Social Network Analysis using RLVECN: Representation Learning via Knowledge-Graph Embeddings and Convolutional Neural-Network

Bonaventure C. Molokwu*, Ziad Kobti
School of Computer Science, University of Windsor, Windsor - Ontario, Canada
{molokwub, kobti}@uwindsor.ca

Abstract
Social Network Analysis (SNA) has become a very interesting research topic with regard to Artificial Intelligence (AI) because a wide range of activities, comprising animate and inanimate entities, can be examined by means of social graphs. Consequently, classification and prediction tasks in SNA remain open problems with respect to AI. Latent representations about social graphs can be effectively exploited for training AI models in a bid to detect clusters via classification of actors as well as predict ties with regard to a given social network. The inherent representations of a social graph are relevant to understanding the nature and dynamics of a given social network. Thus, our research work proposes a unique hybrid model: Representation Learning via Knowledge-Graph Embeddings and ConvNet (RLVECN). RLVECN is designed for studying and extracting meaningful representations from social graphs to aid in node classification, community detection, and link prediction problems. RLVECN utilizes an edge sampling approach for exploiting features of the social graph via learning the context of each actor with respect to its neighboring actors.

1 Introduction
Human habitat is comprised of several systems and ecosystems; and interaction is a natural phenomenon obtainable in any given system or ecosystem. Interaction between constituent entities in a given system/ecosystem is a strategy for survival, and essential for the sustenance of the system/ecosystem. Social graphs are non-static structures. Analyzing and learning underlying knowledge from communities, comprising social actors, using given sets of standard still remain a crucial research problem in SNA. Hence, we have proposed RLVECN which is a hybrid model for classification-based and prediction-related problems in social networks.

On one hand, the classification of nodes induces the formation of cluster(s). Consequently, clusters give rise to homophily in social networks. On the other hand, the prediction of links brings about correlations and/or ties formation; which increases the tendency for transitivity in social graphs. RLVECN is based on an iterative learning approach. RLVECN aims at solving the problems of node classification, community detection, and link prediction in SNA using an edge sampling strategy. Basically, learning in RLVECN is effectuated via semi-supervised training. The architecture of RLVECN comprises two (2) distinct representation-learning layers, viz: an embedding layer and a Convolutional Neural Network (ConvNet) layer [Molokwu, 2019]; which are trained by means of unsupervised learning. These layers are essentially feature-extraction and dimensionality-reduction layers where viable facts are automatically extracted from the social networks [Molokwu and Kobti, 2019]. The embedding layer projects the feature representation of the social graph to a q-dimensional real-number space, \( \mathbb{R}^q \). This is accomplished by associating a real (number) vector to every unique actor in the social network; such that the cosine distance of any given tie (a pair of actors) would capture a significant degree of correlation between the two associated actors or nodes. Additionally, the ConvNet layer feeds on the embedding layer; and it is responsible for further extraction of inherent features from the social graph. With reference to RLVECN’s architecture, a classification layer succeeds the representation-learning layers; and this layer is trained by means of supervised learning. The classifier is based on a Neural Network (NN) architecture assembled using deep (multi) layers of stacked perceptrons [Goodfellow et al., 2017]. Every low-dimensional feature \( \{X\} \), extracted by the representation-learning layers, is mapped to a corresponding output label \( \{Y\} \); and these \( \{X, Y\} \) pairs are used to supervise the training of the classifier such that it can effectively and efficiently learn how to classify actors, identify clusters, and predict links within the given social graph.

Furthermore, we have evaluated RLVECN against an array of state-of-the-art methodologies and Representation Learning (RL) models which serve as our baselines. Thus, the baselines used herein with respect to the node-classification tasks are:

(i) DeepWalk: Online Learning of Social Representations [Perozzi et al., 2014].
(ii) GCN: Semi-Supervised Classification with Graph Convolutional Networks [Kipf and Welling, 2017].
(iii) LINE: Large-scale Information Network Embedding [Tang et al., 2015].
(iv) Node2Vec: Scalable Feature Learning for Networks

*Contact Author
We have modelled the open problems in SNA solved herein while the fitness/utility was measured based on the following features of each social network representation. Conclusively, the baselines used herein for the link-prediction tasks:

(i) ComplEx: Complex Embeddings for Simple Link Prediction [Trouillon et al., 2016] [Lacroix et al., 2018].
(ii) ConvKB: A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network [Nguyen et al., 2018].
(iii) DistMult: Embedding Entities and Relations for Learning and Inference in Knowledge Bases [Yang et al., 2015].
(iv) HoE: Holographic Embeddings of Knowledge Graphs [Nickel et al., 2016].

Also, the baselines benchmarked herein, is primarily attributed to its ground-truth samples per class for each benchmark dataset.

2 Proposed Methodology and Framework

Algorithm 1 Proposed Node Classification Algorithm

Input: \{V, E, Y_{lbl}\} \equiv \{Actors, Ties, Truth Labels\}
Output: \{Y_{pred}\} \equiv \{Predicted Labels\}

Preprocessing:
// V_{lbl} : Labelled actors // V_{ulb} : Unlabelled actors
V_{lbl}, V_{ulb} \subset V = V_{lbl} \cup V_{ulb}
E : (u_{i}, v_{j}) \in \{U \times V\} // (u_{i}, v_{j}) \equiv \{source, target\}
// |E_{train}| = \sum \text{indegree}(V_{lbl}) + \sum \text{outdegree}(V_{ulb})
E_{train} = E_{t}: u_{i}, v_{j} \in V_{lbl}
E_{pred} = E_{p}: u_{i}, v_{j} \in V_{ulb}

Training:
for t \leftarrow 0 to \mid E_{train} \mid do
  f_{t} : E_{t} \rightarrow [X \in \mathbb{R}^{q}] // Embedding operation
  f_{t} \in F = (K * X)^{t} // Convolution operation
  r_{t} \in R = g(F) = \text{max}(0, f_{t})
  p_{t} \in P = h(R) = \text{maxPool}(r_{t})
  f_{c} : \Theta: p_{t} \rightarrow Y_{lbl} // MLP classification operation
end for

return Y_{ulb} = f_{c}(E_{pred}, \Theta)

Algorithm 2 Proposed Link Prediction Algorithm

Input: \{V, E, B_{gTruth}\} \equiv \{Actors, Ties, Truth Entities\}
Output: \{B_{pred}\} \equiv \{Predicted Entities\}

Preprocessing:
B_{gTruth} : \{0, 1\} \equiv \{-ve/False tie, +ve/True tie\}
E = E_{+ve}s \cup E_{-ve}s
// E_{train} : Ground-Truth edgelist // E_{pred} : E_{train}
E_{train} = E : u_{i}, v_{j} \in \{V \times V\} \subset \{V \times V\}
E_{train} = E_{t} : E \rightarrow B_{gTruth} // |E_{train}| = E - E_{pred}
E_{pred} = E - E_{train}

Training:
while E_{train} \neq NULL do
  f : E_{t} \rightarrow [X \in \mathbb{R}^{q}] // Embedding operation
  f_{t} \in F = (K * X)^{t} // Convolution operation
  r_{t} \in R = g(F) = \text{max}(0, f_{t})
  p_{t} \in P = h(R) = \text{maxPool}(r_{t})
  f_{c} : \Theta: p_{t} \rightarrow B_{gTruth} // MLP: \Theta = \text{similarity}(u_{i}, v_{j})
end while

return B_{pred} = f_{c}(E_{pred}, \Theta)

Acknowledgments

This research was supported by International Business Machines (IBM), SHARCNET and Compute Canada.

References


