Negative Sampling for Hyperlink Prediction in Networks

Prasanna Patil(✉), Govind Sharma, and M. Narasimha Murty

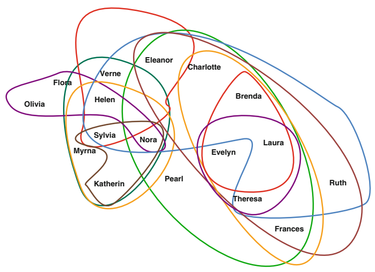
Indian Institute of Science, Bangalore, India
{patilk,govinds,mnm}@iisc.ac.in

Abstract. While graphs capture pairwise relations between entities, hypergraphs dealPrasanna with higher-order ones, thereby ensuring losslessness. However, in hyperlink (*i.e.*, higher-order link) prediction, where hyperlinks and non-hyperlinks are treated as "positive" and "negative" classes respectively, hypergraphs suffer from the problem of extreme class imbalance. Given this context, "negative sampling"—under-sampling the negative class of non-hyperlinks—becomes mandatory for performing hyperlink prediction. No prior work on hyperlink prediction deals with this problem. In this work, which is the first of its kind, we deal with this problem in the context of hyperlink prediction. More specifically, we leverage graph sampling techniques for sampling non-hyperlinks in hyperlink prediction. Our analysis clearly establishes the effect of random sampling, which is the norm in both link- as well as hyperlink-prediction. Further, we formalize the notion of "hardness" of non-hyperlinks via a measure of density, and analyze its distribution over various negative sampling techniques. We experiment with some real-world hypergraph datasets and provide both qualitative and quantitative results on the effects of negative sampling. We also establish its importance in evaluating hyperlink prediction algorithms.

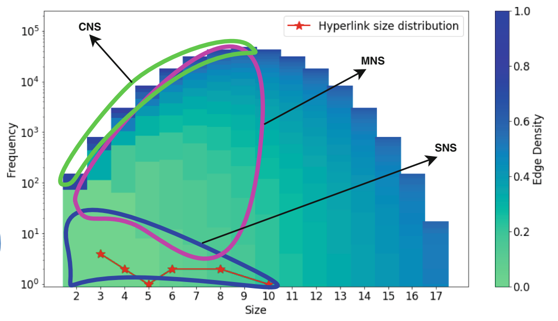
Keywords: Negative sampling · Hyperlink prediction · Hypergraphs · Class imbalance · Hypergraph sampling

1 Introduction

Although the problem of hyperlink prediction (HLP) has not been explored much, we have enough literature on the topic [3,19,21,22] (and much more on its graph-variant, *viz.*, link prediction (LP) [11,14,15,18]) to vouch for its importance. When posed as a supervised learning problem, where "presence of hyperedge" and "absence of hyperedge" are the *positive* and *negative* classes respectively, HLP suffers from extreme class imbalance (ECI), with positive class being the minority one. ECI haunts LP too, and has been thoroughly discussed in the literature as well [7,12,13,20], but with HLP, the situation is much worse owing to the arbitrariness in the number of nodes allowed in a hyperedge. Hence, the solutions provided to combat ECI in usual networks (graphs) for LP could not be directly extended to HLP, at least not without a careful analysis thereof.



(a) The SWC hypergraph



(b) Edge density and size distribution of non-hyperlinks

Fig. 1. The Southern Women Club (SWC) events hypergraph [6]. For practical reasons, we have excluded one 11-sized hyperedge from the original hypergraph. (Color figure online)

We consider the Southern women club (SWC) social hypergraph from Davis et al. [6] illustrated in Fig. 1a that connects eighteen women through twelve hyperlinks, each corresponding to an event they had attended together. All non-hyperlinks from the hypergraph have been plotted in Fig. 1b, with the color shade in each vertical bar denoting edge-density (ref. Definition 1) distribution for a given hyperedge size. Although this being a dense hypergraph is atypical of real-world hypergraphs, we could notice the existence of zero-density non-hyperlinks in the left bottom corner of the plot. However, even for such small a hypergraph, one could compare the positive class size (denoted by red asterisks) *w.r.t.* that of the negative class (the entire histogram), only to reinforce the existence of ECI in a HLP problem.

Of all the solutions available in the literature to treat ECI, *majority-class sub-sampling* (here, *negative sampling* (NS)) is the one that has been prescribed strongly. Other methods (*e.g.*, minority-class over-sampling [5]) further increase the burden on HLP by necessitating computation of prediction scores for each point in the over-sampled positive class as well as those in the negative class (which is already huge in number). Where on one hand, NS makes the HLP problem computationally tractable, on the other, it poses the danger of misinterpretability of results (comparing two HLP algorithms, for instance) due to test set undersampling. The threat has been thoroughly argued about by Lichtenwalter et al. [12,13] and Yang et al. [20] for LP.

In the present work, we provide an extensive analysis of NS for HLP, but since a hypergraph has enormous number of negative patterns, our analysis is limited to a handful of NS algorithms. We propose four different approaches for NS: *Uniform Negative Sampling* (UNS), *Sized Negative Sampling* (SNS), *Motif Negative Sampling* (MNS), and *Clique Negative Sampling* (CNS), with the last three of them focused on the regions bounded by blue, pink and green boundaries in Fig. 1b. Of the four, UNS and SNS are both motivated by random NS in LP [2], and have already been used in the literature to predict new recipes [21] and new email interactions [19]. We derive MNS from a motif-based representative subgraph sampling [4,9] and CNS is our attempt towards developing a 1-hop based equivalent of NS [13] to HLP.

2 Background and Notation

A *temporal hypergraph* is defined as $H = (V, F, T)$, where V is the set of *vertices*, $F \subseteq \wp(V)$[1] is the set of *hyperedges/hyperlinks* over them, and $T : F \to \mathbb{R}$ is a *timestamp specifier function* that maps each hyperlink $f \in F$ to a timestamp $T(f) \in \mathbb{R}$ corresponding to its first occurrence. We define its *clique expanded graph* [1] as $G_H := \eta(H) := (V, E_F, T)$, where $E_F := \eta(F) := \bigcup_{f \in F} E_f$, where, for $f \in F$, $E_f := \eta(f) := \wp_2(f)$ denotes the set of f's induced edges.

In this work, we consider the "temporal" HLP problem, wherein we take a "past" snapshot of H (or *observed* hypergraph), and predict "future" (*unobserved*) hyperlinks. We first define an *HLP triplet* as $(H_{obs}, F_{unobs}, \hat{F}_{sam})$, containing *observed hypergraph* H_{obs}, *unobserved hyperlinks* F_{unobs}, and *sampled non-hyperlinks* $\hat{F}_{sam} \subseteq \hat{F}_{all} := \wp(V) \setminus F$. The HLP problem could be defined as learning/formulating a predictor $\mathcal{P} : F_{unobs} \uplus \hat{F}_{sam} \to \mathbb{R}$ mapping potential hyperlinks $f \in F_{unobs} \uplus \hat{F}_{sam}$ to prediction scores $\mathcal{P}(f)$ proportional to their probabilities of being hyperlinks.

3 State-of-the-Art

A rigorous study of HLP began with the near-seminal works [1,23] on hypergraph Laplacian and spectral clustering methods. Xu et al. [19] explore the latent representation of hyperlinks obtained via those of nodes and a novel entropy-based approach to combine them. More recently, Zhang et al. [21,22] proposed CMM, which is the current state-of-the-art for HLP. Benson et al. [3] study the evolution of hyperlinks of size 3 and 4 in a hypergraph.

As mentioned before, HLP in hypergraphs is analogous to LP in graphs. Though sampling non-links are necessary when LP is posed as a classification problem, it has received little attention in the literature [2,11,13]. Most works randomly sample non-links, which is still justified since the space of possible non-links is *polynomial* in $|V|$. However for hypergraphs, where the space of all possible non-hyperlinks is *exponential* in $|V|$, carefully devised non-hyperlink sampling approaches are mandatory.

Sampling non-hyperlinks is akin to subgraph sampling, which is commonly performed to sample frequent patterns from graphs. More recently, there has been enough attention on mining frequent patterns called *motifs* in a graph to understand the evolution of edges therein. A motif of size k is a k-connected component of the graph. There exists randomized methods such as Mfinder [9] and GUISE [4] to mine these frequent motifs. One of our NS methods (MNS) has been inspired from such motif sampling techniques.

[1] For a set S, let $\wp(S) := \{X \mid X \subseteq S\}$ and $\wp_k(S) := \{X \in \wp(S) : |X| = k\}$.

4 Methodology

4.1 Characterizing Hardness of Prediction

Yang et al. [20] suggest avoiding sampling the test data as much as possible, so that LP could be evaluated fairly. But under unavoidable circumstances, test set has to be sampled, although we propose doing so not without acknowledging some notion of "hardness" in predicting hyperlinks.

Benson et al. [3] point out several properties of a hyperlink $f = \{v_1, \cdots, v_s\}$ that play a key role in its evolution in a hypergraph over time: (i) the *connectivity* among its incident nodes $v_1, \cdots, v_s$ in the projected graph $G_H := \eta(H)$ right before f was formed, and (ii) the *strength* of these connections. These observations can be generalized to arbitrary-sized hyperlinks through the notion of a hyperlink's *edge-density* (ED) defined as follows.

Since ED plays an important role for hyperlink evolution, it could be used to characterize the "hardness" of HLP. In layman terms, hardness in predicting the true class of a test non-hyperlink $f \in \hat{F}_{all}$ denotes how hard it is to predict f as a pattern from the negative class. Let us formally define it as follows:

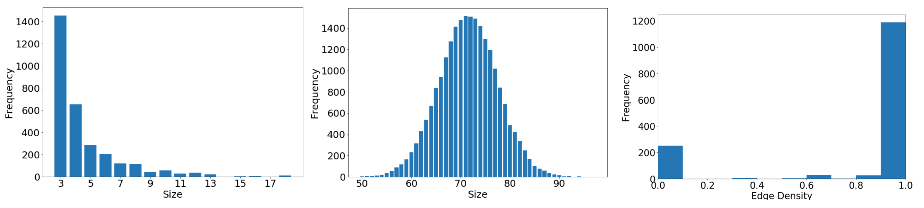
Definition 1 (Hardness of a non-hyperlink). *Given a HLP triplet* $(H_{obs}, F_{unobs}, \hat{F}_{all})$, *the hardness* $h : \hat{F}_{all} \to [0, 1]$ *of predicting the true class of a non-hyperlink* $\hat{f} \in \hat{F}_{all}$ *is defined as one being proportional to its* ***edge-density*** $d(\hat{f}; H_{obs})$ *defined as:*

$$h(\hat{f}) \;\propto\; d(\hat{f}; H_{obs}) := \frac{2 \cdot \left|\eta(\hat{f}) \cap \eta\left(F_{obs}\right)\right|}{|\hat{f}| \cdot \left(|\hat{f}| - 1\right)}, \tag{1}$$

4.2 Uniform Negative Sampling (UNS)

This is the easiest of the four NS algorithms we describe in this section. For a hypergraph $H = (V, F)$, the UNS algorithm picks a sample of k non-hyperlinks $\hat{F}_{sam}$ uniformly at random from the set of all non-hyperlinks $\hat{F}_{all}$. The non-hyperlink sizes of $\hat{F}_{sam}$ are expected to be binomially distributed, which could be validated by Fig. 2b, which shows the size-distribution (SD) of non-hyperlinks in $\hat{F}_{sam}$ for one dataset. Figures 2a and 2b show SD of the positive and negative classes respectively.

As is clear from Fig. 2b, UNS substantially blows-up the non-hyperlink sizes, where median would be around $|V|/2$, which for a 1000-node network amounts to 500-node non-hyperlinks, which is impractical for almost all applications. Moreover, in the context of HLP, since the positive class (of hyperlinks) has an excessively left-skewed distribution, we could end-up solving HLP using a single trivial feature, *viz.*, "hyperlink size"! Hence, we limit our discussion on UNS merely to theory, and recommend **it never to be used in practice**. Neither do we conduct any HLP experiments for UNS in this paper. If it were for UNS, it would sample non-hyperlinks from the entire set shown in Fig. 1b uniformly at random.



(a) SD of hyperlinks and non-hyperlinks sampled via SNS, MNS, and CNS.

(b) SD of non-hyperlinks sampled via UNS

(c) EDD of hyperlinks

Fig. 2. **(a, b)** Size distribution (SD) of hyperlinks and non-hyperlinks sampled via UNS, SNS, MNS, and CNS in `email-Enron`; and **(b)** edge density distribution (EDD) of hyperlinks.

4.3 Sized Negative Sampling (SNS)

SNS overcomes the shortcomings of UNS by sampling non-hyperlinks such that their SD matches that of hyperlinks. SNS is a slight variant of UNS in that the target SD (*i.e.*, that of the sampled negative class) $Pr_{-}(S = s)$ is fixed to that of the positive class ($Pr_{+}(S = s)$), and not a binomial as per Fig. 2b. Once a size s has been sampled according to $Pr_{+}(S = s)$, a non-hyperlink is sampled randomly. The SD of non-hyperlinks sampled with SNS exactly follows the positive class SD (see Fig. 2a).

Since SNS fixes the "size-blow-up" issue in sampled non-hyperlinks, it should be the one-stop solution to negative sampling. But there is yet another problem with SNS—a subtler one at it: since real-world hypergraphs are heavily sparse (much sparser than graphs), sampling non-hyperlinks via SNS biases the binary classification problem *w.r.t.* the challenge in predicting the true class of a test non-hyperlink. As we have already characterized the *hardness* of predicting the true class of a non-hyperlink in Definition 1 via ED, we could monitor the edge-density distribution (EDD) of non-hyperlinks sampled via SNS against hyperlinks.

Figure 3a shows the edge density distribution (EDD) of non-hyperlinks sampled via SNS. It can be seen that most of these non-hyperlinks (negative patterns) have low ED, which makes it "easy" for a HLP algorithm to reject them as positive patterns, whose EDD has been plotted in Fig. 2c. Since most hyperlinks have a high ED, it could be assumed that ED among an arbitrary set of nodes has a positive correlation with their probability of forming hyperlinks in the future. The positive class EDD (Fig. 2c) shows that in most cases, incident nodes of a test hyperlink are well-connected with each other—a pattern not observed for non-hyperlinks sampled via SNS (Fig. 3a). Hence, **an SNS based positive-negative split not only poses little challenge to a predictor trained on such a dataset, but also misleads HLP evaluation.** An SNS algorithm would sample non-hyperlinks from within the "blue" enclosure depicted in the bottom left corner of Fig. 1b. This paves way for yet another NS algorithm: MNS.

Algorithm 1: The MNS algorithm

```
Input: A hypergraph $H = (V, F)$ and size $s$ of the non-hyperlink to be sampled
Output: Sampled non-hyperlink $\hat{f}$
1  $E = \eta(F)$                          // Edges of induced graph $G_H = \eta(H)$
2  $e_0 = \text{RandomChoice}(E)$         // sample initial edge uniformly at random
3  $\hat{f} = \{u \mid u \in e_0\}$       // set $\hat{f}$ to nodes of initial edge $e_0$
4  while $|\hat{f}| < s$ do
5      $S = \{e \in E : |e \cap \hat{f}| = 1\}$
6      if $S = \emptyset$ then
7          go to 2
8      $e = \text{RandomChoice}(S)$
9      $\hat{f} = \hat{f} \cup \{u \mid u \in e\}$
10 return $\hat{f}$
```

4.4 Motif Negative Sampling (MNS)

The hardness of predicting a non-hyperlink $\hat{f} \in \hat{F}_{all}$ to be of the negative class (*i.e.*, True Negative Rate) depends upon the intra-connectivity structure of $\hat{f}$. We have seen that *SNS trivializes the HLP problem by sampling low-density non-hyperlinks* thereby skewing the EDD for negative class towards the left (Fig. 3a). This makes it easy for an HLP algorithm to discriminate it with the positive class (for which the EDD is skewed towards the right (Fig. 2c)). To address this issue, we propose an approach that samples connected subgraph components (CCs) from the clique-expanded graph $G_{H_{obs}} := \eta(H_{obs})$ of H_{obs}. The nodes of these CCs then form the sampled non-hyperedge.

We propose Motif Negative Sampling (MNS) that uses *Mfinder* [9], which is a stochastic algorithm used to estimate the *concentration* of a particular motif in a graph without exhaustive enumeration. Our aim here is to sample non-hyperlinks that are harder to reject by an HLP algorithm, as compared to those sampled by SNS.

Algorithm 1 samples a non-hyperlink of size s by sampling a s-connected component from the underlying graph $G_H = \eta(H)$ of a hypergraph H. Note that there could be more links between a sampled set of nodes than those chosen by the MNS algorithm, and all of them ultimately form the non-hyperlink.

Figure 3b shows the distribution of ED of non-hyperlinks sampled via MNS. It is clear that the number of non-hyperlinks having high ED are quite high as compared to that using SNS (Fig. 3a). Moreover, it is clear to see that ED of any non-hyperlink $\hat{f}$ sampled using MNS (Algorithm 1) satisfies the following: $\frac{2}{|\hat{f}|} \leq d(\hat{f}; H) \leq 1$. This is due to the fact that MNS gives connected subgraphs, and hence, the ED of small-sized non-hyperlinks is likely to be high. Non-hyperlinks sampled via MNS would occupy the "pink" region indicated in Fig. 1b.

Algorithm 2: The CNS algorithm

```
  Input: A hypergraph $H = (V, F)$, $s$ = size of non-hyperlink to be sampled
  Output: Sampled non-hyperlink $\hat{f}$
1 $f_0 = \textsc{RandomChoice}(F)$                    // Randomly sample a hyperlink
2 $V_f = \{u \mid u \in f_0\}$                                         // Nodes of $f_0$
3 $v_0 = \textsc{RandomChoice}(V_f)$          // Randomly sample a node for removal
  /* Randomly select a node from the neighborhood of $f_0 \setminus \{v_0\}$        */
4 $V_n = \{u \in V \mid \exists f \in F \text{ s.t. } \{u, v\} \subseteq f,\ \forall v \in f_0 \setminus \{v_0\}\}$
5 if $V_n = \emptyset$ then
6     go to 1
7 $v_1 = \textsc{RandomChoice}(V_n)$
8 $\hat{f} = (f_0 \setminus \{v_0\}) \cup \{v_1\}$
9 return $\hat{f}$
```

4.5 Clique Negative Sampling (CNS)

Where one extreme NS technique that makes prediction easy for an HLP algorithm is UNS, another extreme is to make it tough, by sampling cliques from the clique-expanded graph G_H of a hypergraph H. This ensures the edge density of sampled hyperedges to always be 1, which, according to our measure of hardness (Definition 1), returns the hardest-to-classify set of non-hyperlinks. However, since clique-finding in a graph is an NP-complete problem, we do not compute them directly. Instead, motivated by the geodesic-distance based NS technique by Lichtenwalter et al. [13], we develop a hypergraph equivalent of their "1-hop" sampling approach via a simple heuristic to efficiently sample non-hyperlinks as per Algorithm 2. Since a hyperlink f (positive pattern) forms a clique in the induced graph G_H of H, this very information could be exploited to sample a non-hyperlink $\hat{f}$ such that $\hat{f}$ too follows f. The exact procedure for CNS has been described in Algorithm 2.

Note that although this heuristic does not guarantee the existence of such common neighbor nodes v_1, we, however, empirically observe that such nodes do exist. Extensions to CNS (*e.g.*, to add/remove multiple nodes at once, *etc.*) could also be implemented. Moreover, by no means does Algorithm 2 sample all possible cliques; it only gives a sample which we use for HLP. CNS ensures all sampled non-hyperlinks to have a unit ED (ref. Fig. 3c), which is much different from SNS (ref. Fig. 3a), where most of them have extremely low ED (if not zero). Hence, **CNS provides the hardest of non-hyperlinks** whereas the hardness of those sampled from MNS lies in the moderate range (ref. Fig. 3b). Non-hyperlinks sampled by CNS gives patterns from the "green" region marked at the top in Fig. 1b.

In summary, there is a whole *spectrum* of NS algorithms that could sample non-hyperlinks, and we have explored four of them, *viz.*, UNS, SNS, MNS, and CNS.

5 Experiments

We take seven different datasets—`email-Enron` (`eE`), `contact-high-school` (`chs`), `contact-primary-school` (`cps`), `tags-math-sx` (`tms`), `MAG-Geo` (`MG`), `coauth-DBLP` (`cD`), `NDC-substances` (`Ns`)—from Benson et al. [3] and perform various HLP experiments on them. Also, we use the same k-core based sampling technique as used by Liben-Nowell et al. [11] to reduce the size of `MAG-Geo` and `coauth-DBLP` datasets since they are huge hypergraphs. More specifically, we retain only those nodes which have *hyperdegree* (number of incident hyperedges) greater than a threshold $k = 16$.

We use five different HLP algorithms: Bayesian Sets (BS) [8], Factorization Machines (FM) [17], Hyper Katz (Katz) [10], Hyper Common Neighbors (CN) [16,21], and Coordinated Matrix Minimization (CMM) [21]. To evaluate an HLP algorithm, we use the area under ROC curve (AUC) metric. In addition, we also report certain statistics on multiple NS techniques, which ultimately gives insights into which technique works best.

All of the datasets used are temporal in nature. We perform a temporal split of $80 : 20$ where hyperedges are sorted according to their timestamp[2] and first 80% of hyperedges are used for training and feature extraction, whereas the remaining 20% are used for testing. The NS ratio (*i.e.*, ratio of negative samples (non-hyperlinks) to positives (hyperlinks)) is fixed to $10 : 1$, except for `NDC-substances` where it is $5 : 1$ (since, on an average, the data has bigger hyperedges). We use the AUC score for the evaluation and comparison of HLP algorithms, since it is a standard metric that has been widely used in the LP literature. We experiment with multiple NS ratios to analyse its impact on the evaluation metric.

6 Results and Discussion

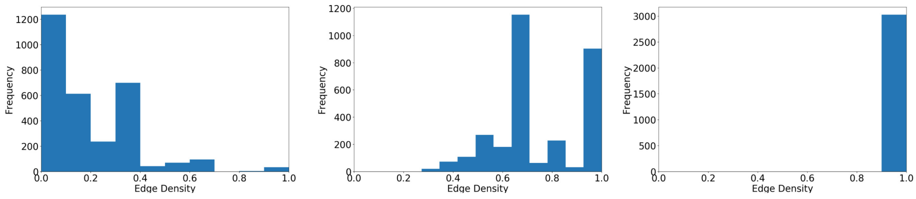
6.1 Hyperlink Prediction Performance

The AUC scores obtained by applying each of the five HLP algorithms on the seven datasets have been populated in Table 1, which has been divided into three parts corresponding to negative sampling techniques SNS, MNS, and CNS. In each row, AUC score for the best performing algorithm has been <u>underlined</u>. But since our main aim is not to compare between HLP algorithms, scores for NS algorithms that give the best performance for a given HLP algorithm has been **bold-faced**. The first observation we make is that except for CMM [21], all other HLP algorithms perform their best when compared against a SNS-sampled negative class. **CMM, which is supposed to be the current state-of-the-art in HLP, performs its best when evaluated against either MNS or CNS based negative sampling**. Another striking point that Table 1 reveals is that **no dataset has a unanimous best performing HLP algorithm, and instead varies with the NS algorithm**. For example, according to SNS,

[2] Multiple timestamps are resolved by using the earliest one.

Table 1. AUC scores (%) for HLP using BS, FM, Katz, CN, and CMM on seven datasets, where NS is performed via SNS, MNS, and CNS. Avg. reduction: SNS $\rightarrow$ MNS: BS = 21%, FM = 12%, Katz = 36%, CN = 43%, CMM = −28% SNS $\rightarrow$ CNS: BS = 35%, FM = 36%, Katz = 44%, CN = 44%, CMM = −33%.

	Sized NS (SNS)					Motif NS (MNS)					Clique NS (CNS)				
	BS	FM	Katz	CN	CMM	BS	FM	Katz	CN	CMM	BS	FM	Katz	CN	CMM
`eE`	**72.7**	**81.8**	**70.1**	**66.0**	55.2	69.3	77.3	29.0	24.3	39.9	37.9	44.8	35.5	27.8	**59.7**
`chs`	**64.6**	**69.9**	**99.4**	**99.2**	57.8	49.9	63.9	77.0	77.4	64.8	47.5	65.8	62.2	66.4	**65.9**
`cps`	**71.1**	**60.2**	**93.9**	**93.4**	49.2	54.1	54.4	67.5	73.1	54.2	49.8	59.6	57.2	62.4	**61.8**
`cD`	**63.9**	**69.9**	**71.2**	**74.9**	38.8	38.3	46.8	16.8	21.9	**61.9**	37.6	38.4	22.1	29.9	54.7
`Ns`	**95.9**	**80.0**	**85.0**	**94.5**	60.6	89.9	75.7	81.2	23.1	**74.2**	73.8	60.4	79.0	58.1	65.7
`tms`	**95.8**	**75.2**	**99.2**	**98.9**	22.8	75.5	64.9	75.7	78.5	53.5	63.9	62.4	51.9	57.3	**59.0**
`MG`	**78.4**	**53.9**	**98.1**	**97.5**	26.7	49.4	50.5	46.5	54.3	**50.2**	41.6	45.7	40.1	51.2	47.1



(a) SNS-sampled non-hyperlinks **(b)** MNS-sampled non-hyperlinks **(c)** CNS-sampled non-hyperlinks

Fig. 3. Edge density distribution (EDD) plots for `email-Enron`.

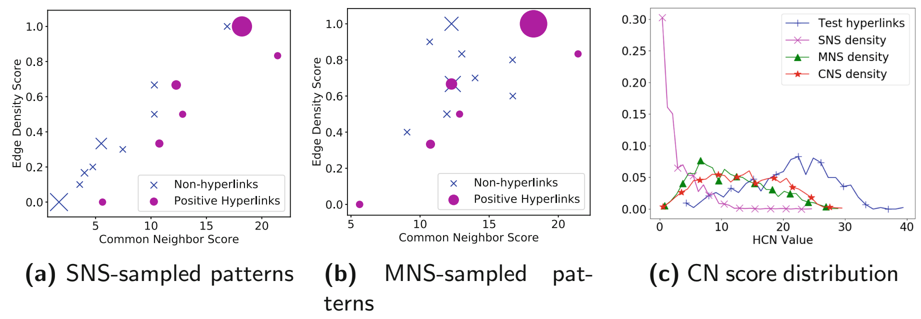
MNS, and CNS, the best algorithm for the `tags-math-sx` (`tms`) dataset turns out to be Katz, CN, and BS respectively. One final point we want to make *w.r.t.* this table is the general trend of reduction in AUC scores as we move from the leftmost block (SNS) to the rightmost one (CNS). The average reduction has been indicated in the table caption, according to which, **simple extensions of link prediction such as CN and Katz have the maximum average reduction (of** $\sim$ **44%)**, and the CMM algorithm, which actually "learns" to *pick* hyperlinks out of a bag of hyperlinks and non-hyperlinks sees an *increment* of 33% as we go from SNS to CNS sampling.

6.2 Edge Density Distribution

We plot the edge-density distributions (EDD) for the `email-Enron` dataset in Fig. 3, wherein EDD for hyperlinks as well as for non-hyperlinks sampled via SNS, MNS, and CNS have been included. For a discussion, see Sect. 4.

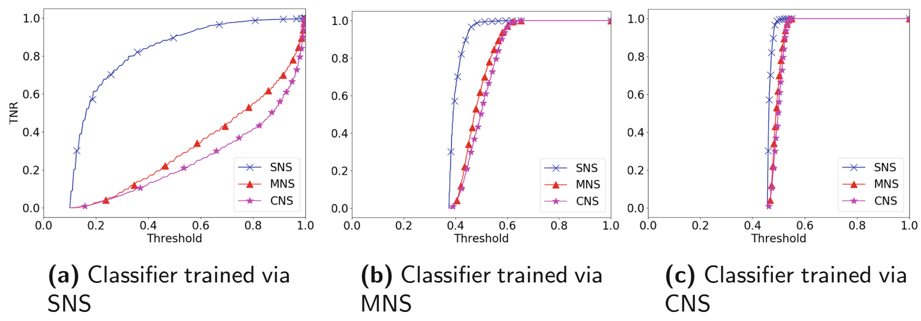
6.3 Common Neighbor vs. Edge Density

Figure 4 shows the scatter plot of common-neighbor (CN) scores and edge-densities (ED) for each test pattern in the `contact-high-school` dataset, where the blue crosses and pink discs represent non-hyperlinks and hyperlinks respectively. It is clear from these plots that while SNS sampled non-hyperlinks have lower ED values and lower CN scores as well, the MNS algorithm samples non-hyperlinks in a way that CN is not able to distinguish between the two classes.



(a) SNS-sampled patterns **(b)** MNS-sampled patterns **(c)** CN score distribution

Fig. 4. (a, b) Common Neighbor (CN) scores vs. Edge Density (ED) for hyperlinks (pink discs) and non-hyperlinks (blue crosses); marker size proportional to frequency. **(c)** CN score distribution of cross-validated test datasets. (All plots for `contact-high-school`.) (Color figure online)



(a) Classifier trained via SNS **(b)** Classifier trained via MNS **(c)** Classifier trained via CNS

Fig. 5. True Negative Rate (TNR) for CN-based Logistic Regression HLP classifiers trained for one NS algorithm and tested on all. (All plots for `contact-high-school`.)

6.4 True Negative Rates for Different Sampling Methods

To better explain the impact of NS algorithms on HLP, we perform HLP via supervised learning, *i.e.*, by using CN scores as a single feature to learn a Logistic Regression classifier (LRC). We perform a cross validation by first preparing three different validation sets, each formed by sampling the negative class by a different NS algorithm. We then train one LRC per NS algorithm (with NS performed via the respective algorithm to generate training data) which is subsequently tested on all three cross validation sets. The performance of LRCs in terms of true negative rates is shown in Fig. 5. An LRC trained using MNS (Fig. 5b) or CNS (Fig. 5c) can easily predict negatives from SNS truly. However, the same is not true for an LRC trained on SNS samples as shown in Fig. 5a. A typical CN score distribution for three different validation sets defined above and positive hyperlinks is shown in Fig. 4c. Cross validation results of LRC models are evident from this distribution as most of the SNS sampled hyperlinks have a low CN score whereas MNS and CNS sampled hyperlinks have CN scores that are comparable with the CN scores positive hyperlinks. Hence, **a model trained on MNS or CNS would be able to generalize better** than that trained on SNS.

7 Conclusions

Under-sampling the majority class in class-imbalanced scenarios is a common practice. But hyperlink prediction (HLP) is atypical, in that there exists extreme class imbalance, with the set of non-hyperlinks being the majority class. We set out to analyze four negative sampling (NS) techniques for HLP, *viz.*, Uniform (UNS), Sized (SNS), Motif (MNS), and Clique (CNS) based NS. We analyzed size, edge-density, and a usual predictor score (CN) distribution for candidate hyperlinks extracted via all NS techniques and found that **while UNS is completely useless for HLP**, **SNS makes the negative class follow the same size distribution as the positive class**. But **MNS and CNS go one step further and focus on matching their edge-density distributions as well, making the HLP problem challenging in nature**. While the evaluation of an HLP algorithm on test sets sampled via SNS, MNS, and CNS is found to vary drastically, a specialized cross-validation of HLP via the supervised learning paradigm further shows that **only MNS and CNS generalize well for HLP**. In essence, **we prescribe using either MNS or CNS** for sampling non-hyperlinks for HLP, since they learn fair and generalized HLP predictors that would perform as expected in practical scenarios.

Acknowledgements. Prasanna Patil was supported by a fellowship grant from the Centre for Networked Intelligence (a Cisco CSR initiative) of the Indian Institute of Science, Bangalore.

References

1. Agarwal, S., Branson, K., Belongie, S.: Higher order learning with graphs. In: Proceedings ICML, ICML 2006, pp. 17–24. ACM, New York (2006)
2. Al Hasan, M., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: Workshop on Link Analysis, Counter-Terrorism and Security, SDM 2006 (2006)
3. Benson, A.R., Abebe, R., Schaub, M.T., Jadbabaie, A., Kleinberg, J.: Simplicial closure and higher-order link prediction. Proc. Natl. Acad. Sci. **115**(48), E11221–E11230 (2018)
4. Bhuiyan, M.A., Rahman, M., Rahman, M., Al Hasan, M.: GUISE: uniform sampling of graphlets for large graph analysis. In: 2012 IEEE ICDM, pp. 91–100, December 2012
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
6. Davis, A., Gardner, B.B., Gardner, M.R.: Deep South: A Social Anthropological Study of Caste and Class. University of Chicago Press, Chicago (1941)
7. Garcia Gasulla, D., Cortés García, C.U., Ayguadé Parra, E., Labarta Mancho, J.J.: Evaluating link prediction on large graphs. In: Artificial Intelligence Research and Development: Proceedings of the 18th International Conference of the Catalan Association for Artificial Intelligence, pp. 90–99. IOS Press (2015)
8. Ghahramani, Z., Heller, K.A.: Bayesian sets. In: Weiss, Y., Schölkopf, B., Platt, J.C. (eds.) Advances in Neural Information Processing Systems, vol. 18, pp. 435–442. MIT Press (2006)
9. Kashtan, N., Itzkovitz, S., Milo, R., Alon, U.: Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. Bioinformatics **20**(11), 1746–1758 (2004)
10. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
11. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: CIKM 2003, pp. 556–559. ACM, New York (2003)
12. Lichtenwalter, R.N., Chawla, N.V.: Link prediction: fair and effective evaluation. In: Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2012, pp. 376–383. IEEE Computer Society (2012)
13. Lichtenwalter, R.N., Lussier, J.T., Chawla, N.V.: New perspectives and methods in link prediction. In: 16th ACM SIGKDD, pp. 243–252. ACM (2010)
14. Lü, L., Zhou, T.: Link prediction in complex networks: a survey. Physica A **390**(6), 1150–1170 (2011)
15. Martínez, V., Berzal, F., Cubero, J.C.: A survey of link prediction in complex networks. ACM Comput. Surv. **49**(4), 69:1–69:33 (2016)
16. Newman, M.E.: Clustering and preferential attachment in growing networks. Phys. Rev. E **64**(2), 025102 (2001)
17. Rendle, S.: Factorization machines with libFM. ACM Trans. Intell. Syst. Technol. **3**(3), 57:1–57:22 (2012)
18. Wang, T., Liao, G.: A review of link prediction in social networks. In: 2014 International Conference on Management of e-Commerce and e-Government, ICMeCG, pp. 147–150. IEEE (2014)

19. Xu, Y., Rockmore, D., Kleinbaum, A.M.: Hyperlink prediction in hypernetworks using latent social features. In: Fürnkranz, J., Hüllermeier, E., Higuchi, T. (eds.) DS 2013. LNCS (LNAI), vol. 8140, pp. 324–339. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40897-7_22
20. Yang, Y., Lichtenwalter, R.N., Chawla, N.V.: Evaluating link prediction methods. Knowl. Inf. Syst. **45**(3), 751–782 (2014). https://doi.org/10.1007/s10115-014-0789-0
21. Zhang, M., Cui, Z., Jiang, S., Chen, Y.: Beyond link prediction: predicting hyperlinks in adjacency space. In: Proceedings AAAI 2018. AAAI (2018)
22. Zhang, M., Cui, Z., Oyetunde, T., Tang, Y., Chen, Y.: Recovering metabolic networks using a novel hyperlink prediction method. arXiv preprint arXiv:1610.06941 (2016)
23. Zhou, D., Huang, J., Schölkopf, B.: Learning with hypergraphs: clustering, classification, and embedding. In: Proceedings of the 19th International Conference on Neural Information Processing Systems, pp. 1601–1608 (2006)