final stages. Availability of data is a major factor in building automatic methods for facilitating the management tasks. Hence, we consider four stages in disasters; the ones in which social media posts are dense enough for Machine Learning methods to achieve reliable results: Warning, Impact, Response, and Recovery.

In the warning stage (Section 4), social media can be used as a complementary source of information to help increase the confidence in predicting disasters and providing warnings. Changes in the frequency of posts with specific words and topics, activity patterns of users [5; 83; 92], and sentiment of posts [63] are used to predict disasters. Predicting disasters before they hit an area provides the opportunity to warn people in danger and evacuate elevators and operation rooms. Currently, USGS uses tweets to check the accuracy of sensor reports and detect earthquakes in a shorter time. Earthquakes can be detected using tweets by 60 seconds earlier than sensors; this time is valuable for warning areas in danger and starting evacuation processes [26].

When disasters impact an area, social media posts show anomalies such as changes in the language. A study [19] on LiveJournal after 11 September 2001 shows that emotional positivity decreased and cognitive processing, social orientation, and psychological distancing increased after the attack [10]. These changes in social media posts can be quantitatively captured in sentence level or topic level. Capturing the change in real-time results in detecting disasters before they are announced by official sources, governmental websites, or major news outlets [82] (see Section 5).

In response to the chaotic environment caused by disasters,

emergency responders want to acquire actionable insight and a big picture of the disaster [14]. Detecting and tracking topics, trends, and memes on social media provides information regarding the status of disasters and the affected people. Damages, casualties, missing animals, and failed structures are some of the topics that people discuss on social media. Tracking these topics, discovering the trends, and monitoring mentioned locations help responder distribute resources more efficiently (see Section 6).

Volunteers are significantly important in the relief process. They post information that increases situational awareness (e.g. status of roads and damages to built structures) and provide technical support for translating social media posts and geotagging them. Some of the systems that exploit social media posts to facilitate disaster management are Ushahidi [69], AIDR [38], and TweetTracker [50] which will be discussed in Section 7 with more details.

In this paper, we clarify the relation between the stages of disasters and relevant research on social media. This effort is towards unwrapping the potentials of social media to be exploited by emergency responders to a larger extent. We consider four stages for disasters which are widely reflected on social media: warning, impact, response, and relief. In each stage, we introduce approaches that use social media to ease the relief efforts. The major difference between this work and previous surveys in the field is the organization of the material in an effort toward facilitating the exploitation of these methods by disaster responders. We use disaster management stages used by first responders [84] to explain limitations and potentials of social media research. We bold available methods and tools that can be used by responders and mention the areas in which social media has not been used to its potential. Moreover, we focus on more recent areas which are not widely reflected in previous studies. We believe that this work establishes a connection between available tools based on social media data and the efforts of first responders.

Contributions of this paper are as follows:

- Introducing four stages for disasters based on activities of users on social media.
- Categorizing research on social media for disaster management based on their application in each stage.
- Connecting the research on social media with disaster management efforts by first responders.
- Including recent studies on social media in this area that have not been included in similar efforts.

2. CHARACTERISTICS OF DISASTERS

Lifecycle of disasters consists of several stages. Powell [76] considers eight socio-temporal stages for disasters: pre-disaster, warning, impact, inventory, rescue, remedy, and recovery. Hill [35] also introduces four coarse-grained stages of warning, impact, reorganization, and change. In a model by Office of US Foreign Disaster Assistance [84], disaster management lifecycle stages are Hazard Analysis, Vulnerability Analysis, Mitigation & Prevention, Preparedness, Prediction & Warning, Response, and Recovery.

Hazard Analysis is concerned with studies on disaster histories and scientific analysis of different disasters. The goal is achieving a thorough understanding of each disaster and

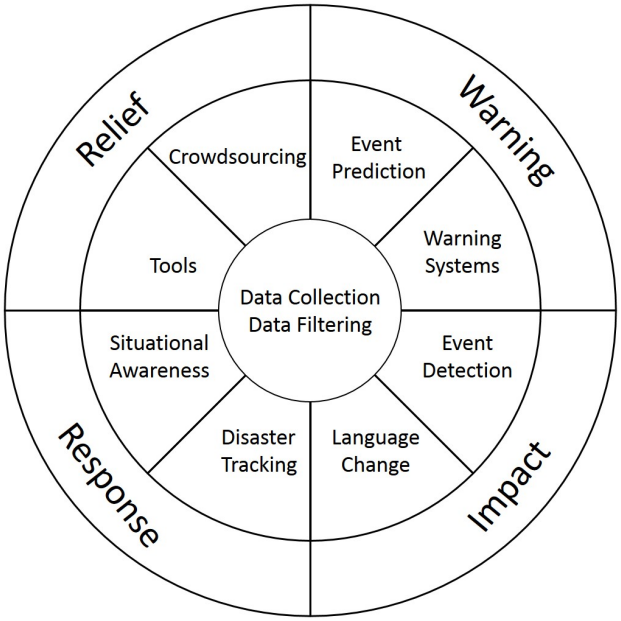

Figure 1: Socio-Temporal stages of disasters which are reflected on social media.

how it can affect land, weather, agriculture, and environment. Moreover, malignant effects such as spread of disease, air pollution, and water contamination are studied in this phase. Based on the research in this phase, responders have knowledge about possible outcomes of disasters.

In Vulnerability Analysis the focus is the area and people that are affected by a disaster. Based on the historic record and hazard analysis reports, responders can estimate types and extent of possible damages of a disaster to specific location. Survey and community experience reports can also help with such estimations.

Mitigation and Prevention is about establishment of rules, regulations, and standards that help the community to reduce risks. Land use regulations and building standards are examples of such efforts. To mitigate the risk, organization of relief groups is pre-defined and well-documented.

Community planning is performed in Preparedness phase. In this stage, communication infrastructure is built and procedures that should be followed after a disaster are defined. Resources, such as food, water, clothes, and medicine, are stockpiled in storages. The community is also prepared by receiving awareness about hazards and actions.

Prediction and Warning is using technology and interpretation methods to forecast disasters and provide early warnings. This phase requires close tracking of disasters and communication with affected areas on the route of the same disaster. Warnings can result in public responses such as evacuation and moving to safe shelters.

Response starts after the disaster onsets. People will be moved to shelters and rescue process for missing persons will start. Responders begin to assess needs of the affected people to make decisions regarding the distribution of resources. Damage to built structures will be estimated and first responders will scatter accordingly.

In the final step, Recovery happens through rehabitation

and reconstruction. During this period, affected citizens are in a stable situation and the aim is returning to normal life. Some activities are rebuilding failed structures, providing temporary/permanent housing, reestablishment of agriculture, and securing water sources.

Although people use social media in all stages of disasters, some stages receive more attention. Volume of data is a major factor in methods that use social media posts. Hence, the phases for witch social media can be used is dependent on how active social media users are during that period of time. The stages that are highly reflected on social media are Warning, Impact, Response, Relief.

Four mentioned stages of disasters are shown in Fig. 1. Warning is facilitated by event prediction and warning systems that use social media data. Impact is the time at which disaster hits the area. Onset of disasters can be detected using changes in the behavior of social media users such as language change. In Response phase, occurrence of the event has been confirmed and social media can be extensively used to gain situational awareness and track changes in the status of disaster and affected people. Relief is the stage in which volunteers are engaged to empower tools that facilitate relief efforts. Social media provides a platform to share information and arrange volunteer efforts. In the following sections, we extensively discuss each of the aforementioned stages and how social media can be used to facilitate the efforts of disaster responders.

Valuable insight that is obtained from social media during disasters, in all four stages, is highly dependent on data and its quality. Numerous posts are published in coarse of disasters, however, they are a mixture of informative and non-informative posts. Non-Informative posts can be in form of rumor, spam, bot-generated content, and daily chatter. These posts need to be removed before any anlysis is performed. Hence, the core of Fig. 1 is data collection and filtering which is required for all the four stages.

3. DATA EXTRACTION AND FILTERING

Data collection and filtering is the core of disaster management using social media. Algorithm that are used for warning, detecting the impact, relief, and response all depend on the posts which are published on social media and their quality. Two tasks need to be performed in this regard: maximizing the amount of relevant data to the disaster and removing non-informative posts.

3.1 Data Extraction

Disaster-related tweets are extracted using lexicon-based [39; 78] or location-based [58] methods. The former uses a set of keywords that are generated by experts and the latter collects all the tweets that are associated with a specific location. Tweets that are filtered using keywords are only a fraction of all the disaster-related ones [13] and tweets with location information are quite rare. Hence both methods lack completeness and have low coverage.

3.1.1 Lexicon Unification and Extension

One way of increasing the visibility of disaster-related tweets is extending expert-defined keyword sets. For example, "CrisisLex" [70] is a a lexicon that increases the portion of disaster-related tweets that are captured from the Twitter Streaming API. The effort is towards finding one set of keywords which are extensively used in different disasters (hurricane, tornado, flood, bombing, and explosion). The process starts with extracting the tweets that contain any word from a set of expert-suggested keywords. These tweets are manually labeled to remove the ones which are not releated to the disaster. From the crisis-related tweets, words and phrases (consisting of two words) that appear in at least 5% of tweets form CirisLex. Another lexicon is "EMTerms 1.0"(CrisisLex and EMTerms 1.0 can be obtained from http://crisislex.org/crisis-lexicon.html) that includes more than 7,000 words categorized into 23 groups was introduced. Their method starts with the keywords of four major events and then extend the lexicon using Conditional Random Field (CRF) on another 35 disasters [90].

Another way is providing instructions for users on how to tweet regarding a disaster. Microsyntaxes, the instructions, unify the format of posts and make it possible for machines to automatically extract all the posts on a specific issue. "Tweak the tweet" [88] is a microsyntax introduced after Red River Floods, 2009. The authors show that visibility of disaster-related tweets increases when users are instructed to use hashtags such as `#fargo`, `#redriver`, and `#flood09`.

3.1.2 Location Estimation

To overcome the challenge of location sparsity in social media data, several methods have been proposed to estimate the location of posts or users. Content of posts, activity characteristics, profiles, and networks of users are exploited to estimate the location in which a user is based or the post is originated from. The granularity of estimation differs from one method to another. Some approaches estimate the coordinates, some remain in the city-level, and some only focus on a disaster areas that can be limited to a neighborhood or expand to several cities or states.

Content is frequently used to estimate the origin of posts. N-grams and "crisis-sensitive" features such as "in" prepositional phrase (such as "in Boston"), existential "there" (which usually describes an abstraction), and part-of-speech tag sequences are signals that discriminate in-region posts from out-region ones in course of a disaster [64]. Moreover, posts from a disaster area are less likely to include multiple hashtags, action words, and reference entities. Majority of such posts are original and contain URLs [51]. Posts from the same location frequently use similar words [16] and rarely use words that are used in other locations [32].

For locating users, the most intuitive features are geolocation or location field in their profile, the location of the websites that they linked to (which can be obtained using the IP or country code), time zone, and UTC24-Offset. These features can be combined using the stacking method [97] by considering an importance weight for each feature to find the most probable location of the user [85].

When location-indicating features are not available for a user, their location can be estimated using the location of users surrounding them. Backstrom et al. [7] observe that there is a power law relation between physical distance and the probability of existing a social link. Based on this finding, they propose a maximum likelihood prediction method that indicates the most probable location for a user given its neighbors. Based on triadic closure, if user a is connected to users b and c, b and c are more likely to be connected to each other [48]. In "Triadic heuristic" [43], the location of users is estimated as the geometric median of their neighbors who

are in triadic closure with them. Moreover, users are more likely to follow users nearby and more often mention the location in which they live [56].

Palen and Anderson [72] introduce the concept of "contextual streams" to combine Lexicon and location in order to overcome the issue of incompleteness. They use a set of broad terms (such as "frankenstorm" and "sandy" in the case of hurricane Sandy, 2012) to collect the first set of tweets. Then, they find the users who have geolocated tweets from their desired location. Finally, they collect their most recent 3200 tweets and extend their previous dataset. Using this information, they can compare the activities of users who are located on the site of the disaster before, during, and after its occurrence.

3.2 Data Filtering

Filtering social media data is a necessary process before exploiting it in any stages of a disaster. The data which is overwhelmed with unwanted content does not show real opinion of the crowd. Hence, any insight based on this fabricated data is flawed. Unwanted data such as rumors and spam have been studied extensively, however, prevalence of social bots has created new challenges.

3.2.1 Rumors

A rumor is an unverified piece of information that circulates in the situation of uncertainty by people who want to make sense of their situation [24]. Rumors, when verified, form two categories of true and false. False rumors can cause panic and distress especially after the onset of a major disaster. A bold example of false rumors is reported after a few cases of Ebola infection were reported in Newark, Miami Beach, and Washington DC. The rumors state that Ebola can spread through water, food, and air [57]. Also after hurricane Sandy, several false rumors have been observed. More than ten thousand tweets which contained fake images circulated after hurricane Sandy and the analysis show that top 30 users in this diffusion process were responsible for 90% of the tweets [30]. This observation verifies the fact that these images were injected into the stream of Sandy-related tweets by a small group of illegitimate users.

There are two mechanisms for manually correcting false rumors after a disaster. The first one is the self-correcting power of the crowd. According to disaster psychologists, social media systems will eventually correct erroneous information as both the sender of the information and other members of that community seek validation of their posts [22]. A study by Mendoza et al. [62] shows that majority of tweets which are related to a true rumor are affirming and this percentage is much lower in regard to false rumors. This suggests that Twitter community works as a collaborative filter for information. However, this mechanism is effective when a false rumor is not originated intentionally and enough time is given to this process to converge. In the aftermath of disasters such as hurricane Sandy, these conditions are not satisfied and so there is a need for emergency managers and officials to intervene which is the second method of correction. An example of this mechanism is the tweet which was posted by ConEdison energy company indicating that none of its employees have been trapped in a damaged plant and end the circulation of rumors about it (this tweet can be accessed via https://goo.gl/TC1Slx).

The aforementioned rumor correction methods are time-consuming and dependent on manual efforts which make them less effective as the social media becomes more prominent as the information channel after disasters. More than 300,000 tweets per minute after Japan Earthquake in 2011 and more than 20 million tweets after hurricane Sandy show the volume of data that needs to be processed to extract rumors. To overcome this challenge, methods have been proposed to automatically detect tweets in the swarm of social media posts by using their specific behaviors. For example, the diffusion process of rumors is different from those of normal posts. The originality of posts is lower (most of the posts are retweets). The number of users involved in the cascade is low in comparison to its virality. Also, the depth of the resulting cascades are lower than those of normal posts as users receive information by search and the posts, mainly, do not diffuse in the friendship network [30].

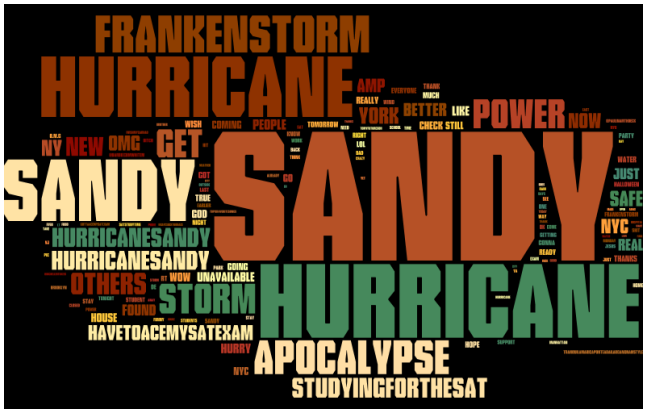

Figure 2: The wordcloud of keywords in tweets related to Hurricane Sandy. The size of each word corresponds to its surprise level in the area under study.

3.2.2 Spam

Spam is the "content designed to mislead or content that the sites legitimate users do not wish to receive [34]." Spammers degrade the credibility of social media and are capable of deviating the discussions and affect the popularity of topics. An example of such manipulations is promoting the material for Scholastic Assessment Test (SAT) that was happening close to the occurrence of Hurricane Sandy. The wordcloud (Figure 2) that has been generated using the surprise level of words [83] from Twitter data after this disaster shows `#HaveToAceMySATExam` as one of the keywords that stands out among thousands of others.

Social media sites such as Twitter and Facebook actively remove who violates their regulations and specifically spammers. The dominance of spammers becomes clearer when Twitter suspension included 50% of the accounts that were created in 2014 [47]. Spammers, although elaborate, show specific characteristics that discriminate them from legitimate users. Posting numerous posts on a handful (and even just one) topics throughout the lifetime of an account is a sign of spamming as marketing spammers advertise a specific product and publish many posts to gain visibility. Profile of spammers in most cases does not show their age and gender [102]. Their connections are different from normal users as the majority of their connections are not reciprocal [98].

3.2.3 Bot Generated Content

A malicious bot is a hijacked or adversary-owned account which is controlled by a piece of software. Bots, that can be automatically generated in large numbers, are overwhelming social media and leave major tracks. In the 2010 Massachusetts Senate Election, a candidate gained 60,000 fake followers on social media by exploiting bots. Such activities manipulate the opinion of the crowd by promoting a specific topic or supporting a specific figure. In these cases, trending topics and popular users are not necessarily real.

Three major methods are proposed to collect bots to observe and study. The prominent method is manual annotation which is expensive, time-consuming, and error-prone [17; 79; 100]. The second one is using the suspension mechanism of social media sites such as Twitter. There is no explicit cost associated with this method, however, it is time consuming and bot behavior is one reason for suspension. The process is sampling users, waiting for a period of time, and then re-examining the status of the sampled user. The ones that have been suspended in that period of time would be considered as bots [41; 91]. In the third method, a set of bots (honeypots) are created to lure other bots in the wild to interact with them. Honeypots are controlled by the researcher to tempt specific bots and avoid interference with the activities of normal users [53; 65].

4. WARNING VIA SOCIAL MEDIA

Accurate and timely warnings could prevent death rates by providing the time that is critical for evacuating elevators or halting medical operations. Warning systems have improved drastically in recent years but they are not perfect yet. In a recent case, Hurricane Matthew was reported as a category 4 hurricane as it approached the Florida coast [73], but it had one official U.S. landfall on the southeast of McClellanville, South Carolina as a category 1 hurricane [3]. Social media can be used as a complementary source of information to improve predicting events and providing warnings.

4.1 Event Prediction

Social media has been used for predicting events that will happen in near future. Forecasting the popularity of products, movie box-office, election results, and trends in stock markets are examples of such predictions [101].

Prediction is based on features of social media posts. Increase in the number of posts which are related to a topic [5] can be indicator of its future popularity. Changes in the patterns of using specific words in a area shows onset of an event [83]. Also, sentiment of posts can show future status of a product [63]. Crime prediction is also possible by semantic role labeling which is used for both finding the events and entities involved in them [96].

Prediction method based on the extracted features can vary based on the problem. Regression method have been used for prediction popularity of posts [89] but do not perform well when sentiment data is being used [103]. For predicting election results, Tumasjan et al. [92] use number of tweets mentioning political parties and their sentiments as indicators of popularity and political views toward them [92].

There is no prediction method with perfect accuracy. However, early detection of natural disasters reduces hazards in nearby locations. For example, quakes in areas with geographic proximity are used to predict earthquakes seconds before they happen [28]. This process has been used to build warning systems which will be explained in Section 4.

4.2 Warning Systems

"Warning systems detect impending disaster, give that information to people at risk, and enable those in danger to make decisions and take action" [87]. There has been a significant improvement in forecasting and warning systems especially for hurricane and earthquakes. Meteorologists can now forecast a hurricane 2 to 6 days before it hits an area and Global Seismic Network constantly monitors activity bellow Earth's surface. However, lack of complete data on natural hazards, monitoring instruments, and high dynamic nature of them keep accurate forecasting and warning a challenge [80] and "a 100% reliable warning system does not exist for any hazard [87]".

Social media facilitates is also used to deliver official and non-official warnings. Emergency managers and governmental organization post their warning messages via social media to be broadly accessed by the public [36]. Citizens also report warnings and advice about possible hazards [40].

One source of information that can be used to improve accuracy of warnings is data of built-in accelerometers in cell phones. This data can be used for quick detection of earthquakes and estimating their intensity and effect. The measurements by these sensors which are transmitted before the loss of communication are used for estimating the degree of damage to different areas; the task that can take up to an hour when performed by helicopters [27; 28].

Social media is another source of information for warning systems. USGS uses tweets to check the accuracy of sensor reports and faster detection of earthquakes. Disasters such as Sichuan earthquake in 2008 show that Twitter is faster at reporting earthquakes than USGS. Earthquakes can be detected using tweets by 60 seconds earlier than sensors which is a valuable time for warning areas under danger and start evacuation [26]. In another effort, Sakaki et al. [81] consider each Twitter user as a sensor. The tweets by these sensors will be used to detect the occurrence of disasters and estimate their location.

5. REFLECTION OF DISASTER IMPACT ON SOCIAL MEDIA

Events are widely reflected on social media even before they are reported by news agencies and official sources. For example, in London Subway Bumbing and Virginia Tech Shooting, social media has been the primary source of information. As the event happens, social media posts show anomalies which can be captured by event detection methods. One of the major impact of disasters on social media are changes in the language of posts.

5.1 Language Change

Qualitative studies on social media show that language of users change after disasters due to distress. A study [19] on LiveJournal after 11 September 2001 shows that emotional positivity decreased and cognitive processing (intellectually understanding the issues), social orientation (how much other people are mentioned), and psychological distancing increased after the attack. Emotions of the crowd after disasters is another area which can be tracked and used by means of sentiment analysis tools. The sentiment of users

in their posts can help distinguish the posts that come from the affected area and track emotional situation of people in different stages of disasters [10].

In quantitative analysis of language change after disasters, language is statistically modeled at the level of sentences or topics. As the disaster happens, the language of the affected people on social media changes. Several measures have been developed to quantify language change. Here we introduce some of the measures that have been used in event detection methods. These methods are based on the assumption that when the language of the people in a specific area changes more than a threshold, it is a sign of an irregular event in that region (for surveys on other categories of event detection refer to [6] and [21]). Here, we enumerate some of these measures in sentence-level and topic-level language models.

5.1.1 Sentence-Level Language Change

Sentence-Level language change measures the novelty of a sentence in comparison to a presumed set of sentences. For event detection in a region using Twitter, language model of a tweet is compared to the language model of the tweets that have been posted in normal situation from the same region. Kullback-Leibler (KL) divergence [49] is a measure that calculates divergence between two sentence-level language models, Θ_1 and Θ_2 as defined in Equation 1.

$$KL(\Theta_1||\Theta_2) = \sum_w p(w|\Theta_1) log \frac{p(w|\Theta_1)}{p(w|\Theta_2)} \tag{1}$$

A sentence-level language model is a statistical model of sequences of words (i.e. sentences). As shown in Equation 2, the probability of observing a sentence, $w_1 w_2 ... w_n$, is calculated based on the assumption that probability of observing each of its words, w_i, is dependent on the previous words.

$$\begin{aligned} p(w_1 ... w_n) = p(w_1) \times p(w_2|w_1) \times \\ p(w_3|w_1 w_2) \times ... \times p(w_n|w_1 ... w_{n-1}) \end{aligned} \tag{2}$$

A common method for calculating the probabilities in Equation 2 is Maximum Likelihood estimation. In this method, the probability of observing word w_n after observing the sequence of $w_1 ... w_{n-1}$, $p(w_n|w_1 ... w_{n-1})$, is calculated using the conditional probability in Equation 3.

$$p(w_n|w_1 ... w_{n-1}) = \frac{C(w_1 ... w_n)}{\sum_W C(W w_n)} \tag{3}$$

Where $C(w_i ... w_j)$ is the frequency of observing the sequence of words w_i to w_j and W is any possible sequence of $n-1$ words in the corpus.

One challenge of using Equation 3 is calculating the denominator. Computing the denominator is computationally expensive in large corpora due to a large number of possible sentences. To overcome this challenge, n is usually set to 1, 2, or 3 which yields to unigram, bigram, or trigram language models respectively [42].

Another challenge is calculating the probability for the sequences (sentences) that have not been observed in the corpus. To overcome this problem, interpolation techniques can be used in which the probability of observing a sequence is calculated by using the probabilities of shorter sequences. For example, in a trigram language model, probability of observing a trigram can be calculated by mixing probabilities of bigrams and unigrams as shown in Equation 4.

$$\begin{aligned} \hat{p}(w_n|w_{n-2} w_{n-1}) = \lambda_1 p(w_n|w_{n-2} w_{n-1}) + \\ \lambda_2 p(w_n|w_{n-1}) + \lambda_3 p(w_n) \end{aligned} \tag{4}$$

5.1.2 Topic-Level Language Change

A set of event detection methods are based on the assumption that burst in observing explicit topics is a sign of the occurrence of an event. Explicit topics are the ones assigned by the author of posts and are mostly known as hashtags in social media. Examples of hashtags are `#frankenstorm` and `#hurricane` in the case of Hurricane Sandy (2012). The burst is considered as an unexpected rise in the frequency or a transformation of hashtags [99]. Each hashtag can be represented in time-frequency space using continuous wavelet transformation on its frequency. Wavelet peaks show unusual bursts in observing that hashtag [20].

Occurrence of events can also be detected using hidden topics. In these methods, language is modeled as a probability distribution over topics. The assumption is that there is a fixed set of hidden topics in a corpus, each document is a random mixture of these topics, and each topic is a distribution over words. Using Latent Dirichlet Allocation (LDA) [11], we can extract the probability of each topic in a document. Moreover, word distribution in each topic gives an insight on what the topic is about. To measure language change in course of a disaster, posts (such as tweets) are considered as sentences. If hidden topics of these posts largely deviate from topics of posts in regular situations, it will be considered the signal of an event.

Event detection methods based on hidden topics detect abnormal topics in a specific region. Chae et al. [15] have extracted major topics in the area of interest using LDA. A time series based on the daily message count on each topic is generated Seasonal-Trend Decomposition Procedure Based on Loess (STL) [18] has been used to decompose the time series into three components: a trend component, a seasonal component, and a remainder. The remainder is supposed to be identically distributed Gaussian white noise. However, when the remainder has a large value, it indicates substantial variation in the time series. This variation can be considered as novelty or abnormality in the language. In this work, if the seven-day moving average of the remainder values has z-score higher than 2, events can be considered abnormal in 95% confidence.

5.2 Event Detection Methods

Events are real-world occurrences that unfold over space and time and. The goal of event detection methods is extracting events in a stream of news or social media posts [4]. Event detection using social media has been extensively studied and the different categorizations are available for proposed methods in this area.

When there is no information about future events available, the event detection method falls into the unspecified category. In this category, detection methods are based on bursts or trends in the stream of posts [74]. In the specified event detection methods, contextual information such as time and venue are available for the anticipated event [8].

Another categorization of events is new versus retrospective. New event detection is extracting previously unseen events from a stream of posts as they come. Retrospective event detection also finds unseen events but the data source is an

accumulation of historic posts [4]. Clustering methods are the most common in detecting both types of events [6]. But there are also supervised methods such as Naive Bayes [9] and gradient boosted decision trees that have been used for new event detection [75].

Clustering methods focus on documents and grouping them based on similarities, i.e. they are document-pivot. There is a group of feature-pivot techniques that use changes and bursts in features of documents to detect events. These features include frequency of specific keywords [77], surprise level of relevant keywords [83], and statistical features of posts (i.e. word frequencies) [81].

6. FACILITATING RESPONSE VIA SOCIAL MEDIA

In the chaotic environment of disasters, emergency responders want to acquire a big picture of the event and actionable insight [14]. Preliminary assessment of the disaster such as the area which is affected, the number of casualties, and failed infrastructures are obtained in the "big picture". "Actionable insights" are concerned with specific information with more details such as requests for help.

6.1 Tracking Disasters

Systems that monitor social media for crisis-related purposes use computational capabilities such as collecting data, Natural Language Processing, information extraction, monitoring changes in data statistics, clustering similar messages, and automatic translation [37]. Three important results of these computations are topics, trends, and memes. In the remaining of this section, we discuss how to gain and use these three for tracking the status of disasters.

6.1.1 Topic Discovery and Evolution

Several methods are used for discovering topics from a corpus of text (news articles, tweets, Facebook posts, etc): Latent Dirichlet Allocation (LDA), Non-negative Matrix Factorization (NMF), Term frequency-inverse document frequency (tf-idf), and PLSI. LDA [11] considers a fixed number of latent topics for the corpus and finds the probability of each document belonging to each topic. A topic itself is a distribution over words (vocabulary). In NMF [52], the document-word matrix would be factorized into two matrices, document-topic and topic-word. NMF describes both documents and terms in the environment of latent topics. tf-idf is widely used in Information Retrieval. Using tf-idf, each document is represented using a vector of size V (vocabulary size). Element ij of this the tf-idf vector is the frequency of word i in document j time the inverse of the total frequency of word i in the whole corpus of documents. Probabilistic latent semantic indexing (PLSI) is based on the assumption that each document has one topic and words and documents are conditionally independent given the topic of the document. Hence the probability of a word w_n in document d is calculated using $p(w_n, d) = p(d) \sum_z p(w_n|z)p(z|d)$.

As the disaster proceeds in its stages, concerns of the affected people evolve and hence the topics of discussion. There is a body of research modeling evolving and fading topics which are usually known as topic discovery and evolution (TDE) methods. Vaca et al. [93] and Kalyanam et al. [45] in separate works, similarly model this problem using a modified version of NMF. In these proposed methods, besides decomposing the document-word matrix into document-topic and topic-word matrices, a matrix M would be considered which models how topics evolve from each time step to the next. Hence one topic matrix is calculated at each point of time. The entropy of topics in each time step indicates its status, continuing, evolving, or new. In another work [44], besides using the evolution of topics, they have also exploited social context. Their method is based on the assumption that members of one community have similar interest in the topics and show that this information improves topic discovery results especially when the topic has a large set of keywords associated with it or evolves much over time and hence is difficult to detect. However, the persistence of the user who shows interest in these topics help the detection methods.

6.1.2 Trend Mining

Mathioudakis et al. [59] define trends as "set of bursty keywords that occur frequently together" which are driven by events and breaking news. Naaman et al. [67] define a score for each word on Twitter and the top 30 words will be considered as trending in each hour. The score is calculated using *((word frequency in a specific hour)-(average frequency of the word in this specific hour across weeks))/(standard deviation of the frequency of this word in this hour across weeks)*. Trends on Twitter are either caused by an external happening (such as a natural disaster) or are specific to the social media (a tweet by a famous user). Based on the source, they are divided into exogenous and endogenous [67]. Trends indicate what are the major subjects that people talk about. Resources that the affected people need and the issued that they talk about can be a subset of trends after a disaster. TwitterMonitor [59] is a system that detect new trends/topics by finding bursty keywords and clustering them based on co-occurrence. In the analysis of trends, they summarize each trend using the most frequent keywords, other words which are not as dominant but they are highly correlated to the frequent keywords, and also the named entities and sources (URLs) which are mentioned frequently. Cui et al. [23] propose a clustering-based model for visualizing topic discovery and evolution. They use a heuristic to hash the topics in each time step and for discovering new topics and detecting the death of topics (these two are critical topics) they compare the hashes from one time step to the next. They also monitor the merge and split of topics and these changes in the topics are shown in a river flow visualization. Width of a flow shows the occurrence number of all involved keywords in that topics and the merge, split, birth, and death of topics are shown by colors.

Another approach is a fine-grain classification of trends that is based on comparing the frequency of words/hashtags before and after a spike. Lehman et al. [54] classify the trends on Twitter into three categories based on the shape of the spike: activity centered before and during the peak, concentrated during and after the peak, and symmetric activity.

6.1.3 Meme Tracking

Memes are short units of text that act as the signature of a topic [55]. Harve et al. [33] introduce a visualization for thematic change in over time for a corpus of documents. Their method shows a "river" of themes that includes colored "currents". Each current is a theme and its width at a specific point of time shows its strength. When a current

becomes wider, the topic or set of topics associated with it are more dominant in the corpus and as the color changes, the themes in the corpus are is changing.

Leskovec et al. [55] detect and track memes (quotes) in news articles. They generate a graph in which each node is a meme and there is an edge from node i to j if the meme in node i is strictly shorter that j and the directed edit distance to j is less than one. This directed acyclic graph is then partitioned in a way that all the nodes in one cluster can be considered "belonging" either to a single long phrase or to a single collection of phrases. To analyze the evolution of memes in a corpus of news articles, they use variations of a meme and extract all the articles that contain those memes and graphical visualize the changes in the volume of corresponding documents over time using stacked plots.

The same concept of memes has been used to monitor bias in news outlet by different parties in the United States. Niculae et al. [68] build a bipartite graph from quotes of the president's speeches to news outlets which are used in a matrix factorization method to predict the future quotes of a news outlet based on its previous quotes. Moreover, they analyze the sentiment and negativity of quotes in different outlets and present the bias present in reporting parts of the speeches with a specific choice of negative words and negative sentiment in one of the parties.

6.2 Situational Awareness

Informative posts provide "tactical, actionable information that can aid people in making decisions, advise others on how to obtain specific information from various sources, or offer immediate post-impact help to those affected by the mass emergency" [95]. Informative posts can be categorized into six groups [39]: caution or advice, information source (photos and videos), people (missing or found), casualties and damage (infrastructures, injured, or dead), donations (requests and offers for money/good/services), people (celebrities and authorities).

One attempt toward increasing situational awareness based on social media posts is extracting requests and responses. Varga et al. [94] use content of tweets to extract problems and aid reports on Twitter after Japan earthquake (2010). They use the notion of "problem nucleus" and "aid nucleus" and exploit features such as trouble expressions (a manually created list of trouble expressions), excitation polarity (excitatory, inhibitory, and neutral), word sentiment polarities and word semantic word class (clusters of words such as food and disease), and location mentions to train a classifier that labels tweets as a problem or air report.

Another attempt is toward matching requests with appropriate responses. Purohit et al. [78] propose a solution using a two-step model. In the first step, they use a binary classifier that uses n-grams and regular expressions to label tweets as requests and offers. In the next step, based on the cosine similarity of tf-idf term vectors, they match requests and offers. In this work they focus on donation related tweets and they consider money, medical, volunteer, clothing, food, and shelter requests/offers as subcategories. Their studies on Hurricane Sandy dataset (2012) shows that the majority of donations lay in "money" category.

7. RELIEF ASSISTANCE VIA SOCIAL MEDIA

Volunteers are significantly important in the relief process. They post information such as road status and damages to built structures which increases awareness. They also provide technical support such as translation of posts and geotagging them. There are also several systems that are built to exploit and organize such efforts.

7.1 Crowdsourcing

Volunteering is part of how community reacts to disasters [25] and this process has been facilitated by social media in recent years. Volunteers provide information and resources to the affected people. Example of such efforts are providing temporary housing for stranded people in the US after terrorist attacks in Paris (2015) [86] and offering food and shelter after Hurricane Sandy (2012) [46].

Digital technologies have broaden domain of volunteer activities in course of disasters. "Digital volunteers" [14], either located in the disaster area or in distant locations, ease the relief efforts by providing variate of services. Translation of posts, geotagging posts that indicate an incident on the map, creating maps of open and blocked roads, increasing accuracy of maps by marking built entities on the maps are example of such efforts [61].

Many systems have been developed to benefit and organize volunteers who use social media. OpenStreetMap is one of the systems that allows volunteers contribute in generating open source maps by marking entities such as roads and buildings. These maps have been used for disasters such as Haiti earthquake (2010) [104]. The details about more systems is presented in Section 7.2.

7.2 Relief Tools

Social media is a unique platform for collaboration between remote volunteers. These volunteers provide technical services such as translation, geolocating posts on the map, and generating maps of the affected area. Several tools have been developed to use crowdsourcing and social media for facilitating volunteering actions.

7.3 Ushahidi

Ushahidi is the first large-scale crowdsourcing system for disaster relief. It has been initially developed to map the reports of Kenyan post-election violence in 2008 and since then has been used in many major disasters such as Hurricane Sandy and Haiti Earthquake. Ushahidi is an open source and free systems which can either be deployed on external servers or on Ushahidi's hosting system CrowdMap. When technical knowledge or hosting servers are not available, CrowdMap is a more suitable.

Ushahidi has three main sections: data collection, visualization, and filtering. As the first step, disaster-related data is collected from several sources, web, Twitter, RSS feeds, emails, SMS, and manual comma separated files. The user-contributed information is then visualized on the map. Each point on the map shows one report and when a user zooms out, aggregated number of reports in each area is represented. As the last step, Ushahidi allows users to filter reports based on their types, e.g. supplies or shelter.

7.4 AIDR

Artificial Intelligence for Disaster Response (AIDR) is a free software platform which can be either run as a web application or created as its own instance. This system allows

the detection of different categories of tweets based on a small sample of labeled tweets. The process has three steps, data collection, annotation, and classification. Tweets are collected based on a pre-selected set of keywords. A small portion of theses tweets is then labeled by volunteers as in-category or out-category. In each disaster, different categories can be considered such as status update, shelter, or food. Labeled tweets which can be as few as 200, will be used as the training set of a classifier which labels remaining set of tweets which were collected based on the keywords. In the training process, n-grams of tweets are used as features and hence the classifier needs to be retrained for every new category and disaster.

7.5 TweetTracker

TweetTracker is a system for tracking, analyzing, and understanding tweets related to a specific topic. To track the status of and event, data can be collected using a set of criteria including keywords, location, and users. The source of the data can be chosen from Twitter, Facebook, YouTube, VK, and Instagram. Changes in the total number of post or frequency of posts with specific words can be plotted for different time periods. Moreover, keywords, hashtags, links, images, and videos with their frequencies are available to the user. To better understand the geographic distribution of posts on the globe, the posts which are geotagged will be shown on a map.

All the features mentioned above are useful for any topic which is discussed on social media. However, there is a module in TweetTracker which is specifically designed for disaster relief. In this module, as the tweets related to the target disaster are captured by the system, the ones which are most probable to contain a request for help will be detected. These tweets in the majority are posted by the affected people and need urgent attention. The classifier for this task works based on both content (n-grams) and metadata of tweets. This brings the flexibility which lets the classifier be used for different disasters. The more certain requests that have geolocation will be also shown on the map. A view of this system is shown in Fig. 3.

8. CONCLUSION

Disasters are widely reflected on social media and this swarm of information provides valuable insight for governments, NGOs, emergency managers, and first responders. It also helps the affected people keep in touch with their loved ones, finds information about the status of the disaster, and be informed about emergency contacts. Social media is the new way of communication in emergencies which transfers information before and faster than traditional news media. It is prevalent in such a way that disaster responders encourage citizens to exploit it to take some load off the cellular systems which usually becomes overwhelmed by calls and text messages in chaotic situations. On social media, both responders and affected people can broadcast information and in contrast with traditional media, people can also provide feedback to officials.

These potentials have encouraged responders to benefit social media in large extent in recent years. However, there are challenges associated with this task. Social media posts come at a fast pace and immense volume. Moreover, it is challenging to collect all the posts which are related to a disaster due to the restrictions posed by social media websites. The collected data contains daily chatters, prayers, and opinions and is only in part insightful information which adds to the situational awareness. Another issue is malicious content such as spam and rumors which can cause panic and stress, especially when produced in large scale using bots. Even after the data is filtered from all the aforementioned posts, it is still challenging to extract specific groups of information such as requests for food or shelter.

In this paper, we focused on four phases of disasters: warning, impact, response, and relief. These stages are the ones during which computer science has been helpful the most. For each stage, we introduced some of the recent impactful research and response has the greatest portion because most of the social media activity after disasters are in this phase. Social media posts can be used to detect the onset of events even prior to official sources and be used in warning systems. Several methods have been used to increase the coverage of data which is collected regarding disasters and filtering it from unwanted content. We also mentioned tools, such as Ushahidi, AIDR, and TweetTracker, for exploiting volunteer efforts.

9. LOOKING AHEAD

Although disaster management has achieved major advancements in using social media, there are still several challenges to overcome:

Warning systems use anomalies in the data to predict an event. This process requires constant data collection and comparison of trends over time. Handling this volume of data is expensive, extracting topics is elaborate, and maintaining trends for future comparison is expensive.

Malicious users and most importantly bots roar over social media and affect the discussion when organized in large groups. It is even difficult to for humans to distinguish complex bots from humans. Misleading content such as rumors is also harmful in aftermath of a crisis and finding the source of a rumor, the intention of spreading it, and intervention, before it goes viral, is laborious.

Another area for potential improvement is ground truth acquisition for machine learning methods that automate extraction of specific posts. There have been efforts toward crowdsourcing such tasks but it is still challenging. Each disaster, location, and time has its own specificity and no global method could have been proposed which can be trained in one situation and be used in others.

The last point is the integration of data and methods from different fields. Seismologists collect abundant amount of data from sensors and have meticulous methods for detecting earthquakes. On the other hand, the enormous amount of data is published on social media, moments after the earthquake. Integration of social media data with other sources could increase the accuracy of the information to be collected/disseminated. There are some efforts in this direction [31; 66] but there is room for improvement.

10. ACKNOWLEDGMENTS

This research was supported, in part, by NSF grant 1461886, ONR grant N00014-16-1-2257,and JST Strategic International Collaborative Research Program (SICORP). The information reported here does not reflect the position or the

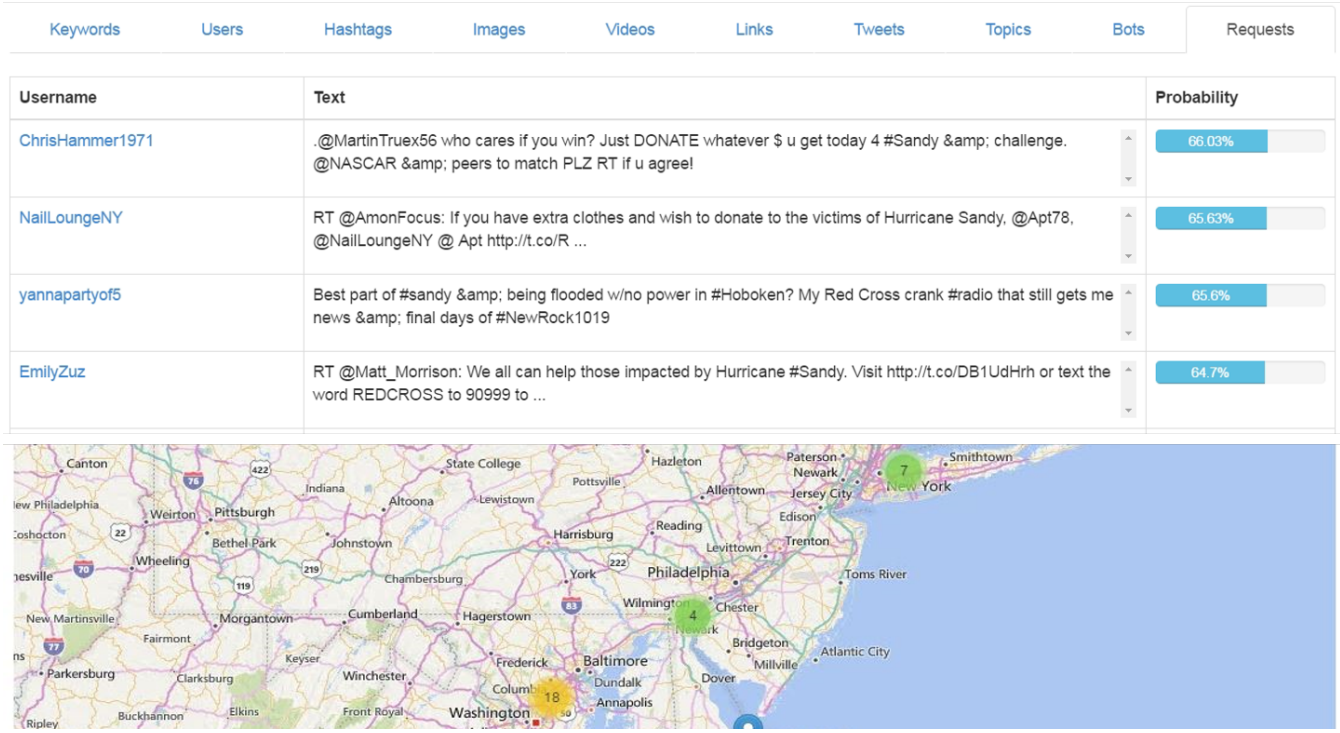

Figure 3: A view of a TweetTracker module which shows requests-for-help tweets related to Hurricane Sandy.

policy of the funding agencies. The authors would like to thank members of Data Mining and Machine Learning (DMML) Lab at Arizona State University and Lei Zhong from National Institute of Informatics, Japan for their feedback and contributions.

11. REFERENCES

[1] Disasters and emergencies: Definitions. http://apps.who.int/disasters/repo/7656.pdf, 2002. accessed 09 Jan 2017.

[2] Twitter responds to the japanese disaster. https://goo.gl/8V1WC7, Mar. 17, 2011. accessed 13 Feb 2017.

[3] Hurricane matthew recap: Destruction from the caribbean to the united states. https://goo.gl/8Rfn6S, Oct. 9, 2016. accessed 10 Oct 2016.

[4] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998.

[5] S. Asur and B. A. Huberman. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE, 2010.

[6] F. Atefeh and W. Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.

[7] L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th international conference on World wide web*, pages 61–70. ACM, 2010.

[8] H. Becker, F. Chen, D. Iter, M. Naaman, and L. Gravano. Automatic identification and presentation of twitter content for planned events. In *ICWSM*, pages 655–656, 2011.

[9] H. Becker, M. Naaman, and L. Gravano. Beyond trending topics: Real-world event identification on twitter. *ICWSM*, 11:438–441, 2011.

[10] G. Beigi, X. Hu, R. Maciejewski, and H. Liu. An overview of sentiment analysis in social media and its applications in disaster relief. In *Sentiment Analysis and Ontology Engineering*, pages 313–340. Springer, 2016.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[12] B. J. Boruff. Environmental hazards: Assessing risk and reducing disasters, 5th edition by keith smith and david n. petley. *Geographical Research*, 47(4):454–455, 2009.

[13] A. Bruns and Y. E. Liang. Tools and methods for capturing twitter data during natural disasters. *First Monday*, 17(4), 2012.

[14] C. Castillo. *Big Crisis Data*. Cambridge University Press, 2016.

[15] J. Chae, D. Thom, Y. Jang, S. Kim, T. Ertl, and D. S. Ebert. Public behavior response analysis in disaster events utilizing visual analytics of microblog data. *Computers & Graphics*, 38:51–60, 2014.

[16] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 759–768. ACM, 2010.

[17] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *Proceedings of the 26th annual computer security applications conference*, pages 21–30. ACM, 2010.

[18] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–33, 1990.

[19] M. A. Cohn, M. R. Mehl, and J. W. Pennebaker. Linguistic markers of psychological change surrounding september 11, 2001. *Psychological science*, 15(10):687–693, 2004.

[20] M. Cordeiro. Twitter event detection: combining wavelet analysis and topic inference summarization. In *Doctoral Symposium on Informatics Engineering*, 2012.

[21] M. Cordeiro and J. Gama. Online social networks event detection: a survey. In *Solving Large Scale Learning Tasks. Challenges and Algorithms*, pages 1–41. Springer, 2016.

[22] A. Crowe. *Disasters 2.0: The application of social media systems for modern emergency management*. CRC press, 2012.

[23] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. Textflow: Towards better understanding of evolving topics in text. *IEEE transactions on visualization and computer graphics*, 17(12):2412–2421, 2011.

[24] N. DiFonzo and P. Bordia. *Rumor psychology: Social and organizational approaches*. American Psychological Association, 2007.

[25] R. R. Dynes. *Organized behavior in disaster*. Heath LexingtonBooks, 1970.

[26] E. Ellis. How the usgs uses twitter data to track earthquakes. https://goo.gl/E6r0b2, Oct. 7, 2015. accessed 28 Nov 2016.

[27] M. Faulkner, R. Clayton, T. Heaton, K. M. Chandy, M. Kohler, J. Bunn, R. Guy, A. Liu, M. Olson, M. Cheng, et al. Community sense and response systems: Your phone as quake detector. *Communications of the ACM*, 57(7):66–75, 2014.

[28] M. Faulkner, M. Olson, R. Chandy, J. Krause, K. M. Chandy, and A. Krause. The next big one: Detecting earthquakes and other rare events from community-based sensors. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 13–24. IEEE, 2011.

[29] A. Gupta, H. Lamba, and P. Kumaraguru. $1.00 per rt #bostonmarathon #prayforboston: Analyzing fake content on twitter. In *eCrime Researchers Summit (eCRS)*, pages 1–12. IEEE, 2013.

[30] A. Gupta, H. Lamba, P. Kumaraguru, and A. Joshi. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*, pages 729–736. ACM, 2013.

[31] M. Guy, P. Earle, C. Ostrum, K. Gruchalla, and S. Horvath. Integration and dissemination of citizen reported and seismically derived earthquake information via social network technologies. In *International Symposium on Intelligent Data Analysis*, pages 42–53. Springer, 2010.

[32] B. Han, P. Cook, and T. Baldwin. A stacking-based approach to twitter user geolocation prediction. In *ACL (Conference System Demonstrations)*, pages 7–12, 2013.

[33] S. Havre, B. Hetzler, and L. Nowell. Themeriver: Visualizing theme changes over time. In *Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, pages 115–123. IEEE, 2000.

[34] P. Heymann, G. Koutrika, and H. Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11(6):36–45, 2007.

[35] R. Hill and D. Hansen. Families in disaster. In *Man and Society in Disaster*, pages 185–221. Basic Books, 1962.

[36] J. B. Houston, J. Hawthorne, M. F. Perreault, E. H. Park, M. Goldstein Hode, M. R. Halliwell, S. E. Turner McGowen, R. Davis, S. Vaid, J. A. McElderry, et al. Social media and disasters: a functional framework for social media use in disaster planning, response, and research. *Disasters*, 39(1):1–22, 2015.

[37] M. Imran, C. Castillo, F. Diaz, and S. Vieweg. Processing social media messages in mass emergency: A survey. *ACM Computing Surveys (CSUR)*, 47(4):67:1–67:38, 2015.

[38] M. Imran, C. Castillo, J. Lucas, P. Meier, and S. Vieweg. Aidr: Artificial intelligence for disaster response. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 159–162. ACM, 2014.

[39] M. Imran, S. Elbassuoni, C. Castillo, F. Diaz, and P. Meier. Practical extraction of disaster-relevant information from social media. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1021–1024. ACM, 2013.

[40] M. Imran, S. M. Elbassuoni, C. Castillo, F. Diaz, and P. Meier. Extracting information nuggets from disaster-related messages in social media. *Proc. of ISCRAM, Baden-Baden, Germany*, 2013.

[41] J. P. John, A. Moshchuk, S. D. Gribble, A. Krishnamurthy, et al. Studying spamming botnets using botlab. In *NSDI*, volume 9, pages 291–306, 2009.

[42] D. Jurafsky and J. H. Martin. *Speech and language processing*. Pearson, 2014.

[43] D. Jurgens. That's what friends are for: Inferring location in online social media platforms based on social relationships. *ICWSM*, 13:273–282, 2013.

[44] J. Kalyanam, A. Mantrach, D. Saez-Trumper, H. Vahabi, and G. Lanckriet. Leveraging social context for modeling topic evolution. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 517–526. ACM, 2015.

[45] S. V. M. C. Kalyanam, Janani and G. Lanckriet. From event detection to story telling on microblogs. In *Proceedings of the ACM/IEEE Conference on Advances in Social Network Analysis and Mining (ASONAM)*, pages 437–442. ACM, 2016.

[46] L. Kavner. Hurricane sandy: Red cross, other relief organizations see social media as double-edged sword for relief efforts. https://goo.gl/angXF8, Oct. 31, 2012. accessed 09 Jan 2017.

[47] Y. Koh. Only 11% of new twitter users in 2012 are still tweeting. https://goo.gl/vl9D3h, Mar. 21, 2014. accessed 13 Feb 2017.

[48] G. Kossinets and D. J. Watts. Empirical analysis of an evolving social network. *science*, 311(5757):88–90, 2006.

[49] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[50] S. Kumar, G. Barbier, M. A. Abbasi, and H. Liu. Tweettracker: An analysis tool for humanitarian and disaster relief. In *ICWSM*, pages 661–662, 2011.

[51] S. Kumar, X. Hu, and H. Liu. A behavior analytics approach to identifying tweets from crisis regions. In *Proceedings of the 25th ACM conference on Hypertext and social media*, pages 255–260. ACM, 2014.

[52] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[53] K. Lee, B. D. Eoff, and J. Caverlee. Seven months with the devils: A long-term study of content polluters on twitter. In *ICWSM*, pages 185–192, 2011.

[54] J. Lehmann, B. Gonçalves, J. J. Ramasco, and C. Cattuto. Dynamical classes of collective attention in twitter. In *Proceedings of the 21st international conference on World Wide Web*, pages 251–260. ACM, 2012.

[55] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[56] R. Li, S. Wang, H. Deng, R. Wang, and K. C.-C. Chang. Towards social user profiling: unified and discriminative influence model for inferring home locations. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1023–1031. ACM, 2012.

[57] V. Luckerson. Fear, misinformation, and social media complicate ebola fight. *Time Magazine*, 2014.

[58] J. Mahmud, J. Nichols, and C. Drews. Where is this tweet from? inferring home locations of twitter users. *ICWSM*, 12:511–514, 2012.

[59] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM, 2010.

[60] P. Meier. *Digital humanitarians: how big data is changing the face of humanitarian response*. Crc Press, 2015.

[61] P. Meier. How crisis mapping saved lives in haiti. https://goo.gl/uASbu8, Jul. 2, 2012. accessed 09 Jan 2017.

[62] M. Mendoza, B. Poblete, and C. Castillo. Twitter under crisis: can we trust what we rt? In *Proceedings of the first workshop on social media analytics*, pages 71–79. ACM, 2010.

[63] G. Mishne, N. S. Glance, et al. Predicting movie sales from blogger sentiment. In *AAAI spring symposium: computational approaches to analyzing weblogs*, pages 155–158, 2006.

[64] F. Morstatter, N. Lubold, H. Pon-Barry, J. Pfeffer, and H. Liu. Finding eyewitness tweets during crises. *arXiv preprint arXiv:1403.1773*, 2014.

[65] F. Morstatter, L. Wu, T. H. Nazer, K. M. Carley, and H. Liu. A new approach to bot detection: striking the balance between precision and recall. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 533–540, 2016.

[66] A. Musaev, D. Wang, and C. Pu. Litmus: Landslide detection by integrating multiple sources. In *11th International Conference Information Systems for Crisis Response and Management (ISCRAM)*, pages 677–686, 2014.

[67] M. Naaman, H. Becker, and L. Gravano. Hip and trendy: Characterizing emerging trends on twitter. *Journal of the American Society for Information Science and Technology*, 62(5):902–918, 2011.

[68] V. Niculae, C. Suen, J. Zhang, C. Danescu-Niculescu-Mizil, and J. Leskovec. Quotus: The structure of political media coverage as revealed by quoting patterns. In *Proceedings of the 24th International Conference on World Wide Web*, pages 798–808. ACM, 2015.

[69] O. Okolloh. Ushahidi, or testimony: Web 2.0 tools for crowdsourcing crisis information. *Participatory learning and action*, 59(1):65–70, 2009.

[70] A. Olteanu, C. Castillo, F. Diaz, and S. Vieweg. Crisislex: A lexicon for collecting and filtering microblogged communications in crises. In *ICWSM*, pages 376–385, 2014.

[71] L. Palen and K. M. Anderson. Crisis informaticsnew data for extraordinary times. *Science*, 353(6296):224–225, 2016.

[72] L. Palen and K. M. Anderson. Crisis informatics—new data for extraordinary times. *Science*, 353(6296):224–225, 2016.

[73] B. Palm. Hurricane matthew reaches category 4 status, barreling toward florida. https://goo.gl/ZW33U3, Oct. 6, 2016. accessed 10 Oct 2016.

[74] A.-M. Popescu and M. Pennacchiotti. Detecting controversial events from twitter. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1873–1876. ACM, 2010.

[75] A.-M. Popescu, M. Pennacchiotti, and D. Paranjpe. Extracting events and event descriptions from twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 105–106. ACM, 2011.

[76] J. W. Powell. An introduction to the natural history of disaster. *Univ. of Maryland: Disaster Research Project*, 1954.

[77] R. Power, B. Robinson, J. Colton, and M. Cameron. Emergency situation awareness: Twitter case studies. In *International Conference on Information Systems for Crisis Response and Management in Mediterranean Countries*, pages 218–231. Springer, 2014.

[78] H. Purohit, C. Castillo, F. Diaz, A. Sheth, and P. Meier. Emergency-relief coordination on social media: Automatically matching resource requests and offers. *First Monday*, 19(1), 2013.

[79] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, S. Patil, A. Flammini, and F. Menczer. Truthy: mapping the spread of astroturf in microblog streams. In *Proceedings of the 20th international conference companion on World wide web*, pages 249–252. ACM, 2011.

[80] A. Reese. How we'll predict the next natural disaster: Advances in natural hazard forecasting could help keep more people out of harm's way. *Discover Magazine*, Sep. 2016.

[81] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[82] N. SAMBULI. How useful is a tweet? a review of the first tweets from the westgate mall attack. https://goo.gl/qRGYZD, Oct. 3, 2013. accessed 10 Feb 2017.

[83] J. Sampson, F. Morstatter, R. Zafarani, and H. Liu. Real-time crisis mapping using language distribution. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1648–1651. IEEE, 2015.

[84] D. Schramm and R. Hansen. Aim & scope of disaster management: Study guide and course text. *Disaster Management Center, University of Wisconsin-Madison*, 1986.

[85] A. Schulz, A. Hadjakos, H. Paulheim, J. Nachtwey, and M. Mühlhäuser. A multi-indicator approach for geolocalization of tweets. In *ICWSM*, pages 573–582, 2013.

[86] J. SEDERHOLM. #strandedinus: Americans open homes to strangers stuck after paris attacks. https://goo.gl/NQqEDh, Nov. 14, 2015. accessed 09 Jan 2017.

[87] J. H. Sorensen. Hazard warning systems: Review of 20 years of progress. *Natural Hazards Review*, 1(2):119–125, 2000.

[88] K. Starbird and J. Stamberger. Tweak the tweet: Leveraging microblogging proliferation with a prescriptive syntax to support citizen reporting. In *Proceedings of the 7th International ISCRAM Conference–Seattle*, pages 1–5, 2010.

[89] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.

[90] I. Temnikova, C. Castillo, and S. Vieweg. Emterms 1. 0: a terminological resource for crisis tweets. In *ISCRAM 2015 proceedings of the 12th international conference on information systems for crisis response and management*, 2015.

[91] K. Thomas, C. Grier, and V. Paxson. Adapting social spam infrastructure for political censorship. In *LEET*, 2012.

[92] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM*, 10:178–185, 2010.

[93] C. K. Vaca, A. Mantrach, A. Jaimes, and M. Saerens. A time-based collective factorization for topic discovery and monitoring in news. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 527–538, New York, NY, USA, 2014. ACM.

[94] I. Varga, M. Sano, K. Torisawa, C. Hashimoto, K. Ohtake, T. Kawai, J.-H. Oh, and S. De Saeger. Aid is out there: Looking for help from tweets during a large scale disaster. In *ACL (1)*, pages 1619–1629, 2013.

[95] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1079–1088. ACM, 2010.

[96] X. Wang, M. S. Gerber, and D. E. Brown. Automatic crime prediction using events extracted from twitter posts. In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 231–238. Springer, 2012.

[97] D. H. Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[98] F. Wu, J. Shu, Y. Huang, and Z. Yuan. Social spammer and spam message co-detection in microblogging with social context regularization. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1601–1610. ACM, 2015.

[99] W. Xie, F. Zhu, J. Jiang, E.-P. Lim, and K. Wang. Topicsketch: Real-time bursty topic detection from twitter. In *2013 IEEE 13th International Conference on Data Mining*, pages 837–846. IEEE, 2013.

[100] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: signatures and characteristics. *ACM SIGCOMM Computer Communication Review*, 38(4):171–182, 2008.

[101] S. Yu and S. Kak. A survey of prediction using social media. *arXiv preprint arXiv:1203.1647*, 2012.

[102] R. Zafarani and H. Liu. 10 bits of surprise: Detecting malicious users with minimum information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 423–431. ACM, 2015.

[103] W. Zhang and S. Skiena. Improving movie gross prediction through news analysis. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 01*, pages 301–304. IEEE Computer Society, 2009.

[104] M. Zook, M. Graham, T. Shelton, and S. Gorman. Volunteered geographic information and crowdsourcing disaster relief: a case study of the haitian earthquake. *World Medical and Health Policy by Wiley Online Library*, 2(2):7–33, 2010.