Link prediction challenge

Kanvaly FADIGA[a,c,1] **and Mohamed TRAORE (Team: Warrior-x)**[b,1,2]

[a]Ecole polytechnique; [b]Ecole polytechnique

For this semester we study a lot of machine learning technics and the challenge was a good opportunity to pratice. Her the subjet was link prediction, whiwh aim is predict new connections based on existing ones in the network. They give a training graph with 425737 edges and 33750 nodes. Our approches where to explore features extration methods using handcraft features or deep learning. Handcraft 33 extrated features guve the best f1-score and the classification where made using Randoms Forest

Link prediction | Machine Learning | Graph | ...

1. Exploratory Data Analysis (EDA)

The first step of all machine learning start we EDA. So we analyse the graph provided to us to understand some propoerty about it. First, we look at indegrees and we observe that most of the node have few incoming edge while few are like hubs receving many incoming edges.

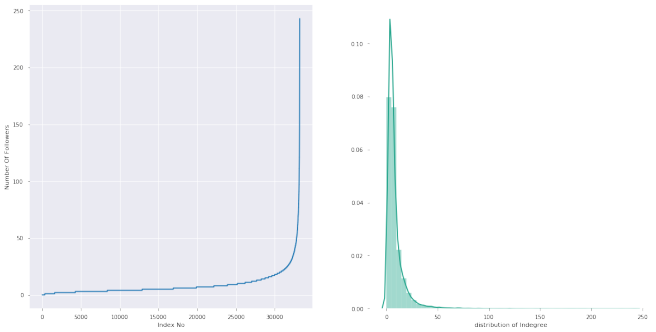

Fig. 1. Indegree distribution. (Left) sorted indegree of nodes, (Right) Density of indegree

This observation is more significant in the outdegree, where 50% have 0 outgoing egde. We have 13 with more than 3000 with the big one having 10k outgoing nodes.

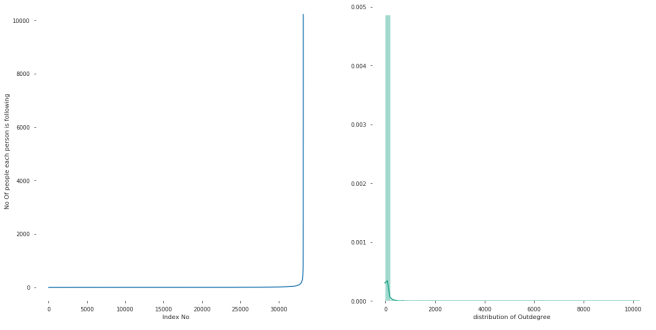

Fig. 2. Outdegree distribution. (Left) sorted outdegree of nodes, (Right) Density of outdegree

Seeing this behevior we need to be carefull on how we choose features because here we have many nodes with zero outdegree and low indegree.

2. Features Engineering

There two way of getting features from a graph. The first one is by extrated features using similarity between nodes on the very simple idea that if two nodes are more similar, they are more likely to be linked. Based on this hypothesis similarity between unconnected node pairs (x and y) are assigned s (x, y) and ranked; not-yet-existing links can be predicted with from these score. Secondly, we can instead learning the nodes representation using embeddings. These learning methods can be viewed as graph dimensionality reduction techniques which map the graph structure based on features best describing the relations between the nodes and embed the nodes to a low dimensional feature space. Embedding algorithms try to preserve the structure of the embedded graph in the vector space by keeping the neighboring nodes closer to each other.

Feature Extraction Methods. It is the one we choose at the end. We extract local and global similarity to represent each nodes. We end up with 43 features representing the edge. Inspired from David Liben-nowell (2004), Who talk about like prediction as estimating the frequency for *unseen bigram (n-gram for two entities)* in language modeling, we found Cukierski et al. (2011) which proposed an approximation to the method. A similarity score was then defined as three separate problems:

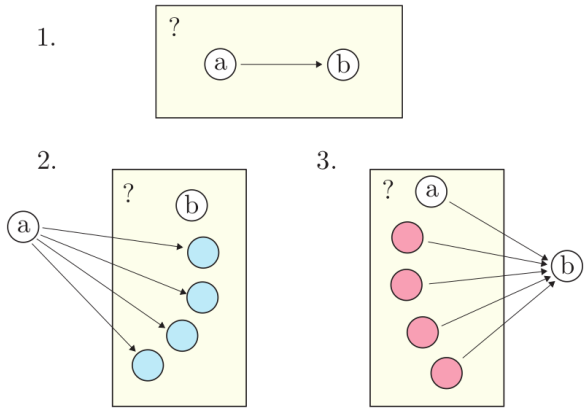

Fig. 3. Node similarity can be defined not just between v_a and v_b , but also between the appropriate neighbors of both nodes. Cukierski et al. (2011)

1) How similar is node v_a to v_b ?

$$score(v_a, v_b)$$

2) How similar are the outbound neighbors of v_a to v_b ?

$$score(v_a, v_b) = \frac{1}{|\Gamma_{out}(v_a)|} \sum_{v \in \Gamma_{out}(v_a)} score(v_b, v)$$

3) How similar are the inbound neighbors of v_a to v_b ?

$$score(v_a, v_b) = \frac{1}{|\Gamma_{in}(v_b)|} \sum_{v \in \Gamma_{in}(v_b)} score(v_a, v)$$

So we code some of our similarity measure using this method. Then another problem we face while computing was time of computation. using the whole graph to compute some features take too long. The solution we use is to compute them using subgraph formed with first level neighbors of both nodes as suggest by Cukierski et al. (2011).

Feature Learning Methods. Representation learning of nodes in a graph refers to dimensionality reduction techniques where nodes are embedded in a continuous space and have dense representations recent node embedding methods have been focused on undirected graphs with limited attention to the directed setting. We try many of them here are the more relevant. The first is random walk based like node2vec, DeepWalk. These one have a lot of hyperparameter and we don't know how to chose them. the second shortcoming is that it represents an edge (u, v) identically to reverse counterpart (v, u), and is unable to capture asymmetric relationships. To overcome this we found Abu-El-Haija et al. (2017) which propose a method for embedding graphs while preserving directed edge information. Using embedding from node2vec, he project source node and destination node on two different space. Another asymetry preserving approache is HOPE Ou et al. (2016) which preserves the asymmetric role information of the nodes by approximating high-order proximity measures like Katz measure, Rooted PageRank etc. The one we choose here was VERtex Similarity Embeddings (VERSE) Tsitsulin et al. (2018), a simple, versatile, and memory-efficient method that derives graph embeddings explicitly calibrated to preserve the distributions of a selected vertex-to-vertex similarity measure7.

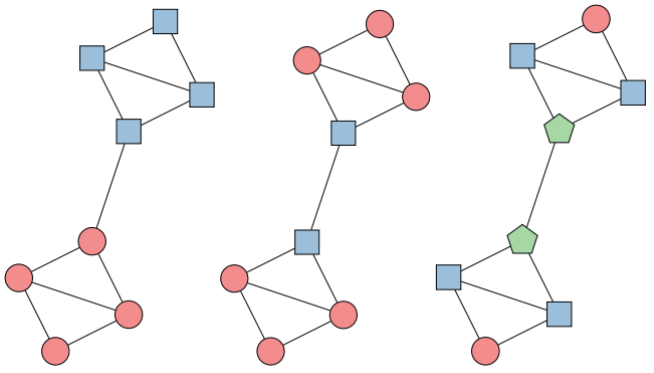

Fig. 4. Figure 1: Three node properties are highlighted on the same graph that verse capture

Node text features. Here we want to generate similarity score between nodes text document. Our approches is the following one. first, we generate a bag-of-word of a document and we eliminate irrelavant word using whose tf-idf weighting score is low. then, we embed each word using word2vec. we sample one million of word over all document and apply kmeans to get 1000 centroid. So each document will be represent by histogram of the 1000 word. the comparison of two documents is made by cosine product of the two histogram.

Latent Dirichlet Allocation (LDA). is unsupervised Learning a technique of dimension reduction. In our case, we choose 20 topics for all documents. The algorithm returns the representation rate of each topic in the selected text. So we have a vector of dimension 20 per document.

	Topic0	Topic1	Topic2	Topic3	Topic4	Topic5	Topic6	Topic7	Topic8	Topic9	Topic10	Topic11	Topic12	Topic13	Topic14	Topic15	Topic16	Topic17	Topic18	Topic19
Doc0	0	0	0	0	0	0.98	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Doc1	0	0	0.02	0.08	0	0	0.05	0.03	0.23	0	0.23	0	0	0.33	0	0	0	0	0.03	0
Doc2	0	0	0.08	0.05	0	0	0.38	0	0	0	0	0.03	0	0	0	0	0.01	0	0.28	0.18
Doc3	0	0	0	0.34	0.11	0	0	0	0	0	0	0.02	0	0.4	0	0	0	0.06	0.07	0
Doc4	0	0	0	0.34	0.06	0	0.03	0	0.09	0	0.28	0	0.1	0.09	0	0	0	0	0	0
Doc5	0	0.03	0.27	0.11	0.15	0	0.1	0	0	0	0	0	0.11	0	0	0	0.07	0	0.15	0
Doc6	0.02	0	0	0	0	0.98	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Doc7	0	0	0.22	0.5	0.05	0	0	0	0	0	0	0	0	0	0	0	0	0.22	0	0
Doc8	0	0.14	0	0	0	0	0	0	0	0	0.04	0	0	0.77	0	0	0	0	0.04	0
Doc9	0	0	0.04	0	0	0	0	0	0	0	0.06	0	0	0.68	0	0	0	0	0.21	0
Doc10	0	0	0.14	0.22	0	0	0	0.04	0	0	0.11	0	0	0.42	0	0	0	0	0.06	0
Doc11	0	0.01	0	0	0.01	0	0.09	0.03	0	0	0.06	0	0.02	0.29	0	0	0	0	0.33	0.17
Doc12	0	0	0.14	0	0	0	0	0.31	0.01	0	0	0	0.03	0.51	0	0	0	0	0	0
Doc13	0	0	0	0.08	0	0	0	0	0.02	0	0.06	0	0	0.84	0	0	0	0	0	0
Doc14	0	0	0	0	0	0	0.68	0.24	0	0	0.05	0	0	0	0	0	0	0	0.03	0

Fig. 5. Figure 2: The distribution of the first 15 documents.

We use pyLDAvis for view the topics-keywords distribution.

A good topic model will have non-overlapping, fairly big sized blobs for each topic.

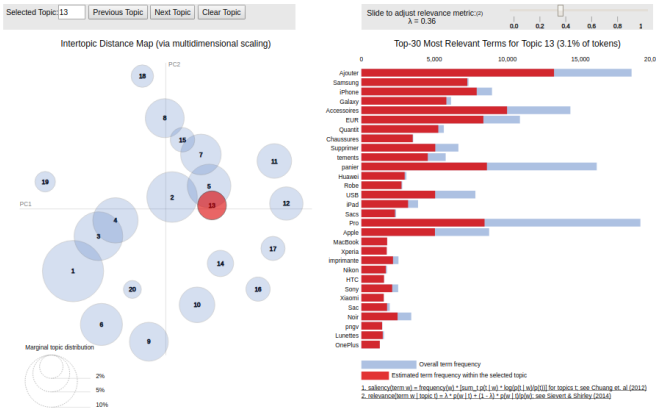

Fig. 6. Figure 3: Visualisation of LDA performance

We can see that most of the circles defining a topic do not overlap. When this is not the case, the intersection is not very significant. Our model seems to have worked well.

Features selection. For feature learning methods embeddings we take all features provided. we use only features selections methods on hancraft one. first of all we remove higly variable

3. Model evaluation and tuning

I use well known machine learning classifier on the resulting features to evalute which one predict the best. Such as SVM, KNN, Logistic Regression, Ensemble Learning and Random Forrest, Multilayer Perceptron, Radial Basis Function network, and Naive Bayes. The training data for the Random Forests was obtained by creating eight independent validation sets, each consisting of 4,000 data points (2,000 real edges and 2,000 fake edges). These validation sets were constructed to approximate the distributions of nodes and edges seen in the test set. Each validation set had 4,000 unique v_a nodes, while v_b nodes could be duplicates. The best model was xgboot and we find tune it using gradient boosting

4. Implementation and Result

We encountered huge concerns in using the tf-idf results to reduce the size of documents using a pre-trained Word2Vec template. This inability to perform this task is due to the fact that we have over a million unique words representing all the text except that most of the pre-trained templates are done on about 50 thousand words. Their use would have grossly reduced the diversity of the texts. So we turned to the LDA which gives us satisfactory results but this required several changes in parameters such as the number of topics.

We compute handcraft with python using numpy and networkX. Extracted features were computationaly expensive. We use multithreading but it still take a long time. That's because some nodes are really big. we run get out of memory and it stop while computing. Finaly, we leave some higly computing features or use subgraph of first neighborhood to compute them. We end up with feature of size 33 list above well define by Cukierski et al. (2011):

- Jaccard
- Adar
- Cosine similarity
- Common neighbors
- Ressource allocation
- kNN 1,2 3, 4
- indegree of source and target
- outdegree of source and target
- SVD nodes, dot, means
- SimRank (subgraph)
- Katz
- Pagerank of source and target
- Shortest paths
- hubs and autorithy
- ...

The two approach we talked about give me same range of f1-score. The best submission was obtained by learning random forests. Classifier like xgboost tend alway to overfit the training data by achieving high score on it but poor one the test set. With feature learning one we see a lot of methods. We try the one we cite above. And the result were slightly less better than handcraft one. In the result shown below we have not yet used text feature. We are confident that it would improve the result.

5. Conclusion

We take advantage from the feature extraction techniques, supervised learning models to approach the task of link prediction as a classification method and combine node attributes to the similarity based metrics as the model inputs. We learn a lot about graph theory and topology and we also see how algebra are usefull in graph theory. In machine learning aspect, we try as best as we can to apply what we learn along this semester. It was a great opportunity to deal with machine learning technics and also nlp. As everyone we take time debugging stuff but finally we are proud of what we did.

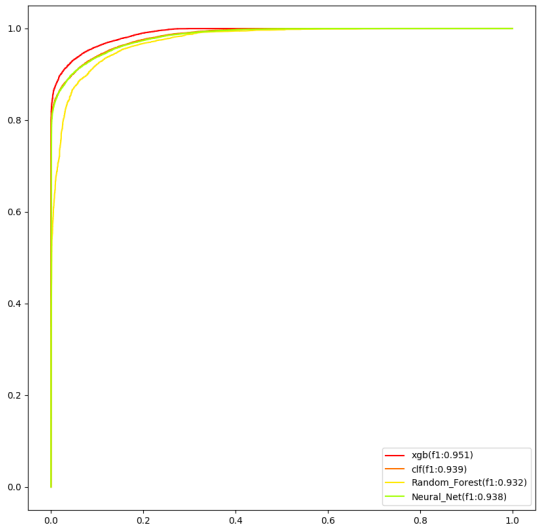

Fig. 7. Figure 3: Roc curve of classifier with predict proba

References

Abu-El-Haija, S., Perozzi, B., and Al-Rfou, R. (2017). Learning edge representations via low-rank asymmetric projections. *CoRR*, abs/1705.05615.

Cukierski, W., Hamner, B., and Yang, B. (2011). Graph-based features for supervised link prediction. pages 1237–1244.

David Liben-nowell, J. K. (2004). The link prediction problem for social networks.

Ou, M., Cui, P., Pei, J., Zhang, Z., and Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. pages 1105–1114.

Tsitsulin, A., Mottin, D., Karras, P., and Müller, E. (2018). VERSE: versatile graph embeddings from similarity measures. *CoRR*, abs/1803.04742.