Fair Link Prediction with Multi-Armed Bandit Algorithms

Weixiang Wang
wwang69@syr.edu
Syracuse University
Syracuse, New York, USA

Sucheta Soundarajan
susounda@syr.edu
Syracuse University
Syracuse, New York, USA

ABSTRACT

Recommendation systems have been used in many domains, and in recent years, ethical problems associated with such systems have gained serious attention. The problem of unfairness in friendship or link recommendation systems in social networks has begun attracting attention, as such unfairness can cause problems like segmentation and echo chambers. One challenge in this problem is that there are many fairness metrics for networks, and existing methods only consider the improvement of a single specific fairness indicator [16, 17, 20].

In this work, we model the fair link prediction problem as a multi-armed bandit problem. We propose `FairLink`, a multi-armed bandit based framework that predicts new edges that are both accurate and well-behaved with respect to a fairness property of choice. This method allows the user to specify the desired fairness metric. Experiments on five real-world datasets show that `FairLink` can achieve a significant fairness improvement as compared to a standard recommendation algorithm, with only a small reduction in accuracy.

KEYWORDS

social networks, fairness, link prediction

ACM Reference Format:
Weixiang Wang and Sucheta Soundarajan. 2023. Fair Link Prediction with Multi-Armed Bandit Algorithms. In *15th ACM Web Science Conference 2023 (WebSci '23), April 30–May 01, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3578503.3583624

1 INTRODUCTION

Link prediction/friendship recommendation is a fundamental topic in the field of complex network analysis, and has attracted a great deal of attention over the last two decades [19]. Algorithms for this problem are used in a number of important applications, most visibly in online social networking platforms, such as Facebook, LinkedIn, or Twitter. However, although these networks exist in the online ecosystem, they nonetheless have the potential to affect a person's offline life: for example, professional online social networks can play an important role in broadening users' career prospects and providing access to promotion [12].

WebSci '23, April 30–May 01, 2023, Austin, TX, USA

ACM ISBN 979-8-4007-0089-7/23/04…$15.00
https://doi.org/10.1145/3578503.3583624

Because of these offline consequences, the effect of network algorithms on fairness-related network properties has gained serious attention. Network fairness is usually defined with respect to protected groups of interest (often based on race, gender, or other protected attributes), and can be measured in different ways. One common fairness property is *homophily*, which measures segregation between groups. Homophily describes a phenomenon in sociology in which individuals are more likely to connect with others who have similar characteristics, such as habits, beliefs, or values [21]. The recently-proposed *information unfairness* metric quantifies fairness from the information diffusion perspective, evaluating whether information is spreading fairly between all groups in a network [14]. Such types of 'unfairness' are not always a bad thing– for instance, individuals tend to follow online social accounts who are similar to them and are thus of greater interest, or minorities in a network may organize into groups to better advocate for themselves and defend their rights and interests. However, depending on the context, unfairness in network structure can be linked to serious negative consequences, such as segregation [21] and echo chambers [11].

Other common fairness metrics do not evaluate network structure directly, but examines the distribution of predicted edges: In the *statistical parity* metric, the probability that an edge (u, v) is predicted should be the same regardless of whether nodes u and v are in the same group or in different groups [16], and in the *accuracy disparity* metric, fairness is the link prediction accuracy difference between inter-group and intra-group edges [17].

Previous works have observed that link prediction algorithms can sometimes worsen fairness-related properties, including homophily [3], information unfairness [14] and accuracy disparity [16, 17]. This occurs because most link prediction methods predict links between vertices that are similar (in terms of topological properties or attributes). When dealing with sensitive features like race or gender, this can be highly problematic. If link prediction algorithms intensify segregation or information unfairness, already-sparse communications between two communities may lessen further. Communities will thus tend to polarize, and opinions become more extreme [31].

The literature contains only a few existing works on fair link prediction. Masrour *et al.* propose a GAN-based method for predicting links in a network without increasing homophily [20]. Laclau *et al.* propose an embedding-agnostic repairing procedure for the adjacency matrix of an arbitrary graph with a trade-off between the group and individual fairness [16], while Li *et al.* mitigate the accuracy disparity through a pre-processing that flattens the density of edge groups [17]. However, as discussed above, there are multiple ways in which one can measure the 'fairness' of a network structure, and so flexibility with respect to fairness properties is

important. Moreover, while these works are effective, they are restricted to homophily or other dyadic properties that look only at node class memberships, not higher-order structural properties of the network.

In this work, we propose FairLink, a link prediction framework with the simultaneous goal of predicting links that are both *accurate* and *mitigate unfairness* in the network structure, where unfairness is measured with respect to a user-specified definition. Because fairness can mean different things in different contexts, we allow the user to *specify the fairness metric* and weightage between accuracy and fairness.

In this work, we approach the fair link prediction problem as a multi-armed bandit problem, in which the goal is to optimize the total fairness benefit by consecutively recommending edges to users that simultaneously have high accuracy and fairness properties. This technique can easily be generalized to other fairness properties.

The major contributions of this work include:

(1) We propose FairLink, a multi-armed bandit framework for fair link prediction, in which the user specifies the fairness metric as well as the desired balance between fairness and accuracy.
(2) We perform an extensive experimental analysis on 5 datasets, and compare to both standard and state-of-the-art 'fair' link prediction algorithms.
(3) We demonstrate that FairLink achieves up to 110% fairness improvements as compared to a standard link prediction algorithms, with only a small reduction in accuracy.

2 RELATED WORK

2.1 Link prediction in Social Network

In the link prediction problem, one assumes that there is an observed network that is a subset of a larger, unobserved network. The goal is to predict which edges that do not exist in the observed network are most likely to exist in the unobserved network.

Link prediction algorithms take advantage of many network properties, including local topology features (e.g., common neighbors), path-based network structural features (e.g. shortest distance), and vertex attributes (e.g., beliefs) to make predictions. Many traditional link prediction algorithms score each candidate edge according to the corresponding measurement criteria and select the candidate edges from high to low. A variety of link prediction algorithms exist, using Jaccard similarity [8], random walks [18], common neighbors [18], and others. In addition, there are a number of machine learning methods, which combine features into a predictive model. Such methods can be divided into feature classification methods, probability graph model methods, and matrix decomposition methods [4]. These methods treat link prediction as a two-class classification problem, with the goal of distinguishing between existent edges (+1) and non-existent edges (-1), and assign candidate edges a probability (score) of being in the former class, and as before, the highest-scoring candidate edges are selected. Many of these methods use structural features, as well as node attributes [26]. For instance, centrality, common neighbors, community information and path-related measures are used as features in [9].

2.2 Link Prediction and Fairness

In the last few years, there has been a great deal of interest in the notion of fairness in machine learning [13]. Closely related to our work is the literature on bias in recommendations. For example, Burke *et al.* consider bias in personalized recommendation systems [6], Chaney *et al.* show how recommendations can lead to homogenization [7], and Adomavicius and Kwon discuss diversity in recommendations [2].

In the context of recommendation in social networks, Daly *et al.* in [10] explore the effects of different prediction algorithms. A rich-get-richer phenomenon is observed as the result of follow recommendation in [25] by Su *et al.* Structural diversity in taken into account for evaluating link recommendation algorithms by Sanz-Cruzado *et al.*, who show that recommending weak ties can improve structural diversity [23]. Most closely related to our work are the recent works on glass ceiling algorithms [5, 24], which show that under-representation of minorities can be worsened by recommendation algorithms, due in large part to homophily in social networks.

One closely related work to ours is that by Masrour *et al.* [20], which proposes a dyadic-level fairness criterion based on Newman's modularity measure [22]. This work proposes FLIP, an adversarial learning approach to alleviate the filter bubble problem. The main idea of this framework is to ensure that the generator learns a feature representation that is good enough for link prediction while preventing the discriminator from distinguishing between inter-group and intra-group edges. One limitation of this framework is that the adversarial learning model only focuses on the modularity/homophily criterion, which only takes into account the differences of edge density within and between communities, and cannot accommodate higher-level fairness properties.

Another important work approaches the fair link prediction problem from a group accuracy perspective, as is common in the machine learning fairness sector. Li *et al.* define fairness as the difference in accuracy between inter-group and intra-group edge predictions. [17] They argue that one of the main reasons for this difference is the difference in edge densities, and try to achieve the same edge density by deleting or adding edges without changing the structural features of the overall graph, thus mitigate the disparity in accuracy. They introduce a heuristic pre-processing method called FairLP. which removes or adds one least weight (in their work, *weight* is the number of common neighbors) edge each time until the density of inter-group and intra-group edges are same. The limitation of this work is that during this flattening of density, although the predictive accuracy of the test set is not greatly affected, the addition of a large number of low-weight edges (i.e. high-likely non-existent edges) can itself lead to prediction error. Another potential issue is that this method attempts to balance inter- and intra-group edges, but not intra-group edges across communities.

Laclau *et al.* also focus on statistical disparaty, and extend the metric to multi-class link prediction. They propose a repairing procedure known as Optimal Transport (OT) for the adjacency matrix with a trade-off between the group and individual fairness [16]. FairLP and OT have similar shortcomings as FLIP, and can only focus on a single criterion.

2.3 The Multi-Armed Bandit (MAB) Problem

The multi-armed bandit problem is a classic problem in probability theory, and belongs to the category of reinforcement learning [15]. It is aimed at training an algorithm that can dynamically adjust its choices according to different types of feedback returned by the choices in a potentially evolving environment. Imagine that a gambler is presented with N slot machines, and does not know in advance the true profit distribution of each slot machine. The challenge is in determining which slot to play the next time in order to maximize the profit from beginning to end. By playing more and more slots, the gambler learns more about the behavior of the various machines. Several algorithms have been proposed to solve the multi-armed bandit problem, such as the $\epsilon-$greedy method [15].

In more general scenarios, the multi-armed bandit problem has a wide range of applications. In particular, in the link prediction problem, there may be N potential edges to predict, but we do not know in advance the probability that various types of edges exist. This makes multi-armed bandit algorithms a good fit. Our goal is to predict edges that are likely to be correct predictions, but also have a positive effect on fairness.

3 PROBLEM STATEMENT

We assume that we are given the following input:

(1) An attributed network $G = \langle V, E, A \rangle$, where V represents the set of nodes and $E \subseteq V \times V$ is the set of edges. G is assumed to be a subgraph of an unobserved underlying network H.
(2) Attribute matrix $A \in \mathbb{R}^{|V| \times d}$, a d-dimensional matrix describing the d attributes associated with each node in V. These attributes might be inferred from the network topology (e.g., centrality), if desired.
(3) An attribute s (represented in A) that has been denoted as the sensitive attribute of interest. Such an attribute might represent, for instance, race or gender. Fairness will be computed with respect to this attribute.
(4) A fairness metric $\mathbf{F}$, which is a function of s and G.
(5) A prediction budget b.
(6) Tunable parameters γ that balance the importance of fairness vs. accuracy (optional).

Goal: This is fundamentally a multi-objective optimization problem. As such, there are are multiple ways to formulate a single objective. (The following formulations assume that higher values of $\mathbf{F}$ correspond to greater values of fairness.) If γ is specified, the goal is to predict b edges that do not already exist in G such that $\gamma \times Acc + (1 - \gamma) \times \mathbf{F}(G')$ is maximized, where Acc is the fraction of those b edges that exist in G_0, and G' is the network obtained by adding those correctly predicted edges to G. If γ is not specified by the user, the goal is to maximize fairness subject to an accuracy constraint (one could also formulate the opposite: maximizing accuracy subject to a fairness constraint). In our work, we follow the latter approach, with a lower bound accuracy of 0.8, and perform a grid search to find the value of γ that achieves this.

Importantly, we assume that these predictions can be made *sequentially*. That is, after each prediction is made, the network is updated to include that edge, if correctly predicted. In other words, we learn whether an edge was correctly or incorrectly predicted immediately after making the prediction. (We allow for a small number of multiple edges to be predicted at once if needed for efficiency reasons.)

4 THE FAIRLINK FRAMEWORK

4.1 Overview

Our proposed framework uses a multi-armed bandit algorithm to maintain an up-to-date model according to the past long-term interaction behavior, while weighting most recent observations to account for changes in graph structure as edges are added. The process consists of the following four steps:

(1) **Identifying Candidate Edges**: In the extreme case, the candidate links may consist of all edges that do not exist in the observed network; however, for the sake of efficiency (time and space), one might choose to reduce this search space by, e.g., only considering node pairs that have at least some number of neighbors in common (Section 4.2).
(2) **Slot Generation**: The candidate links are divided into n clusters (slots) based on link characteristics (including topological and attribute-related properties), or fairness related characteristics (Section 4.3.)
(3) **Edge Selection**: In each iteration, the desired number of links are selected by the multi-armed bandit algorithm (Section 4.4).
(4) **Benefit Computation**: Once the links are predicted to users, the fairness benefit of these predictions is computed. If the link is accepted, it will be added into the network and the benefit is computed based on the change in value of the fairness metric. Otherwise, the link is discarded and the benefit set to 0 (Section 4.5).
(5) **Reward Update**: The benefit of the cluster(slot) containing the predicted links is updated (Section 4.6).

Pseudocode for `FairLink` is provided in Algorithm 1.

4.2 Identifying Candidate Edges

The set C of candidate edges is the set of all edges that do not exist in the network. However, this set is likely to be extremely large; and, moreover, contains many edges that are highly unlikely to exist. Thus, in practice, one can apply any reasonable pruning technique to shrink this set: for example, considering only node-pairs with at least one (or some other number) neighbor in common.

In our work, we identify the candidate edges $C \subseteq V \times V - E$ as a randomly selected subset of edges that do not already exist in the network. We do this purely for purposes of demonstrating the framework, primarily because it was the same assumption made in the experimental setup in [20], for sake of equal comparison between algorithms.

4.3 Slot Generation

The multi-armed bandit action space is constructed by partitioning the set of candidate edges C into n clusters $c_1, c_2, c_3 ... c_n$. Each clusters contains node-pairs with similar structural and attribute properties (including the effect of adding that edge on the network fairness). Ideally, edges in the same cluster have roughly equal probability of existing in the true network, and similar fairness

properties. Correctly separating the candidate edges into these clusters is critical to the performance of the framework.

The features used to generate these clusters are network and application dependent. In our experiments, we use *k-means* clustering to cluster the edges according to strucutral information and well as fairness metric-specific features. Full details are provided in Section 5. For instance, if the fairness metric is homophily, we use common neighbor diversity which defined at section 5.4, and if the fairness metric is information fairness, we use the MaxFair score, which measures the effect of adding an edge on information fairness [14]. Other features can be used as desired: the user has flexibility in determining features as well as how the edge candidates are clustered.

4.4 Edge Selection

Here, we use a multi-armed bandit algorithm to predict the next link. The action of the agent is choosing one cluster c_j among the n clusters described above, and then randomly select one link $e_j \in c_j$ from this group. In Section 4.7, we discuss how to use a batching process to speed this step up.

The literature contains numerous multi-armed bandit algorithms [30]. Because our goal is to illustrate the framework, rather than recommend a specific multi-armed bandit algorithm, we use the classic $\epsilon - Greedy$ algorithm, which is the simplest and most widely used strategies to solve multi-armed bandit problem [30]. $\epsilon - Greedy$ tries to find the balance between exploration and exploitation: given a user-defined $\epsilon \in [0, 1]$, with ϵ probability, the agent randomly chooses a slot, while with $1 - \epsilon$ probability, the agent select the highest reward slot based on history observed so far.

`FairLink` can be be modified to fit the user's needs and computational power: e.g., by using a multi-armed bandit algorithm of choice, by internally ranking the edges in selected group c_j by another link-prediction algorithm, or even applying an adversarial bandit algorithm to deal with the impact of network evolution on the rewards distribution. Our goal in this work is to propose and evaluate `FairLink` as a general framework.

4.5 Benefit Computation

After a link has been predicted, the framework computes the benefit of these links with respect to the target fairness criterion.

We define the fairness benefit as follows:

$$B_e = \begin{cases} \frac{F_{G^* \cup e} - F_{G^*}}{F_{G^*}} & \text{if e is accepted} \\ 0 & \text{if e is refused} \end{cases} \tag{1}$$

where G^* is the network obtained after the previous iteration, $F_{G^* \cup e}$ is the fairness measure of the network after accepting edge e, the edge predicted in the current iteration, and F_{G^*} is the fairness measure of the network after the previous iteration (i.e. the network before the current prediction). A positive fairness benefit means that adding e will improve the fairness of the network. (Note that the signs must be reversed if using an *unfairness* metric, rather than a *fairness* metric.)

4.6 Reward Update

Once the fairness benefit is computed, the final step is updating the selected cluster c_j's comprehensive reward. The comprehensive reward consists of two parts: accuracy and fairness benefits.

Because the fairness benefits of links change as edges are added to the network, the importance of benefit values decreases over time– e.g., if an edge is added early on in the process, its fairness reward is not necessarily of great relevance to predictions later in the process. Thus, we introduce the following decay function for selected group c_j:

$$benefit_{new} = \alpha * benefit_{old} + (1 - \alpha) * B_e, \tag{2}$$

where $benefit_{old}$ is the fairness benefit of the selected cluster c_j before the current round, and $benefit_{new}$ is the updated fairness benefit of the selected cluster c_j in this round. This decay function allows the framework to adapt to changes in fairness behavior. α are user-set parameters that control the speed of decay. If α equals to 0, the framework only consider the latest fairness benefit. Users can set parameters empirically in proportion to application scenario and network property. Through multiple experiments, we find $\alpha = 0.8$ is a choice to get good performance.

Ultimately, the comprehensive reward of the selected cluster c_j is calculated according to the following merge function:

$$R = \gamma * benefit_{new} + (1 - \gamma) * accuracy \tag{3}$$

where $benefit_{new}$ is the updated fairness benefit, and $accuracy$ is the overall accuracy rate of selected cluster c_j. The user can set the values γ to make the trade-off between the benefit of fairness improvement and the risk of incorrect predictions.

4.7 Batching Process for Efficiency

Because updating rewards after every prediction can be slow, we propose an optional batch process method. In each round, m edges from selected group c_j are predicted, and the benefit of this set is calculated as:

$$B_e = \frac{F_{G^* \cup e'} - F_{G^*}}{F_{G^*}} \tag{4}$$

where $e' \subseteq e$ is the subset of predict edges which accepted by users. Like any batching method, while this method improves the speed of the framework, it can reduce the performance.

4.8 Parameter Selection for FairLink

There are multiple hyperparameters in the `FairLink` model. If not specified, these can be set with a grid search, as described below.

- the γ parameter is the importance of fairness improvement benefit to the comprehensive reward. This parameter controls whether the model is more inclined to obtain accurate predictions or to obtain greater improvements in fairness. γ may be set by user; but if its value is not provided, the problem can simplify to a constraint-based problem of achieving the maximum fairness given a specified lower bound on accuracy (or vice versa). In such cases, a grid search can be used to select an appropriate γ.
- ϵ refers to the probability of choosing to explore in $\epsilon - Greedy$ algorithm. If $\epsilon = 0$, `FairLink` never explores, and if $\epsilon = 1$, `FairLink` never learns.

- The decay parameter α controls the impact of historical data on the FairLink. For a very small network into which adding an edge will greatly change the structure of the network, a larger decay parameter is needed to enable FairLink is more sensitive to the changes in the observed data.

With the exception of ϵ, which reflects user priorities, we use a grid search to efficiently and systematically find suitable parameters and ensure stable performance.

A set of edges is randomly stripped from G and mixed with the same number of negative edges to form the train data required for grid search. FairLink then performs k-fold cross-validation for the grid search. First, it divides the training data into k equal-sized groups. In each round of verification, FairLink takes the k-1-th group as $train\_set$, and the remaining groups as $valid\_set$. It aggregates the alternative values of all parameters that need to be set, for example, $\gamma \in [0.1, 0.2, 0.3], \epsilon \in [0.01, 0.05, 0.1], \alpha \in [0.3, 0.5, 0.8]$. According to each combination of parameters, we generate a new FairLink model, training the model with $train\_set$ ('training' here is merely adding $train{-}set$ to G_t), use FairLink to predict $valid\_set$, and score its results according to the score function. Finally, we select the group of parameters with the highest average value among all parameter combinations as the final optimal parameter.

When γ is specified, the score function of the grid search is

$$Score = \gamma \times Acc + (1 - \gamma) \times \mathbf{F}(G').$$

When γ is not specified, there are two potential grid search score functions from which the user can select: (1) Maximize the improvement of fairness while ensuring a certain accuracy, and (2) Maximize the correct rate while ensuring a minimum fairness improvement. As an example of the first function, when the validation set accuracy is lower than the accuracy lower bound, the score of this parameter combination is $-\infty$, otherwise, the score is the accuracy of the validation set.

$$Score = \begin{cases} -\infty & \text{if } Acc < bound \\ \mathbf{F}(G') & \text{otherwise} \end{cases}$$

Grid search then returns the parameter combination with the highest score.

Algorithm 1 FairLink Framework

Input:
The attributed network, G ;
The set of candidate edges, C;
Output:
The predicted edges, $e_1, e_2, ..., e_b$;

// Slots Generation
$c_1, c_2, ..., c_n \leftarrow$ SlotsGenerationFunction(G, C);
for $i \leftarrow 1$ to n **do**
  $reward_i,\ benefit_i,\ accuracy_i \leftarrow 0$;
end for
$G^* \leftarrow G$;
for $t \leftarrow 1$ to b **do**
  // Multi-armed Bandit Algorithm selects edge
  $j \leftarrow$MultiArmedBanditsFunction$(c_1, c_2, ..., c_n)$;
  $e_t \leftarrow random(c_j)$;
  // Compute Benefit
  $B_e \leftarrow$ ComputeBenefitFunction(G^*, e_t);
  update $accuracy_j$;
  if e_t is correct **then**
    $G^* \leftarrow G^* \cup e_t$;
  end if
  // Update Rewards
  $benefit_j \leftarrow \alpha * benefit_j + (1 - \alpha) * B_e$;
  $reward_j \leftarrow \gamma * benefit_j + (1 - \gamma) * accuracy_j$;
end for

5 EXPERIMENTS

In this section, we discuss our network datasets and experimental setup. We test our framework and several baseline algorithms on five real datasets under multiple fairness criteria and compare results with respect to fairness and accuracy.

5.1 Datasets

We consider the following datasets, which include friendship networks, co-authorship networks, and citation networks. Protected node attributes include gender, language and political party. In our datasets, the protected attributes are all specified as part of the data. Dataset statistics are shown in Table 1.

- **Facebook100** [29]: These networks represent the early Facebook networks of universities. Every user is associated with various personal characteristics, such as age and department. We use the network of **Bowdoin** College, and measure fairness with respect to the 'Gender' attribute.
- **DBLP** [28]: The DBLP dataset is a computer science co-authorship network, where nodes represent authors and edges represent two authors who have published an article together. From the full DBLP network, we extracted a sample of papers published in top conferences from two subfields– **Data Mining** and **Graphics**– between 2015 and 2019.[1] For each author, we use gender as the protected attribute, where gender is obtained from the genderize.io tool.
- **Pokec** [27]: The Pokec dataset is a popular online social network in Slovakia, where users are associated with gender, age, hobbies, language etc. The original network dataset contains millions of nodes and edges. For demonstration, we extracted the induced subgraph corresponding to individuals living in the Bytča town in the **Zilinsky Kraj** region. Singleton nodes are removed from the network. Individuals who speak English or German are placed in one group and those who speak neither of these languages, but do speak another language are placed in a second group.
- **Polblogs** [1]: The Polblogs dataset is a network of blogs with various political leanings. Each node represents a blog, and edges represent hyperlinks between blogs. Each node has an attribute indicating political leaning liberal or conservative.

5.2 Training/Test Data Generation

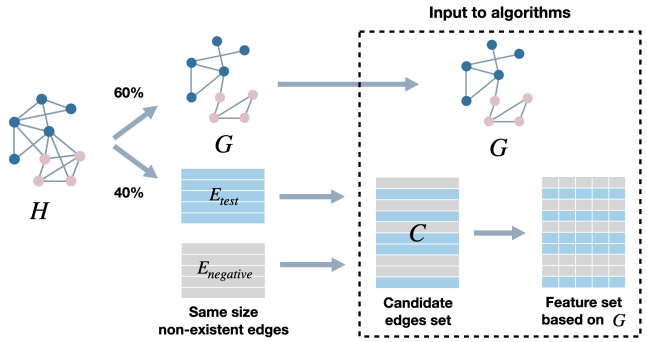
Denote each dataset as an attributed network $H = \langle V, E_H, A \rangle$. We divide the network edges E_H into a sample set E_{sample}, containing a random 60% of the edges in H, and a test set E_{test}, containing the

[1] https://webdocs.cs.ualberta.ca/~zaiane/htmldocs/ConfRanking.html

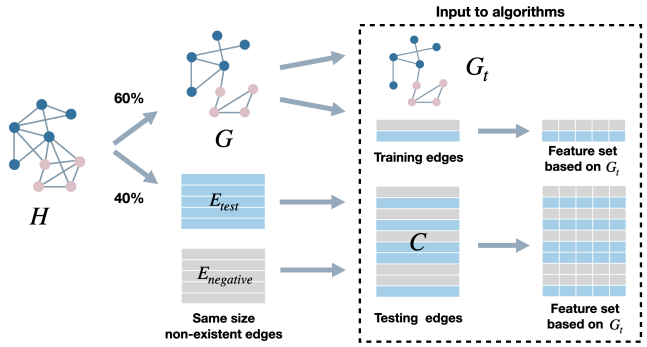
Table 1: Dataset statistics

Name	Bowdoin	Zilinsky	Data Min	Graphics	Polblogs
# Nodes	2.3k	3k	2.3k	3.5k	1.2k
# Edges	84k	23.5k	7.6k	10.4k	16.7k
Homo.	0.02	0.01	0.05	0.07	0.41
IU	0.08	0.42	0.20	0.22	0.77
Intra Den.	0.0184	0.0024	0.0018	0.0010	0.0184
Inter Den.	0.0217	0.0019	0.0032	0.0019	0.0022
Attribute	gender	language	gender	gender	pol. party

remaining edges in H, to simulate a scenario in which users try to predict missing edges. To generate E_{sample} we use a random walk with 10% probability of restart. We chose to use the random walk method in this work because it is a common technique of crawling networks in real world application.



(a) Unsupervised Learning Methods



(b) Supervised Learning Methods

Figure 1: Experimental Setup for Supervised and Unsupervised Learning Methods

We treat those node-pairs that are connected by an edge in E_{sample} as members of the positive class, and all other node-pairs as members of the negative class (even though some are actually in the complete network H). In order to measure the methods' accuracy performance, we add to E_{test} the same number of non-existent links $E_{negative} \cap E_H = \emptyset$ to generate the candidate edge set C. This is the same approach taken in [20], and we use the same basic setup for all methods (in practice, one would need to generate candidate edges in another way: for example, the set of all edges whose endpoints share at least one neighbor). For supervised learning methods, this is considered the test dataset. For unsupervised learning methods, such as `Jaccard` link prediction and `FairLink`, $G = \langle V, E_{sample}, A \rangle$ denotes the sample network.

Because **supervised learning methods**, such as `Node2Vec` and `FLIP`, need training data, we randomly select one-sixth of the edges from E_{sample}, denoted as E_{train}, and mix it with the same number of negative links to construct the training data. The remaining edges $E_{sample'}$ form the sample network G_t for supervised learning methods, denoted as $G_t = \langle V, E_{sample'}, A \rangle$. Details of the data construction processes are illustrated in Figure 1.

5.3 Experimental Overview

Link prediction algorithms can be evaluated in many different ways, and we wish to fairly compare the accuracy, time performance and fairness improvement performance of each methods. First, we describe several key points of our experimental approach:

- We are considering the task of *link prediction*, in which the goal is to predict which edges are missing from an incomplete observed network, rather than *link recommendation*, in which the goal is to suggest new edges to be added to the network (e.g., recommending new friends).
- In general, it would be best to predict a total of $b = 1 \times |E_{test}|$ edges: this would allow a perfectly accurate link prediction method to demonstrate itself. However, this causes problems with some fairness criteria, because when so many edges are added, the homophily/information unfairness simply reverts to that of the original, underlying network: there is no opportunity for the link prediction method to meaningfully improve fairness. This is not an issue with the accuracy disparity criterion, however. Thus, for the accuracy disparity criterion, we predict $b = 1 \times |E_{test}|$ edges, while for the others, we predict $b = 0.1 \times |E_{test}|$ edges from the candidate edges set.
- To determine which edges to add to the network, we sort the candidate edges according to their scores (as computed by each baseline model) and select the top b edges for addition.
- For a fair comparison, because `FairLink` predicts the edges sequentially, we allow the baseline algorithms to also make sequential predictions. Given the user-customized parameter *batch_step*, there are $\left\lceil \frac{b}{batch\_step} \right\rceil$ iterations. In each iteration, after the models predict *batch_step* edges, the remaining candidate edges are passed to the next iteration. The model is retrained, and scores for the remaining candidate edges recomputed.

5.4 Details of FairLink Implementation

`FairLink` is a general framework, and here, we describe the specific details used in our experimental setup.

We characterize edges using the following features:

- The resource allocation index of each candidate edge, measuring structural similarity between endpoints [19].
- The common neighbor diversity of the candidate edge. For a binary attributed network in which node i's attribute A_i is either a_1 or a_2 , the common neighbor diversity of edge $e_{i,j}$

is defined as:

$$D_{i,j} = \frac{\min(|CN_1|,\ |CN_2|)}{\max(|CN_1|,\ |CN_2|)}$$

where CN_d is the set of common neighbors of nodes i and j whose attribute is a_d, i.e.

$$CN_d = \left\{v \mid e_{i,v},\ e_{j,v} \in E_{sample},\ A_v = a_d,\ v \in V\right\}$$

- When the objective is minimizing information unfairness, we use a normalized version of the MaxFair score [14]. For each candidate edge, we compute $score(u,v) = \sum_{f,g} s_{fg} * (vec_f(u) * vec_g(v) + vec_g(u) * vec_f(v))$, where vec_f represents each node's centrality with respect to group C_f (described in [14]). $vec_f(u), vec_f(v), vec_g(u), vec_g(v)$, quantify how well nodes u and v spread information to groups C_f and C_g. Adding a high-scoring edge facilitates information flow between groups.
- The inter-intra group feature distinguishes whether the edge is connecting inside one group or connecting across two groups.

$$\sigma(u,v) = \begin{cases} 0 & \text{if } A_u = A_v \\ 1 & \text{if } A_u \neq A_v \end{cases}$$

Using these features, we divide candidate edges into 20 groups, using k-means clustering.

The reward function is connected to the fairness criterion (described in further detail in Section 5.6). We consider reward functions related to homophily and information unfairness [14]. Initially, all cluster/arm rewards are set to 0, indicating that `FairLink` has no preference between clusters.

We use a ε−greedy policy in the multi-armed bandit algorithm. This means that in each step, with ε probability, the algorithm randomly selects a group from among the n groups, while with $1 - \varepsilon$ probability, the algorithm selects the group with the best reward so far. If we are predicting multiple edges in each iteration for the sake of speed, then the desired number of edges are selected from that group.

The predicted edge e is accepted if e exists in the true network; otherwise, e is refused. In real applications, this can be inferred by observing whether a user accepts a suggestion. Such inference is common in reinforcement learning applications. If the edge is accepted, its fairness benefit is either the decrease in homophily, if homophily is the fairness objective, or the decrease in information unfairness, if information unfairness is the objective; and if it is rejected, then the fairness benefit is 0.

To update group rewards, we use decay parameter $\alpha = 0.8$. For the grid search on γ, we set the lower bound on accuracy to be 0.8. For computational efficiency, we set the batch size m to be 10.

5.5 Baselines

We consider four baseline algorithms. The first two focus only on accuracy, while the third and the fourth also consider fairness criteria– homophily and accuracy disparity, respectively.[2]

- **Jaccard link prediction**: We apply the Jaccard coefficient to score all edges in E_{test}, and sort these edges according to the score that they are given by the algorithm. We then add those predicted edges that exist in the original network to the sample network in descending order of score. Jaccard coefficient of nodes u and v is defined as [18]

$$J(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|},$$

where $\Gamma(u)$ denotes the set of neighbors of u.
- **Node embedding + SVM**: We use a node embedding algorithm (**Node2Vec**) to represent the sample network in a new feature space and generate a Hadamard embedding of the edges. The training data E_{train} consists of two equally sized sets of edges that do and do not exist in the original network, which are treated as the positive class and negative class (we remove the positive training edges from the network before embedding, so that the node embedding algorithm does not learn the proxy feature of edge existence). After feature reduction by PCA, we train a Support Vector Machine(**SVM**) and used grid searching method to find the best slack parameter and kernel function.
- **FLIP**: In this baseline, we applied the `FLIP` framework proposed by Masrour *et al.* [20].[3] Each round, the framework deploys a Generative Adversarial Network to generate a representation of the current network while the training data keeps consistent, which is used for scoring. We sort the test edges based on this score and select the top b edges. We then retrain with the correctly predicted edges.
- **FairLP**: While code for `FairLP` was not available, we were able to implement the **FairLP+AA** version of the algorithm (which in [17] achieved the best results) with the **MaxD** policy. Because the pre-processing method here adds inter-group edges whose endpoints share no common neighbors, we perform an optimization in which edge weights (number of common neighbors) are not updated, but are checked as needed. This does not change results, but improves running time substantially.

We set the batch size of the `Jaccard` and `FairLP` methods to be 10, to match `FairLink`. However, `Node2Vec` and `FLIP`'s training processes are extremely slow, making a batch size of 10 entirely impractical. Thus, we set the batch parameter to be 100 for `Node2Vec` and `FLIP`. (On smaller datasets, where a batch size of 10 was feasible for these methods, we saw nearly identical results regardless of batch size.)

5.6 Evaluation Metrics

We evaluate the various algorithms with respect to both fairness and accuracy. Recall that `FairLink`'s reward function changes according to the fairness objective. We consider the following fairness metrics.

- **Homophily:** Homophily can be used as a measure of segregation. We measure homophily using the modularity coefficient described in Section 2,as implemented by the *community.modularity* function from the NetworkX library in Python.

[2]Optimal Transport code can be found in https://github.com/laclauc/FairGraph, but missing core Module 'ot' make the results not reproducible.

[3]https://github.com/farzmas/FLIP

- **Information unfairness (IU):** Information fairness measures the fairness of information flow between groups in the network. We use a normalized version of the metric described in [14]

$$IU_{G,p,k} = \frac{max(Dist(D_{f_1g_1}, D_{f_2g_2}))}{max(D_{f_3g_3})},$$

where $f_1, f_2, f_3, g_1, g_2, g_3 \in \{1, ...t\}$.

- **Accuracy Disparity (TPD):** True-positive rate disparity (TPD) measures the absolute value of recall difference between inter-group and intra-group edge [17].

$$TPD = \left| Pr\left(\hat{Y} = 1 \mid Y = 1, s = s'\right) - Pr\left(\hat{Y} = 1 \mid Y = 1, s \neq s'\right) \right|,$$

where $\hat{Y}$ is the result of link prediction model and Y is the ground truth, S and S' is the sensitive attribute of edge ends.

For the homophily and information unfairness metrics, we compare the networks before adding correctly predicted edges (**ground truth network**) and after adding correctly predicted edges (**new network**). The fairness improvement is defined as:

$$Fairness\_improvement = \frac{F_{ground} - F_{new}}{F_{ground}}$$

.

We also evaluate the accuracy of the various algorithms. Denote the original network as $G_0 = (V, E)$ and the sample network as $G = (V, E')$, so $E' \subset E$.

Let $Pred_i$ denote the i predicted edges. We then define the accuracy of a link prediction method as:

$$A_i = \frac{|\{u \in Pred_i \cap E\}|}{i}$$

.

In other words, the accuracy is simply the fraction of those edges that actually exist in the original graph.

6 RESULTS

We compare the three FairLink models (one mitigating homophily, one mitigating information unfairness, the third mitigating accuracy disparity) to the baseline methods. Results are shown in Tables 2 - 4. As mentioned in Section 5.3, the accuracy results in Table 4 are different from those in Table 2 and 3 because different numbers of edges are being predicted.

As expected, the two non-fairness focused methods (Jaccard and Node2Vec/SVM) have the highest accuracy. However, although the Jaccard link prediction algorithm has the highest accuracy, its worsening of unfairness is also generally the largest. This is a reasonable result: this method adds edges between nodes that already have many common neighbors, so tends to reinforce group structure, to the extent that that structure is reflected in the network. For example, in the Pokec network, while Jaccard achieves 98.74% precision, it worsens the information fairness by 16.26%, and in the Bowdoin College Facebook network, Jaccard achieves 98.75% precision but increases the homophily by approximately a quarter. While the machine learning methods do not significantly increase unfairness, they also do not mitigate it. We see similar, albeit less dramatic, results with Node2Vec: accuracy is high, but unfairness is often unchanged, or even worsened.

Of the three fairness-focused methods, our proposed method, FairLink, shows the best overall fairness results across fairness metrics, with accuracy less than that of FairLP, but better than that of FLIP. As seen in the 'Average' column of each table, FairLink consistently performs well with respect to fairness, regardless of metric. This demonstrates its purpose well: it is intended to be flexible to any user-specified fairness metric.

In contrast, FLIP and FairLP often worsen the fairness metrics for which they were not designed. For instance, on several datasets, FLIP and FairLP make information unfairness substantially worse.

FairLP shows very high accuracy– higher than the other fairness-focused methods, and sometimes the highest of all considered methods. However, its fairness performance is middling. On average, with the TPD metric– the metric for which it was designed– its performance is best, though FairLink performs best on two of the five networks; and for the other two metrics, its performance is poor. FLIP often shows poor accuracy, and is not a standout for any of the fairness metrics considered.

In conclusion, for applications in which fairness matters, we make the following observations:

- Standard link prediction methods perform poorly with respect to fairness.
- If the user wishes flexibility in choosing a fairness metric, FairLink is the best choice. Across metrics, FairLink consistently performs well. Thus, in contrast to existing fair link prediction methods, which are tailored for specific fairness metrics, FairLink becomes a more universal choice.
- FairLP uses preprocessing to increase the density of edges, so when it predicts only 10% of edges, its accuracy is good. However, when it tries to predict 100% of edges, its accuracy is weaker. It performs well for the TPD metric– for which it was designed– but not for other fairness metrics. For the link prediction task, TPD suffers from serious flaws: most importantly, it does not account for accuracy disparities between protected groups, only between inter-vs. intra-group edges.

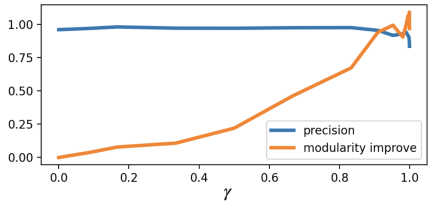


Figure 2: Results on Pokec network as γ (fairness weight) increases.

6.1 Parameter Analysis

Next, we explore the effect of the γ parameter, which balances accuracy and fairness. Figures 2 and 3 show results on the Pokec and Bowdoin network, respectively, as the fairness parameter γ increases. The precision curve shows that FairLink does not struggle with link prediction accuracy. On both networks, there is a sharp increase in fairness with no sacrifice in accuracy, until γ is

Table 2: Accuracy and information unfairness improvement results. Higher is better for both.

Methods	Bowdoin		DBLP-datamining		Pokec		DBLP-Graphic		Polblog		Average	
	Acc.	IU Imp.	Acc.	IU Imp.	Acc.	IU Imp.	Acc.	IU Imp.	Acc.	IU Imp.	Acc.	IU Imp.
Jaccard	**0.9875**	0.1309	**1**	0.041	0.9874	-0.1626	**1**	-0.061	0.8569	-0.155	0.9664	-0.0413
Node2Vec	0.93	0.0585	0.9567	0.0553	0.942	-0.0824	0.98	**0.0211**	0.8686	-0.0391	0.9355	0.0027
FairLP	0.9861	0.0125	0.9944	0.1094	**0.9925**	-0.1981	**1**	-0.235	**0.9931**	-0.1518	**0.9932**	-0.0926
FLIP	0.8282	-0.0803	0.6167	**0.1499**	0.878	-0.1192	0.7	-0.4895	0.9529	-0.0153	0.7952	-0.1109
FairLink	0.8098	**0.1406**	0.9250	0.1486	0.8993	**0.3362**	0.9393	-0.044	0.8388	**0.031**	0.8824	**0.1225**

Table 3: Accuracy and homophily improvement results. Higher is better for both.

Methods	Bowdoin		DBLP-datamining		Pokec		DBLP-Graphic		Polblog		Average	
	Acc.	Hom. Imp.	Acc.	Hom. Imp.	Acc.	Hom. Imp.	Acc.	Hom. Imp.	Acc.	Hom. Imp.	Acc.	Hom. Imp.
Jaccard	**0.9875**	-0.2502	**1**	-0.1166	0.9874	0.0753	**1**	-0.053	0.8569	-0.0203	0.9664	-0.0730
Node2Vec	0.93	-0.0309	0.9567	-0.0278	0.942	-0.0012	0.98	0.0124	0.8686	-0.0124	0.9355	-0.0120
FairLP	0.9861	0.0093	0.9944	-0.0046	**0.9925**	0.1164	**1**	-0.01	**0.9931**	-0.0224	**0.9932**	0.0177
FLIP	0.8282	0.0994	0.6167	0.0048	0.878	0.0179	0.7	-0.0134	0.9529	0.0119	0.7952	0.0241
FairLink	0.9291	**0.7027**	0.95	**0.2944**	0.9698	**1.0992**	0.9837	**0.2589**	0.9020	**0.0347**	0.9469	**0.4780**

Table 4: Accuracy and accuracy disparity (TPD) results. Lower values of TPD are better.

Methods	Bowdoin		DBLP-datamining		Pokec		DBLP-Graphic		Polblog		Average	
	Acc.	TPD	Acc.	TPD	Acc.	TPD	Acc.	TPD	Acc.	TPD	Acc.	TPD
Jaccard	0.8400	0.0200	0.7575	0.0180	0.6554	0.0215	0.7301	0.0095	0.8	0.1863	0.7566	0.0511
Node2Vec	0.7995	0.0116	**0.8788**	0.0162	**0.8008**	0.0411	**0.8422**	0.0197	0.7682	0.2821	**0.8179**	0.0741
FairLP	**0.8472**	0.01	0.7572	0.0208	0.6552	**0.0205**	0.7301	0.0078	0.7543	**0.0649**	0.7488	**0.0248**
FLIP	0.6984	0.0067	0.5551	0.0460	0.6879	0.024	0.5524	**0.0038**	0.7897	0.071	0.6567	0.0303
FairLink	0.8168	**0.0018**	0.7919	**0.0024**	0.7330	0.0349	0.8114	0.0152	**0.8174**	0.1352	0.7941	0.0372

Table 5: Running times (seconds)

	Bowdoin	DBLP-DM	DBLP-GR	Pokec	Polblog
Jaccard	2.7k	83	280	1.1k	97
Node2Vec	73k	1.1k	4.5k	18k	5.1k
FairLP	3.9k	150	532	1.6k	216
FLIP	62k	5.4k	16k	42k	3.8k
FL (IU)	3.4k	250	960	3.3k	910
FL (Hom)	160	2.3	3.9	22	6.8

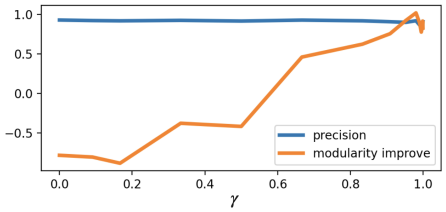


Figure 3: Results on Bowdoin network as γ (fairness weight) increases.

almost 1, when accuracy decreases rapidly. $\gamma = 1$ indicates that the method is entirely concerned with fairness, and predicting a few edges that are unlikely to exist but have a high fairness-benefit edges eventually yields higher rewards than predicting many low-benefit high-confidence edges. Because link prediction only adds correct edges, the decease of accuracy means that overall fewer edges will be added to the network, which also affects the upper bound of performance of fairness improvement. Thus, the critical point before accuracy drops sharply can be considered as an approximate Pareto efficient point. This gives some guidance to users in selecting an appropriate γ value; however, as discussed earlier, if the user wishes to use a grid search to find γ, that can be done instead.

6.2 Running Time:

Running time results are shown in Table 5. In comparison to `FLIP` and `Node2Vec/SVM`, `FairLink` is extremely fast. This is primarily because of training time. In each iteration, `FLIP` and `Node2Vec` spend a huge amount of time producing new network embeddings and retraining. In contrast, `FairLink` 'retrains' (updates rewards) on the fly, with no large-scale retraining required. `FairLP` is also fairly fast, comparable to `FairLink`.

7 CONCLUSION

In this work, we introduced `FairLink`, a general multi-armed bandit-based framework for fair link prediction in networks, which allows users to specify the fairness metric of interest. In contrast, the existing method for fair link prediction uses a fixed homophily-based fairness criterion. We conducted an experimental analysis over five real-world networks of different domains, and demonstrated that `FairLink` consistently achieves a balance of accuracy and fairness with respect to the specified fairness metric.

ACKNOWLEDGMENTS

Wang and Soundarajan were supported by NSF Awards #908048 and #2047224.

REFERENCES

[1] Lada A Adamic and Natalie Glance. 2005. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*. 36–43.
[2] Gediminas Adomavicius and YoungOk Kwon. 2012. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2012), 896–911.
[3] Luca Maria Aiello and Nicola Barbieri. 2017. Evolution of ego-networks in social media with link recommendations. In *WSDM*. ACM, 111–120.
[4] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. 2006. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, Vol. 30. 798–805.
[5] Chen Avin, Barbara Keller, Zvi Lotker, Claire Mathieu, David Peleg, and Yvonne-Anne Pignolet. 2015. Homophily and the glass ceiling effect in social networks. In *ITCS*. 41–50.
[6] Robin Burke, Nasim Sonboli, and Aldo Ordonez-Gauger. 2018. Balanced Neighborhoods for Multi-sided Fairness in Recommendation. In *FAT**.
[7] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2017. How Algorithmic Confounding in Recommendation Systems Increases Homogeneity and Decreases Utility. *arXiv preprint arXiv:1710.11214* (2017).
[8] Hsinchun Chen, Xin Li, and Zan Huang. 2005. Link prediction approach to collaborative filtering. In *JCDL*. IEEE, 141–142.
[9] SA Curiskis, TR Osborn, and PJ Kennedy. 2015. Link prediction and topological feature importance in social networks. In *Australasian Data Mining Conf., Australian Comp. Soc. Inc.* 39–50.
[10] Elizabeth M Daly, Werner Geyer, and David R Millen. 2010. The network effects of recommending social connections. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 301–304.
[11] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2018. Political discourse on social media: Echo chambers, gatekeepers, and the Price of bipartisanship. *arXiv preprint arXiv:1801.01665* (2018).
[12] Anatoliy Gruzd, Kathleen Staves, and Amanda Wilk. 2011. Tenure and promotion in the age of online social media. *Proceedings of the American Society for Information Science and Technology* 48, 1 (2011), 1–9.
[13] Jindong Gu and Daniela Oelke. 2019. Understanding Bias in Machine Learning. *arXiv:1909.01866* (2019).
[14] Zeinab S Jalali, Weixiang Wang, Myunghwan Kim, Hema Raghavan, and Sucheta Soundarajan. 2020. On the Information Unfairness of Social Networks. In *SDM*.
[15] Volodymyr Kuleshov and Doina Precup. 2014. Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028* (2014).
[16] Charlotte Laclau, Ievgen Redko, Manvi Choudhary, and Christine Largeron. 2021. All of the fairness for edge prediction with optimal transport. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1774–1782.
[17] Yanying Li, Xiuling Wang, Yue Ning, and Hui Wang. 2022. FairLP: Towards Fair Link Prediction on Social Network Graphs. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 16. 628–639.
[18] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.
[19] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2017. A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR)* 49, 4 (2017), 69.
[20] Farzan Masrour, Tyler Wilson, Heng Yan, Pang-Ning Tan, and Abdol Esfahanian. 2020. Bursting the filter bubble: Fairness-aware network link prediction. In *AAAI*, Vol. 34. 841–848.
[21] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.
[22] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.
[23] Javier Sanz-Cruzado, Sofía M Pepa, and Pablo Castells. 2018. Structural Novelty and Diversity in Link Prediction. In *WWW*. 1347–1351.
[24] Ana-Andreea Stoica, Christopher Riederer, and Augustin Chaintreau. 2018. Algorithmic Glass Ceiling in Social Networks: The effects of social recommendations on network diversity. In *WWW*. 923–932.
[25] Jessica Su, Aneesh Sharma, and Sharad Goel. 2016. The effect of recommendations on network structure. In *WWW*. 1157–1167.
[26] Lionel Tabourier, Anne-Sophie Libert, and Renaud Lambiotte. [n. d.]. Predicting links in ego-networks using temporal information. *EPJ Data Science* 5, 1 ([n. d.]).
[27] Lubos Takac and Michal Zabovsky. 2012. Data analysis in public social networks. In *International scientific conference and international workshop present day trends of innovations*, Vol. 1.
[28] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *KDD*. 990–998.
[29] Amanda L Traud, Peter J Mucha, and Mason A Porter. 2012. Social structure of Facebook networks. *Physica A: Statistical Mechanics and its Applications* 391, 16 (2012), 4165–4180.
[30] Joannes Vermorel and Mehryar Mohri. 2005. Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*. Springer, 437–448.
[31] Sarita Yardi and Danah Boyd. 2010. Dynamic debates: An analysis of group polarization over time on twitter. *Bulletin of science, technology & society* 30, 5 (2010), 316–327.