International Journal of Research in Engineering, Science and Management
Volume 8, Issue 10, October 2025
https://www.ijresm.com | ISSN (Online): 2581-5792

Link Prediction and Cascading within Subreddits

Sunjeet Jena[1], Aayush Sharma[2], Oluwaseun Talabi[3], Ankit Sharma[4], Akhil Mittal[5], Rajath Manjunath[6*]
[1,2,3,4,5]*Arizona State University, Arizona, USA*
[6]*Business Process Quality Analyst, Infosys BPM Quality, Bengaluru, India*

***Abstract*: Reddit is a social news aggregation and discussion website, where members post content like text, posts, images etc. Posts are organized by subject boards called "communities" or "Subreddits", which cover various topics such as news, politics, science, movies, video games etc. "Subreddits" become a channel of propagation of information when a post at one subreddit is linked or tagged in another subreddit. We formulate this network as a Graph, where each node is a "Subreddit" and edges are existing hyperlinks between them. Given a pair of nodes ("Subreddits") in that graph, we wish to predict the probability that of one subreddit hyperlinking or tagging a post in the other subreddit.**

***Keywords*: Link Prediction, Graph, Reddit.**

1. Introduction

Reddit is a social news aggregation and discussion website, where registered members share content like text, posts, images etc. which are then shared, discussed, upvoted or downvoted by other registered users. Reddit was founded by Steve Huffmank, Alexis Ohanian and Aaron Swartz, in 2005. Conde´ Nast Publications acquired the website in October 2006 and as of September 2021, Reddit was ranked as the 19th most- visited website in the world and 7th most-visited website in the U.S. [12]

Posts in the Reddit are organized by areas of interest or subject boards called "communities" or "Subreddits", which cover various topics such as news, politics, science, movies, video games etc. There are about 140,000 active Subreddits as of July, 2018 [12]. "Subreddits" become a channel of propagation of information when a post at one subreddit is linked or tagged in another subreddit. These hyperlinking of posts lead to various degrees of discussion sometimes leading to conflicts (disagreeing on the original content). Over the entire Reddit website, these hyperlinks between the subreddits form a network. Figure 1 shows a sample network created on 20 subreddits. The direction of the hyperlink indicates the source subreddit hyperlinking the target subreddit(original post). For example, subreddits like 'globaloffensive', 'worldnews' and 'pcmasterrace' have all individually hyperlinked a post from the 'video' subreddit.

Over a large set of such connections, this network develops into a *Complex Network*, which makes it a substantially difficult and interesting problem to study and experiment with. From here onwords we shall refer this network as *"Subreddit-Sphere"*. We formulate this *complex network* as a graph, where each node is a "Subreddit" and edges are the existing hyperlinks between them. Given a pair of nodes ("Subreddits") in that graph, we wish to predict the probability that of one subreddit hyperlinking or tagging a post in the other subreddit or shall hyperlink or tag a post in the *near* future.

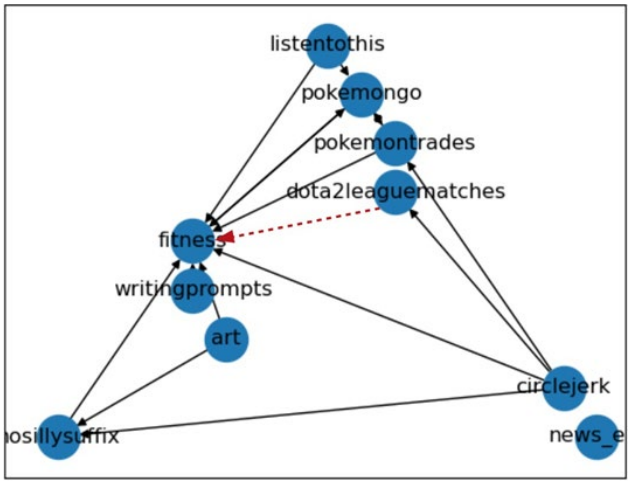

Fig. 1. The red dotted directed edge is the possible connection between two subreddits and we wish to predict the probability of such a link

For this work we consider the Stanford Subreddit Hyperlink Network dataset [5] which contains 55,863 number of nodes (subreddits) and 858,490 number of edges (hyperlink between subreddits). This dataset perfectly fits the objective of our project and contains enough data to generate training and test samples. Section 4 gives a more detailed explanation about the dataset and the data processing we implemented for the project.

2. Problem Formulation

As mentioned in the earlier section, we consider the *Subreddit-Sphere* network as a directed graph, with each node representing a "Subreddit" and the edges among them as the existing hyperlinks between them. Consider a pair of nodes, between which there might already exist a hyperlink connection or may exist in future, that is one "Subreddit" (node) hyperlinking a post in another "Subreddit" (node). We design a simple yet effective algorithm that generates the probability of such a link (edge) between them.

Consider figure 2 for example. Subreddits such as "pokemongo" and "pokemontrades" have hyperlinked at least one

*Corresponding author: rajathm64@gmail.com

post from the "fitness" subreddit. Given this sub-graph we wish to predict the probability of a link(edge) between the "dota2leaguematches" subreddit and the "fitness" subreddit. More specifically we wish to predict if "dota2leaguematches" subreddit hyperlinks or may hyperlink in future a post in the "fitness" subreddit.

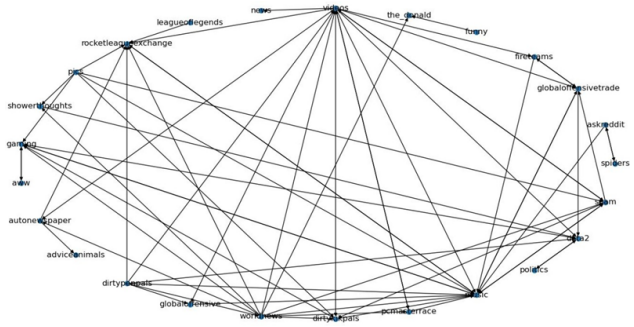

Fig. 2. Sample Network Generated from the first 20 Subreddits and the hyperlinks between them from Stanford Reddit Hyperlink Network dataset

We consider this problem as a binary classification problem with label '0' indicating no-existence or no possible existence of hyperlinks between two subreddits and label '1' as an existing or future existence of hyperlinks between the two subreddits.

From here on wards, we shall use the words "node" and "subreddit" interchangeably in various sections referring to the same entity. Similarly, the words "link", "hyperlink", "edge" and their corresponding plural forms shall be used interchangeably referring to the connection between two nodes.

3. Related Works

With the advances in complex networks research, link prediction in graphs have long been studied with potential applications in systems biology [10], [11], social media [6], [8], supply chain [1], recommendation systems [9] etc. For the scope of the project work, we limit our discussion to three different works [2], [5], [8], as these works are very similar to our objective in the project.

Leskovec et al. [8] studied the cascading pattern of the graphs in the blog-to-blog network. Their work involved around generating a probability with which information is shared between two blogs. Blogs are sort of personal websites, where people can post articles, opinions etc. These posts are then cited or hyperlinked by other blogs, and then the blogs which hyperlinks the original posts are then again hyperlinked by some other blogs, thus forming cascades among the nodes (blogs). Leskovec et al. *empirically* studied and observed the probability over which one blog hyperlink a post from another blogs. To verify their observations, they used this probability to regenerate cascades in the original dataset and observed if similar cascading patterns were observed or not. Although they were successful in replicating a majority of the cascading structures, the probability of propagation was constant irrespective of the node features and links, thus missing out on a large portion of possible links between any pair of nodes.

Kumar et al. [5] studied inter-community interactions in the subreddits and did sentiment analysis on the discussion involving the hyperlinked posts. More specifically, given a possible hyperlink connection or an existing connection between the two subreddits, they provided an algorithm to predict the probability of a conflict between the subreddits. They used an LSTM Architecture for predicting this probability. Although, their work did implicitly involve a portion of link prediction between two subreddits, they explicitly did not provide details on how such a link occurs or what are the features over which such links exists.

Grover et al. [2] gave an algorithmic framework for learning continuous feature representations for nodes in networks. They designed an algorithm to learn a mapping of nodes to a low-dimensional space of features, which then can be used to teach a function approximation method such as deep learning etc. to predict the probability of a link between two nodes. Although, this framework is a very effective method of generating features which maximizes the likelihood of preserving network neighborhoods of nodes, this method is computationally expensive and time-consuming.

For this project we propose a link prediction framework in the "Subreddit-Sphere", that utilizes 20 predefined node features that preserves neighbourhood information and use a gradient boosting based decision tree method to predict the probability of the existence of a link between two nodes. Our method is light, fast and highly accurate over the entire dataset.

4. Dataset and Data Preprocessing

A. Dataset

For our work we use the Stanford Reddit Hyperlink Network dataset [5]. This subreddit-to-subreddit hyperlink network is generated from the posts that create hyperlinks from one subreddit to another. The source community is where the hyperlink originates from a post and links to post in the target community. Each hyperlink or edge is annotated with three properties: timestamp, sentiment of the source community post towards the target community post, and the text property vector of the source post. The edges of the network is directed, signed, temporal, and attributed.

The dataset contains three sub-datasets: a) Network of subreddit-to-subreddit hyperlinks extracted from hyperlinks in the body of the post. b) Network of subreddit-to-subreddit hyperlinks extracted from hyperlinks in the title of the post. c) Subreddit embeddings: Embedding vectors representing each subreddit. The embedding vectors representing subreddits are 300 dimensional vectors [5], [7].

B. Data Preprocessing

For the purpose of our project we combined the network of hyperlinks extracted from hyperlinks in the body of the post and hyperlinks extracted from title of the post into one dataset/network. Of all the edge attributes mentioned in the previous section, we only keep the source and the target label attributes. To reiterate, the source is the node where the hyperlink originates and the target is the node where the hyperlink ends. The direction of the edge is from the source to

the target.

The vector embeddings of the subreddits from the *Subreddit Embeddings* were also normalized, using the standard normalization methods. Table 1 shows sample links in the graph.

Table 1
Sample links in the graph

	Source Node	Target Node
0	rddtgaming	rddtrust
1	xboxone	battlefield4
3	ps4	battlefield4
4	fitnesscircle	leangains
5	fitnesscircle	leangains

C. Graph Generation

Once we have preprocessed the dataset, the sample pairs as shown in Table I can be used to generate the graph representing the entire ”Subreddit-Sphere”. We call the original graph as the *Validation Graph* and we model this graph and all the subsequent graphs using the NetworkX library [3].

Using the *Validation Graph* we generated two additional graphs, *Test Graph* and *Training Graph*. The *Test Graph* contains 90% of the all the links/edges of the *Validation Graph* and the *Training Graph* contains all 70% of all the link/edges of the *Test Graph*. Figure 3 shows sample *Validation, Test and Training Graphs*.

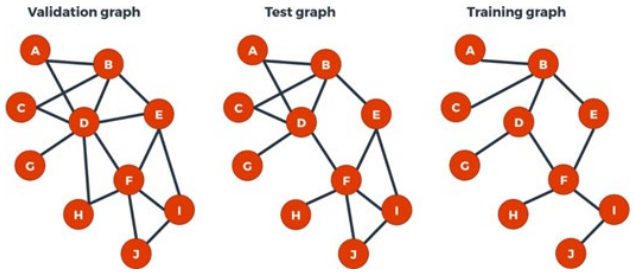

Fig. 3. Sample validation, test and training graphs

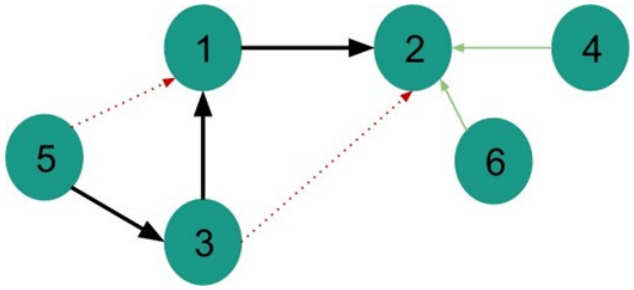

Fig. 4. The red dotted line represents a possible direct edge between the pair of nodes. The hopping distance between 3 and 2 is 2

We would like to explicitly present the fact that the number of nodes in all the three graphs defined in the earlier paragraph remains constant and it is the edges/links that vary across the three graphs. In overall consideration, the *Test graph* is a subgraph of the *Validation graph* and the *Training Graph* is a subgraph of the *Test Graph*.

In figure 3 it can been that the edge connection present in between D and H in the *Validation Graph*is absent in the *Test Graph* and the edge connection present between A and D in the *Test Graph* is absent in the *Training Graph*. These missing links in the *Training and Test Graphs* constitute the original data samples with label ‘1’.

D. Dataset Creation

The test dataset to verify the effectiveness of the algorithm is generated from the *Test Graph* and the training dataset to train the model is generated from the *Training Graph*. As discussed in the previous sections, we consider this problem as the binary classification problem, with two labels *0* and *1*.

We define a term called as *Hopping Distance*, which is the shortest path between two nodes other than the direct edge between them (if any). For example consider figure 4. In this figure, the hopping distance between the node 3 and the node 2 is 2 and the hopping distance between 5 and 1 is also 2. The dotted red line represents there may or may not be a direct edge between those two nodes.

We consider the base case of sample pair with a hopping distance of 2 for our work. To create the test data, we consider all the node pairs in the *Test Graph* with a hopping distance of 2 between them and to create the training data, we consider all the node pairs in the *Training Graph* with a hopping distance of 2 between them. The pair nodes extracted from the above method and containing a direct edge between them is labeled as ‘1’ and the the pair of nodes without a directed edge is labeled as ‘0’. Using the above the directive on both the *Test* and *Training Graphs*, we obtain the test dataset and the training dataset.

Fig. 5. Pair of nodes represented using a 20-dimensional feature vector

We observed that the above method generated a highly imbalanced dataset. Consider the training set, which was generated from the *Training Graph*. It contains 6903 pairs of node which have a label of ‘1’ and 5359669 pairs which have a label of ‘0’. To mitigate this issue, we removed the sample pairs with label ‘0’ with a probability 0.8. Following the procedure, we obtained the final dataset which contained 6903 node pairs which have a label of 1 and 1070907 node pairs which have a label of 0.

5. Feature Engineering

Having generated the dataset for training and testing the model, we need to represent the node pairs in the vector space. More specifically we need to embed these pair of nodes in the embedded vector space, which then can be used by the model as a set of features of the pair of nodes, to do the hypothesis testing. Although word2vec algorithm [2] is an excellent algorithm to generate node features and the node pairs into the vector space, it is slow, time consuming and does not give an intuitive explanation as to why two neighbouring node are distinctively so different to each other. To obtain a more intuitive understanding of the sample pairs features we consider 20 pre-defined individual metrics on each node-pairs. These 20 features represent the node pair in the embedded vector space

and is taken as an input by our algorithm. See figure 5.

We consider 20 different individual metrics to construct the embedded vector representation of the pair. The individual metrics are as follows:

1) *Cosine similarity between subreddit embeddings*: Each embedding is real-valued 300-dimensional vector, and is generated from the user-to-subreddit posting network using a word2vec-style objective function. The idea of this method is based on the dot product of the embeddings of the two subreddit in the sample pair. It is often used to compare documents in text mining.
2) *Common neighbours*: It is defined as the number of shared neighbors between both nodes.
3) *Jaccard Coefficient*: It measures the ratio of shared neighbors in the complete set of neighbors for two nodes.
4) *Adamic Adar index* : It measures the similarity between two entities based on their shared features. However, each feature weight is logarithmically penalized by its appearance frequency.
5) *Preferential Attachment* :Based on the observation that the probability of link formation between two nodes increases as the degree of these nodes does.
6) *Sørensen Index* : Despite its similarity with the Jaccard index, it is less sensitive to outliers. This index is designed for comparing the similarity of two samples and originally used in analysis plant sociology.
7) *Hub Promoted Index* : The main goal of this similarity measure is to avoid link formation between hub nodes and promote link formation between low-degree nodes and hubs. The property of this index is that the links adjacent to hubs are likely to obtain a higher similarity score.
8) *Hub Depressed Index (HDI)*: The hub depressed index promotes link formation between hubs and between low- degree nodes, but not between hubs and low-degree nodes. Approach that uses the idea of hub in totally different manner than HPI is Hub Depressed Index (HDI). It gives links adjacent to hub a lower score.
9) *Salton Index*: The Salton index yields a value that is approximately twice the Jaccard index. Salton cosine is common cosine metric that is used to compute similarity between pair of nodes.
10) *Resource Allocation Index*: Resource allocation punishes the higher degree nodes more heavily. This similarity measure performs better for the network which has high average degrees. It uses not only neighbors but also neighbor of neighbors.
11) *Deductive Metric*: It follows deductive reasoning and is supported by the generalisations of a node. The metric gives us the likelihood of edge x → y. This process is a top-down one, for estimating edge likelihood.
12) *Inductive Metric*: It follows inductive reasoning and is supported by the specialisations of a node. The metric gives us the likelihood of edge x → y. This process is a bottom-top one, for estimating edge likelihood.
13) *Deductive Log Metric*: Modified Deductive metric with logarithmic term.
14) *Inductive Log Metric*: Modified Inductive metric with logarithmic term.
15) *INF score*: Combine both deductive and inductive into a single score. This aggregates the evidence provided by both the ancestors and the descendants of a vertex in order to determine the likelihood with which edges originating from that vertex exist.
16) *INF_2D score*: Deductive score typically achieves higher precision than IND. The combination of Deductive and Inductive outperforms Deductive alone. This is modification of INF as INF_2D where Deductive score is given twice the weight of the Inductive score.
17) *INF_2I score*: The combination of DED and IND out-performs Deductive alone. This is modification of INF as INF_2I where INF score is given twice the weight of the IND score.
18) *INF_LOG Score*: A modification of INF in which the amount of satisfying ancestors and descendants, not only their proportion, is taken into account.
19) *INF_LOG_2D Score*: Combining INF_LOG and INF _2D to build INF_LOG_2D, which addresses issues of 1)proportion of satisfying ancestors/descendants 2) low precision in normal INF score.
20) *INF_LOG_2I Score*: Combining INF_LOG and INF_ 2D to build INF_LOG_2I, where IND_LOG score is given twice the weight of the DED_LOG score.

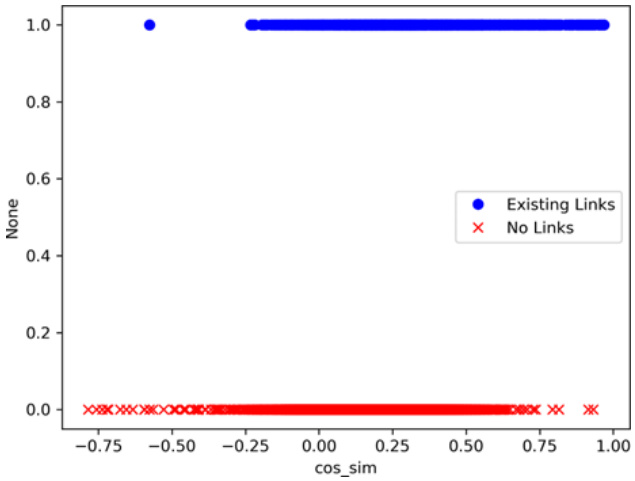

Fig. 6. Cosine similarity

6. Visualizing the Features

In the previous section we formally defined the individual metrics which has been used to construct the 20 dimensional feature vector. In this section we provide a visualization of these features on 1D, 2D and 3D graphs on the entire validation set to provide an intuitive explanation as to why these features preserve the necessary information for the link prediction algorithm.

A. 1-D Features

In this sub-section we consider each of the metrics individually and plot them on a 1-D graph.

Figure 6 to figure 10 are the plots visualizing 5 individual metrics-Cosine Similarity, Adamic Adar Score, Common Neighbour, Jaccard Coefficient and Salton Index. The 'blue circles' are the sample pairs which have at least one edge between them and the 'red cross' are the sample pairs of nodes that do not have any edge between them. Please note that the 'blue circles' representing the sample pairs with existing link have been plotted a unit length above the *x-axis* to allow the readers to see both the clusters.

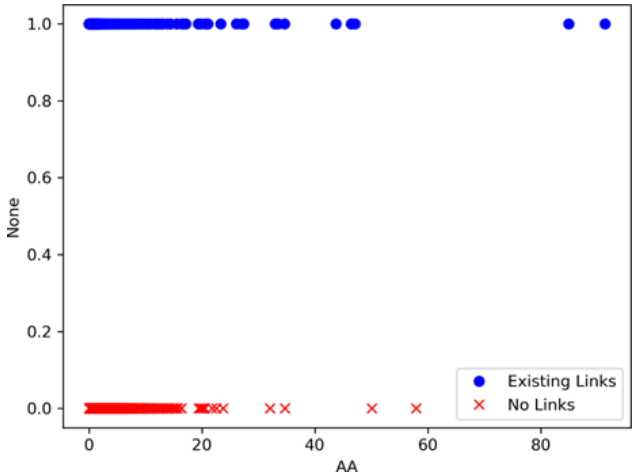

Fig. 7. Adamic Adar score

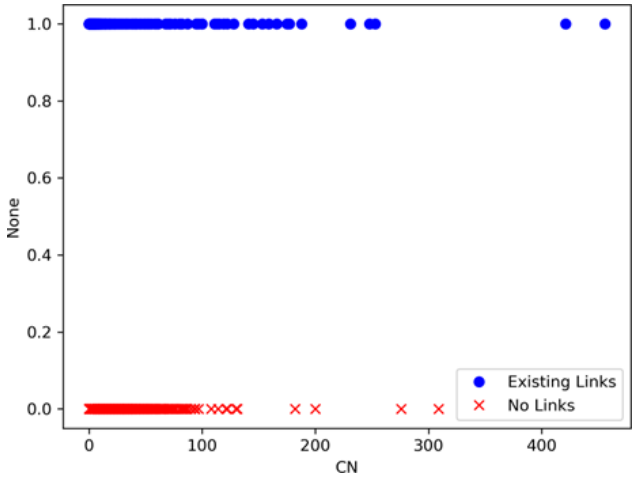

Fig. 8. Common neighbour

It is very much evident that, utilizing these feature points individually won't allow the model to create a good decision boundary as the cluster are practically not separable from each others.

B. 2-D Features

This section covers the plots when 2 features are combined. Figure 11 to figure 14 shows these plots - (Deductive Metric vs Cosine Similarity), (Salton Index vs Jaccard Coefficient), (Hub Promoted Index vs Sorensen Index) and (Sorensen Index vs Salton Index).

It is evident from the plots that these clusters are visually more separable than the one in the individual feature graphs in the previous sub-section.

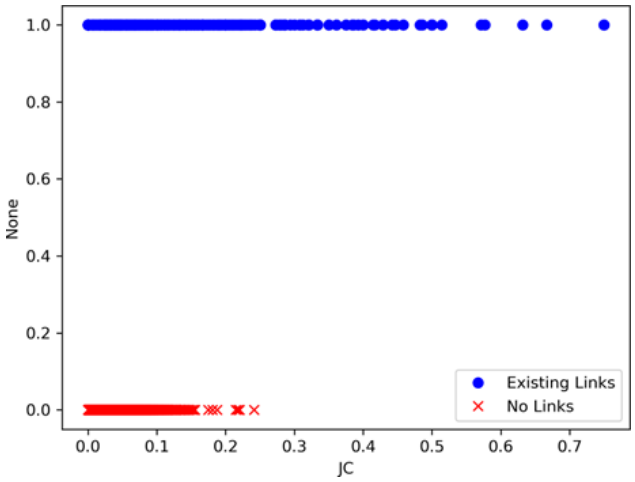

Fig. 9. Jaccard coefficient

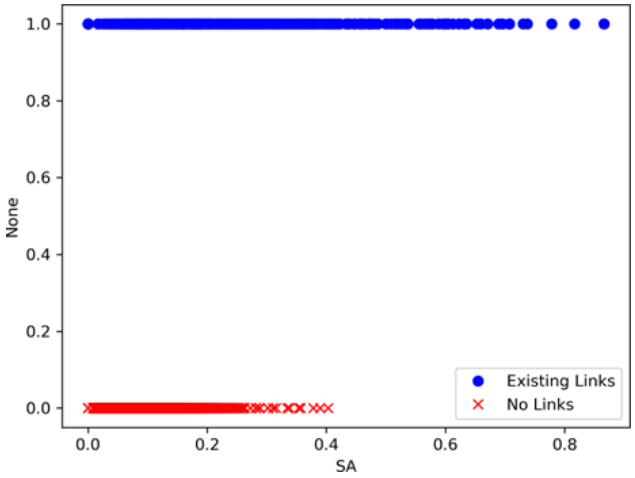

Fig. 10. Salton index

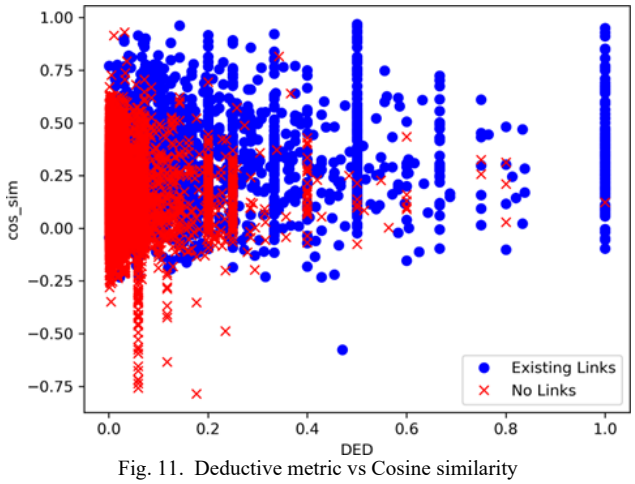

Fig. 11. Deductive metric vs Cosine similarity

C. 3-D Features

This section covers the 3D plots which visualizes data points when three different features are combined. Figure 15 to figure 17 are the 3D plots - (Cosine Similarity vs 2x(Inductive + Deductive) vs Hub Promoted Index), (Cosine Similarity vs 2x(Inductive Score) vs Resource Allocation Index) and (Cosine

Similarity vs Jaccard Coefficient vs Hub Promoted Index).

The plots seen in this sub-section can be seen to have clear decision boundary and this gives an intuitive understanding as to why combining these individual features shall help the model generate a good decision boundary between the data points for this classification problem.

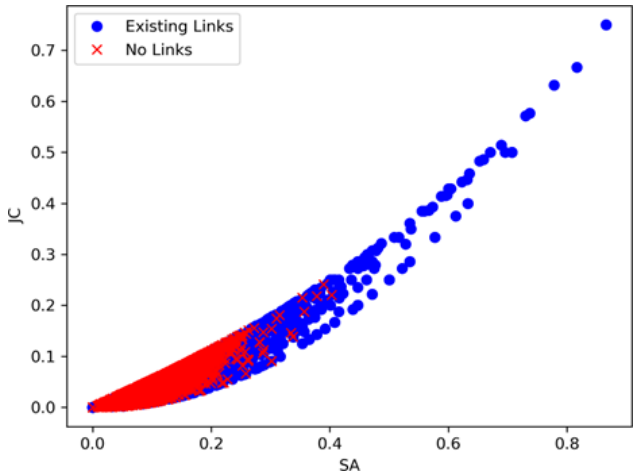

Fig. 12. Salton index vs Jaccard coefficient

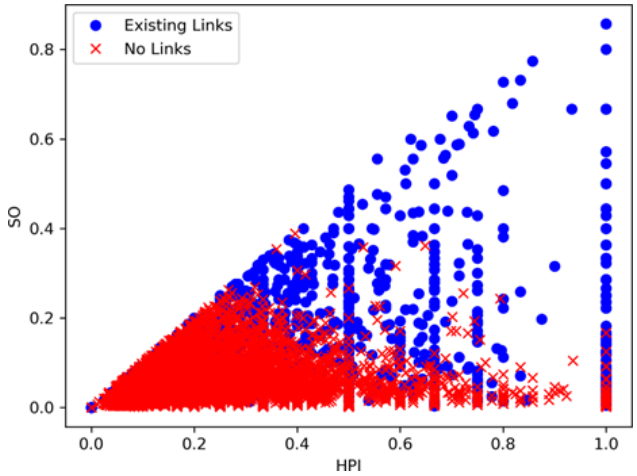

Fig. 13. Hub promoted index vs Sorensen index

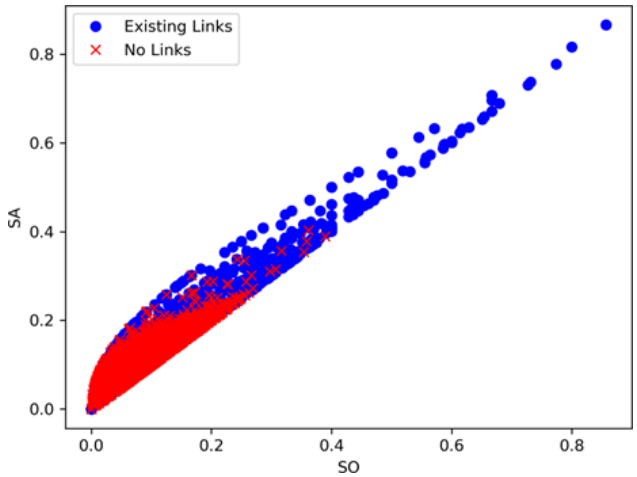

Fig. 14. Sorensen index vs Salton index

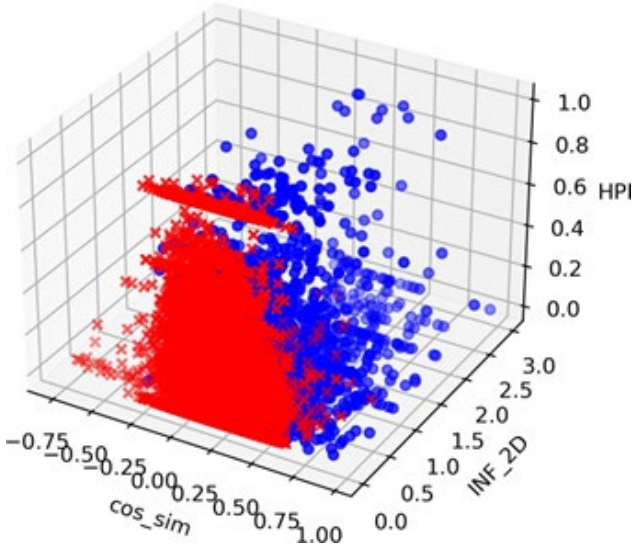

Fig. 15. Cosine similarity vs 2x(Ind+Ded) vs Hub promoted index

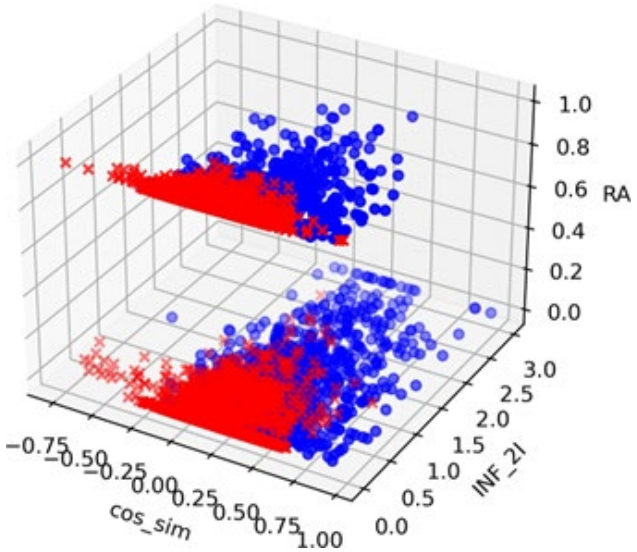

Fig. 16. Cosine similarity vs 2x(Ind) vs Resource allocation index

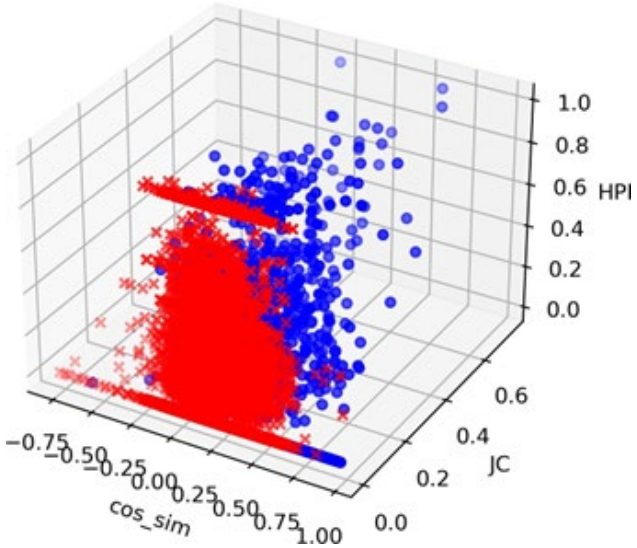

Fig. 17. Cosine similarity vs Jaccard coefficient vs Hub promoted index

7. Experimental Settings

Once we have generated the 20 dimensional feature vector for all the sample pairs in both the test and the validation set, we can use any standard machine learning models to learn the decision boundary for this binary classification problem. For our work we make use of Light Gradient Boosting Method(LGBM) for this problem [4]. Light GBM is a gradient boosting framework that uses tree based learning algorithm. We make use of LGBM for three reasons: 1) LGBM is pretty fast. Both training and inference runs were completed in few minutes in our case. 2) It can easily handle large dataset without much resources. 3) The algorithm unlike deep learning methods and node2vec [2] needs low memory to run.

8. Results

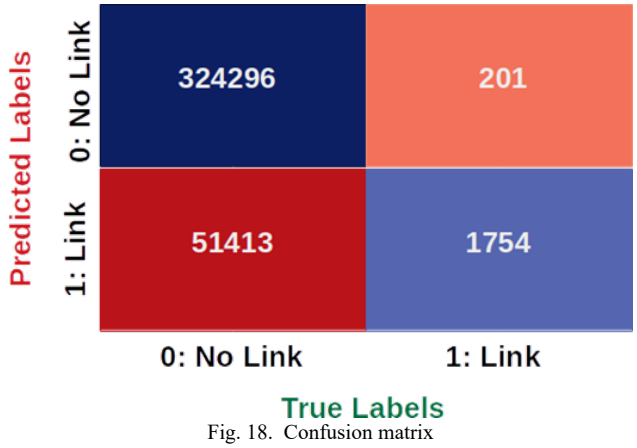

Fig. 18. Confusion matrix

After training the LGBM model we achieved an AUC score of 0.87 on the test set. Figure 18 shows the confusion matrix generated on the predictions made by the the trained LGBM model on the test set. We set a threshold of 0.5 to classify the predicted outputs as '1' (link exists) or '0' (no link) between the pair of nodes. It can been seen that most of the errors originates from the *False Positives*, that is when there is no existing link between the nodes but the models predicts as having one. Please note that we consider all the samples pairs in the test set, which includes sample pairs with and without missing link with a hopping distance of two.

9. Conclusion

We designed a method to predict the existence of a link between two subreddits. We constructed a feature vector representing the sample node pairs by combining 20 individual metrics such as cosine similarity, inductive score etc. These feature vector represents the node pairs in the embedded vector space. The Light Gradient Boosting Method model takes these feature vectors as inputs and generates the probability of the existing connection between those two nodes. We achieve an AUC score of 0.87 on the test set. All in all, we were successfully able to model the problem of link prediction with a *Hopping Distance* of 2 and our main contributions are as follows:

1) Designed a Feature Vector Based on Classical Link Prediction Algorithms.
2) Used Gradient Boosting Method for Link Prediction in the Graphs.
3) Provided Empirical Results to show good results can be achieved in link prediction problems without the use of more sophisticated node-feature extraction methods such as node2vec.

10. Future Work

For the future work we wish to experiment with higher *Hopping Distance* like 3,4 5 etc. and evaluate the algorithm's effectiveness in such cases. We expect that with greater hopping distances, many of the individual features may not be effective enough to model the network to predict the link between the nodes.

References

[1] Alexandra Brintrup, Pascal Wichmann, Philip Woodall, Duncan Mcfarlane, E Nicks, and W Krechel. Predicting hidden links in supply networks. 01 2018.

[2] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[3] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[4] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[5] Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. Community interaction and conflict on the web. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 933–943. International World Wide Web Conferences Steering Committee, 2018.

[6] Srijan Kumar, William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Community interaction and conflict on the web. *CoRR*, abs/1803.03697, 2018.

[7] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1269–1278. ACM, 2019.

[8] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie Glance, and Matthew Hurst. Cascading behavior in large blog graphs, 2007.

[9] Jing Li, Lingling Zhang, Fan Meng, and Fenhua Li. Recommendation algorithm based on link prediction and domain knowledge in retail transactions. *Procedia Computer Science*, 31:875–881, 2014. 2nd International Conference on Information Technology and Quantitative Management, ITQM 2014.

[10] Giulia Muzio, Leslie O'Bray, and Karsten Borgwardt. Biological network analysis with deep learning. *Briefings in Bioinformatics*, 22(2):1515–1530, 2020.

[11] Fei Tan, Yongxiang Xia, and Boyao Zhu. Link prediction in complex networks: A mutual information perspective. *PLOS ONE*, 9(9):1–8, 2014.

[12] Wikipedia contributors. Reddit — Wikipedia, the free encyclopedia, 2021.